memopad

Memotech Computer User Club Magazine

Christmas Issue 4

CONTENTS

£1.35

Editorial

It is always difficult taking over a post that has been occupied by someone else. It's even more daunting to follow in the footsteps of Keith, who not only founded the magazine, but has also brought Memopad to new boundaries in User Group professionalism. However, I can but try ....

Christmas sales for Memotech related items have reached new heights. The link of magazine advertising and the concerted effort made by members has seen Memotech sales jump through the roof. Memotech are trying to fulfill all outstanding orders before the Christmas recess, and after a brief discussion with them, this morning, it is possible that no one will be disappointed.

With luck, and a little sense on the part of Memotech, we may see an increase in advertising in the coming new year. If Memotech don't realise that their increased prosperity is directly due to advertising then they have reached a new low in business acumen and are doomed to failure.

With the massive figures experienced by Dixons in the sale of the Amstrad word processing package Memotech would be wise to reconsider their pricing policies, and I am positive that it is not beyond the Company to come up with a similar package similar in price. One outstanding point with the Amstrad package is the fact that the ordinary, small business man can plug in the computer and use the package, to do something useful, immediately, or at least within the time it takes to familiarise oneself with the instruction manual. Computer companies should be aware that an awful lot of small businesses do not want to become computer fanatics.

When I was talking to Keith, yesturday, he mentioned that he is convinced that he has managed to secure the educational software, originally written for the Russian package, and hopes that Scisoft will release it within the very near future.

Manic Miner & Jet Set Willy are now on release and can be obtained ex-stock - a long time from Software Projects original promise, but they are now available.

One startling fact has emerged over the past few months ..... instead of club membership shrinking - due to selling off and moving on to new models - membership is still increasing weekly !

All at Syntaxsoft & Genpat would like to take this opportunity of wisheing each of you a MERRY CHRISTMAS & A HAPPY NEW YEAR. We hope that Santa brings you all you wished for .....

Keep those fingers moving and make this Christmas a BLACK ONE !

TM

GENPAT HIT LIST....

arcade

1.	SEPULCRI SCALERATI	MEGASTAR
2.	SUPA CODER	SYNTAX
3.	ROLLER BEARING	MEGASTAR
4.	QUAZZIA	MEGASTAR
5.	USER BASIC	SYNTAX
6.	FELIX IN THE FACTORY	MICROPOWER
7.	OBLITERATION ZONE	MEGASTAR
8. *	MANIC MINER	SOFTWARE PROJECTS
9.	EDASM	SYNTAX
10.*	JET SET WILLY	SOFTWARE PROJECTS

* DUE TO BACK ORDERS NOW RELEASED....

adventure

1.	SNOWBALL	LEVEL NINE
2.	LORDS OF TIME	LEVEL NINE
3.	EMERALD ISLE	LEVEL NINE
4.	THE KEYS TO TIME	SENTIENT
5.	MURDER AT THE MANOR	SENTIENT

educational

1.	HELLI-MATHS	SENTIENT
2.	MATHS 1	CONTINENTAL
3.	WORDS & PICTURES	CONTINENTAL
4.	SPELLI-COPTER	SENTIENT
5.	FIRST WORDS	CONTINENTAL

High Scores

Can you do better ?

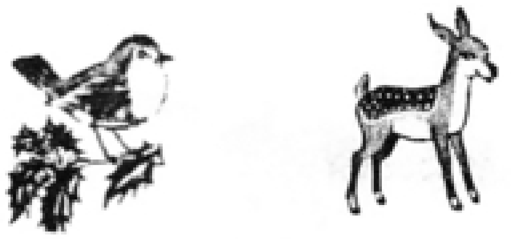
HIGH SCORES : HIGH SCORES...Can you do better ??

GOLDMINE	16,300+	P. Howard
ASTRO-PAC	185,990	Richard Nash
BOUNCING BILL	128,142	Alan Dobson
SNAPPO	128,688	Richard Franks
KNUCKLES	999,999+	Sally Street
CHAMBEROIDS	Completed 6 mins	M. Allcorn
NERO	17,610	Richard Nash
COBRA	8,924	Richard Nash
MISSION ALPHATRON	88,250	T. Eriksson
TAPEWORM	175,980	Richard Frnaks
TOADO	178,292	Gavin Gaunt and Nicholas Locke
POT HOLE PETE	108,830	Richard Franks
MAXIMA	1,479,710	S. Olander
STAR COMMAND	140,430	Ian Nichols
PHAID	28,000	Sally Street
OBLOIDS	82,400+	M. Hurley
KILOPEDE	82,253	Richard Nash
3D TACHYON FIGHTER	10,700	Lesa Woodger
CONTINENTAL RAIDERS	108,240	Sean Haverty
BLOBBO	148,283	Elizabeth Mahon
QUANTUM	?	Andrew Miller
GOGO 2	205,000	R. Siddall
MINEFIELD	1,500	David Nash
FLUMMOX	239,550	Andrew Miller
TURBO	15,630	N. Crighton
FATHOMS DEEP	2,300	Leslie Banks
ACROVATOR	675,000+	P. Howard
FIREHOUSE FREDDIE	29,620	T.Eriksson
GOGO	43,960	T. Eriksson
ARCADIANS	25,900	Adrian Joynson
MISSILE COMMAND	27,580	Adrian Joynson
LITTLE DEVILS	34,320	Leslie Banks
FELIX IN THE FACTORY	14,740	Peter Crighton
HUNCHY	7,908	R.Harmer
SON OF PETE	8,051	Gavin Gaunt
HAWKWARS	16,600	T. Eriksson
ESCAPE FROM ZARCOS	48 Items	P+G Howard
SALTY SAM	40,842	Andrew Johnson
MISSION OMEGA	9,350	R.Harmer
ICEBURG	17,431	Alan Dobson
SNOWBALL	450	P. Crighton
EMERALD ISLE	300/1000	Richard Franks
SUPERBIKE	23.9mms	B. Clark
REVERSI	beaten by 47	Richard Franks
DOODLEBUG	3,820	Andrew Miller
DR. FRANKIE	54,425	S. Olander
TARGET ZONE	14,580	P. Howard
MINER DICK	22,520	R. Siddall
JUMPING JACK	26,120	Andrew Miller
SURFACE SCANNER	35,830	P. Howard
CAVES OF ORB	421/500	P. Korsten
SEPULCRI SCELERATI	7,925	Andrew Miller GAME COMPLETED
S.M.G.	8,140	Andrew Miller
COMBAT	42,310	Andrew Miller
QUAZZIA	28,680	Andrew Miller

Mike Nash has completed Gogo 2 and has quoted the final message - "At last, you have found the Gogon diamonds"

Can you beat these high scores ? Do you have a high score for a game not mentioned above ?

SOFTWARE REVIEW



REVIEW - MTX PURCHASE LEDGER

SYNTAXsoft

The purchase ledger program takes about 4 minutes to load. It is designed for use with the MTX 512 . It holds up to 35 suppliers/creditors accounts and each with 30 records (i.e. Invoices, Credit Notes and Bank Payments). I don't think that 30 records are enough for real life applications.

Anyway, when the program has finished loading, you are asked to key in a password or entry code and the date before access to the main menu. The entry code "Login Purchl" is already set for you, but can be easily changed. This is explained in the user instruction leaflets.

The function key 'F1' is for creating details of your creditors - i.e. company name, address,credit limit, opening balance, etc. All these can be looked at at any time by the 'F5' key, but you have to remember the creditor's account number (which was given when setting up new account).

Amending and/or deleting creditor's account can be easily done using the 'F3' key. But if, for example, you are changing the address from "ABC Avenue" to "ABC Park", you could end up with the new address as "ABC Parkue". If you didn't make sure that the rest of the field is cleared out after keying in the new data! You'll also be able to delete a creditor account provided it shows a nil balance.

The next important step is to analyse your expenses - i.e. whether they are raw material purchases, rent & rate, insurance, travelling expenses, etc. The program allows up to twenty types of expwense (from P10 to P200). You must do this before trying to post any transactions to the ledger. Description of these expense codes can be changed, but must not exceed 15 characters.

You may now proceed to the 'F8' key which is the purchase ledger input menu. There is no need to show a minus sign as a credit item. The program is well aware of this. If you are inputting invoices or credit notes, you need to know the type of expense they relate to. If you are inputting cash payments, you need to give it a payment number (which can be your cheque number or any other reference number). There is no matching of invoice & amount and you could well make mistakes and end up overpaying your creditor! However, the 'F6' key displays creditor's balance and should therefore be checked before sending out cheques. Another drawback is that there is no allowance made for payments with deduction of discounts claimed or agreed. To do this, you may have to create a credit not entry together with the cash payment - the two being equal to the invoiced amount.

Data can be saved to & loaded from tape and hard copy obtained via the printer. A particular creditor's account can be date sorted and complte records can be allocated. These are all quite clearly indicated in the user instructions leaflets.

Overall, this is a good package. It's easy to use and is certainly good value for money. I would, however, like to see a disc version of the program with improvements made to it. ★

40 Column Graphics *Peter Knaggs*

FORTY COLUMN GRAPHICS

The reason for this article is three fold:
Firstly, I promised Keith something like this a little time ago, and my not being one to go back on a promise, well here it is.
Secondly, It's a subject and program that you all might find useful.
Finally, It means that I have got to document the damned program for you lot. This is something that I don't like doing, but is useful.

In a recent release (or escape) of the comic I noticed a program that would be of use to a lot of people. Obviously it is, as so many have already written this program. It is of course the old 40 column text on a graphic screen problem.

Well, after the last program I saw to do it, which used miles of BASIC and redefined characters to get the desired effect. I decided that there must be a simpler way.

What I wanted was for both the CSR and PRINT commands to operate in a 40 column environment, but in graphic mode. To do this, I would have to define two new commands. Well, Memotech being them, have ruled out using the USER command. If I was to use it then I would have to disable all of the disc functions (for those of you who do not have disc, the USER command is used to operate the disc drive).

So me, being me, decided to use another command altogether. It is a command that is rarely used and is simple to add into. I am of course talking about the NODE command, this is only used in the communications room, as far as I know of.

Well the NODE or my NODE command would have to do two different things, the equivalant of a CSR command and a PRINT command. Well this is quite a simple problem to solve. You simply type NODE CSR or NODE PRINT.

O.K.,but what if you have the communications ROM. Well, that is solved by taking the 3 bytes at NODLOC (#FA9B) and placing them at the end of my coding. This means that if I do not find the CSR or PRINT after the NODE then I pass the command on down the line for the real NODE command. In this manner you can have my NODE command and still have the old one.

The coding for the NODE command in the ROM goes something like this:

```
          CALL  NODLOC
          RET
```

On entry to NODLOC the only thing of any interest is the register pair DE.
These are pointing to the next character after the NODE command. This character could be a letter or command word, so this is how we can tell what comes after the command, be it CSR, PRINT or anything else.

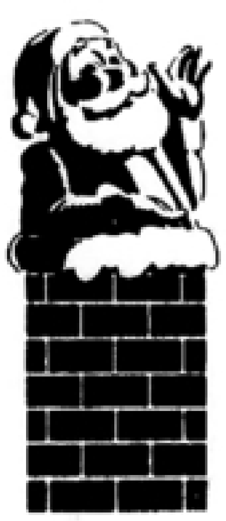
Well, that's one big secret out of the way. Now I'm going to give you another one. (Not that it matters much. You won't get to read this until after SWINDON when all will know).

I do have a MEMOTECH MTX500 for which I have the RS232's. This is in fact on loan to me from K.H. I do most of my programming on another, rather obscure, system. I then down load the programs into the Memotech to test them. (That would make quite a good article in itself) Now, the reason I am telling you all this is so I can use a listing of the Machine Code file that I send to the Memotech rather than the Memotech one. (With mine it is easier to describe what is happening). You should have enough information to either "POKE" or "PANEL" the program into a line or at least type it in (without comments).

If you are confused, as I'm sure you are, then please get in touch with me. I'll be only too happy to help. If you enclose a SAE with a little bit of the old spondulix, then you might persuade me to send you a listing from the Memotech.
My address, as if you don't know by now,is:
P.Knaggs, 12 Seymour Rd., Chippenham, Wilts. SN15 3NH.★

```
                    ;Forty coloum graphics for Memotech MTX
                    ;25/08/85 P.Knaggs
                    ;Assembled on: 30/08/85 at 1048
                    ;
                    ;
                    ;**********************
                    ;***  INITLISATION  ***
                    ;**********************
                    ;
                    ;Start code hear
                    CURVS:   EQU  0FF5B         ; Pointer to current VS data
                    CALCST:  EQU  0FA81         ; Top of Calculater stack
                    NODEVEC: EQU  0FA9B         ; NODE command Vector address
                    ;
                    ;Initlis NODE command
                    ;Code at NODE in Rom is:
                    ;          RET              ; End of previous command
                    ;          DB 7             ; Syntax code byte
                    ;          CALL 0FA9B       ; Proform the function
                    ;          RET              ; Return to basic for next command
                    ;Code at FA9B (NODEVEC) normaly is: RET
                    ; or a JP if the comunications ROM has been inistaled.
                    ;
4007 212540         EXEC:     LD   HL,MYVEC     ; Point to end of NODE coding
400A 119BFA                   LD   DE,NODEVEC   ; Point to the NODE vector
                    ;
                    ;Copy the code at the NODE VECtor to the end of my coding
                    ;Also Sets to NODE command to come to my NODE coding
                    ;
400D 0603                     LD   B,3          ; Set Number of bytes to copy
400F 1A             ILOOP:    LD   A,(DE)       ; Get value from NODE Vector
4010 F5                       PUSH AF           ; Save it
4011 7E                       LD   A,(HL)       ; Get the code from my NODE coding
4012 12                       LD   (DE),A       ; Set the NODE vector to it
4013 F1                       POP  AF           ; Recover the old NODE vector value
4014 77                       LD   (HL),A       ; Set MYVEC to it
4015 23                       INC  HL           ; Move on to the next byte
4016 13                       INC  DE           ; -------- ' ' --------
4017 10F6                     DJNZ ILOOP        ; Repeat for all 3 bytes
                    ;
4019 C9                       RET               ; Exit, return to Basic
                    ;
                    ;*****************************************************************
                    ;
                    ;******************************
                    ;***  NODE command coding  ***
                    ;******************************
                    ;
                    ;Syntax:   NODE command (pramiters)
                    ;Function: Use my Forty column graphic screen coding
                    ;          Command my be CSR or PRINT
                    ;
                    ;
401A 1A             NODE:     LD   A,(DE)       ; Get the command toggle
401B 13                       INC  DE           ; Move to the next byte
401C FE8E                     CP   08E          ; Is it the CSR toggle ?
401E 2808                     JR   Z,CSR        ; Yes => Preform the CSR coding
4020 FE90                     CP   090          ; No => Is it the PRINT toggle ?
4022 2818                     JR   Z,PRINT      ; Yes => Preform the PRINT coding
4024 1B                       DEC  DE           ; No => Move the pointer back a byte
4025 C31A40         MYVEC:    JP   NODE         ; Do what ever I replaced
                    ; This is done so as the normal NODE command will stil
                    ; operate.  The only time it will not is when you call up
                    ; my screen commands via NODE CSR and NODE PRINT.
                    ;




                    ;
                    ;*********************
                    ;***  CSR coding  ***
                    ;*********************
                    ;
                    ;Syntax:   NODE CSR x,y
                    ;Function: Place cursor at x,y
                    ;
4028 DDE5           CSR:      PUSH IX           ; Save Basic pointer
402A F7                       RST  30           ; Extract the First number: X
402B F5                       PUSH AF           ; Save the value
402C F7                       RST  30           ; Extract the Second number: Y
402D DD2A5BFF                 LD   IX,(CURVS)   ; Find start of current VS data
4031 3C                       INC  A            ; Add 1 to Y value
4032 DD7702                   LD   (IX+2),A     ; Set the current ROW location
4035 F1                       POP  AF           ; Recover the X value
4036 DD7701                   LD   (IX+1),A     ; Set the current COLUMN location
4039 DDE1                     POP  IX           ; Recover the Basic pointer
403B C9                       RET               ; Return to Basic for next command
                    ;
```

```
                  ;*******************************************************************************************
                  ;
                  ;***********************
                  ;***   PRINT coding   ***
                  ;***********************
                  ;
                  ;
                  ; This for the most part is a copy of the ROM coding for the
                  ; PRINT Command.  My coding is at CHR
                  ;
                  ; Syntax:  MODE PRINT (item)
                  ; Function: Will display item at the current cursor position on
                  ;           the Graphic screen in forty column width.
                  ; Note:  TAB is not recommended.
                  ;        Characters from 0 to 31 will display:
                  ;           0 the last known cursor character.
                  ;           1 to 26 the user defined characters 129 to 154.
                  ;           27 to 31 are not recommended.
                  ;
                  *******************************************************************************************

403C EF           PRINT:    RST   28               ; Call up ENDLINE routine
403D B8                     DB    0B8              ;     => Are we at end of command ?
403E CA1B00                 JP    Z,0001B          ; Yes => Command ended
                  ;                                ;     Use rom to move to next line
                  ;                                ;     Code at 1B will print a
                  ;                                ;     Carage Return/Line feed pare
4041 FE3B         PRT:      CP    03B              ; Is character a ";"
4043 2004                   JR    NZ,NEXT          ; No => Get on with it then
4045 13                     INC   DE               ; Yes => Move to next character
4046 1A                     LD    A,(DE)           ;     Read the character in
4047 18F8                   JR    PRT              ;     Repeat until character not ;
                  ;
4049 EF           NEXT:     RST   28               ; Call up ENDLINE routine
404A B8                     DB    0B8              ;     => Are we at end of command
404B C8                     RET   Z                ; Yes => Return back to basic
                  ;
404C D5                     PUSH  DE               ; Save line pointer
404D EB                     EX    DE,HL            ; Place it into HL
404E CD453C                 CALL  03C45            ; Do I print a Numbrer or a string ?
4051 D1                     POP   DE               ; Recover line pointer
4052 2804                   JR    Z,PRT2           ; Go if number
                  ;
4054 EF                     RST   28               ; Call up EVALS routine
4055 BB                     DB    0BB              ;     Evaluate string
4056 1807                   JR    PRT3             ; Display the string
                  ;
4058 EF           PRT2:     RST   28               ; Call up EVALAR routine
4059 80                     DB    080              ;     Evaluate number
405A D5                     PUSH  DE               ; Save pointer
405B CDC63F                 CALL  03FC6            ; Convert Number to string
405E D1                     POP   DE               ; Revocer pointer
                  ;
405F D5           PRT3:     PUSH  DE               ; Save Line pointer
4060 2A81FA                 LD    HL,(CALCST)      ; Get top of calculater
4063 CDEC3F                 CALL  03FEC            ; Make room on stack ?????
4066 ED5381FA               LD    (CALCST),DE      ; Reset top of calculater
406A CD7040                 CALL  DISPLAY          ; Display the string
406D D1                     POP   DE               ; Recover line pointer
406E 18CC                   JR    PRINT            ; Repeat until end of command
                  ;
                  ;Entry:BC =  Number of bytes to print
                  ;      DE => String to print
                  ;Exit: Z     Flag set
                  ;      BC =  0000
                  ;      DE => One character parst end of string
                  ;      A  =  00
                  ;
4070 78           DISPLAY:  LD    A,B              ; Quic way to say is BC = 0
4071 B1                     OR    C                ;     I.e., Is it end of string ?
4072 C8                     RET   Z                ; Yes => Exit
4073 0B                     DEC   BC               ; No => Decrement counter by one
4074 1A                     LD    A,(DE)           ;     Get the Character to print
                  ;                                ;     *** This is RST 28
4075 CD7B40                 CALL  CHR              ;                DB  0AC ; Print it
                  ;                                ;     *** in the ROM
4078 13                     INC   DE               ; Move to the next character
4079 18F5                   JR    DISPLAY          ; Repeat for all of counter
                  ;
                  ;*******************************************************************************************
                  ;
                  ; Now,this is where my coding comes into the program
                  ; Function: To print a character on to the current graphic screen
                  ;           at the current cursor location.
                  ; Entry: A  = Character
                  ; Exit:  No alterations to any registers
                  ;
                  *******************************************************************************************
```



```
407B DDE5      CHR:     PUSH IX          ;)
407D E5                 PUSH HL          ; )
407E D5                 PUSH DE          ;  )  Save all registers used
407F C5                 PUSH BC          ; )
4080 F5                 PUSH AF          ;)
               ;
4081 F5                 PUSH AF          ; Save the CHR code
4082 DD2A5BFF           LD   IX,(CURVS)  ; Point to the Current VS screen
               ;
               ;*** Set Pixal location ***
               ;Calculate Column value
4086 DD7E01             LD   A,(IX+1)    ; Get Cursor Column location
4089 4F                 LD   C,A         ; Multiply by 6 pixals per character
408A 87                 ADD  A,A         ;     x 2 = 2
408B 81                 ADD  A,C         ;     + 1 = 3
408C 87                 ADD  A,A         ;     x 2 = 6
408D C604               ADD  A,4         ; Now add another 4 pixals to bring
               :                         ;     it in from the left hand side
               ;                         ;     of the screen.
               ;Note: This can be removed.  In which case you can have 41
               ;and a half character to a line on the graphic screen.
408F 6F                 LD   L,A         ; Save the result
               :
```

```
               ******************************************************************************************

               ; Calculate the Row value
               ; As the cursor addressing is from the top left hand corner and the
               ; Pixel addressing is from the bottom left then the column value
               ; does not have to be altered.
               ; However, to find the correct Pixel location for the line we
               ; must make the line reference from the bottom left.  This I do in
               ; the code so as not to confuse us poor humans that have to
               ; write the Basic.
               ;
               ******************************************************************************************

               :
4090 DD7E02             LD   A,(IX+2)    ; Get the Row position
4093 D624               SUB  24          ; 24 becomes -1, 1 becomes -24
4095 ED44               NEG              ; Now make the number positive
                : Multiply by 8 lines per character
4097 87                 ADD  A,A         ;  x 2 = 2
4098 87                 ADD  A,A         ;  x 2 = 4
4099 87                 ADD  A,A         ;  x 2 = 8
409A 67                 LD   H,A         ; Save the result
               ;
               ******************************************************************************************

               ; Find CHARACTER DEFINITION  in the PATTEN TABLE in VRAM.
               ; This is where the VDP stores all its character patterns.
               ; The table starts at VRAM address #1800 and each character has
               ; an 8 byte pattern ( one for each line).
               ;

               ******************************************************************************************
               :
409B F1                 POP  AF          ; Recover the character code
409C E5                 PUSH HL          ; Save Pixal address
409D 6F                 LD   L,A         ; Set HL = Character code
409E 2600               LD   H,0
               ;Note: HL is used as the Acc is not big enough
40A0 29                 ADD  HL,HL       ; x 2 = 2
40A1 29                 ADD  HL,HL       ; x 2 = 4
40A2 29                 ADD  HL,HL       ; x 2 = 8
               :
40A3 110718             LD   DE,01807    ; Point to the end of the first CHR
40A6 19                 ADD  HL,DE       ; Add that to the table offset
40A7 C1                 POP  BC          ; Recover Pixal address
```

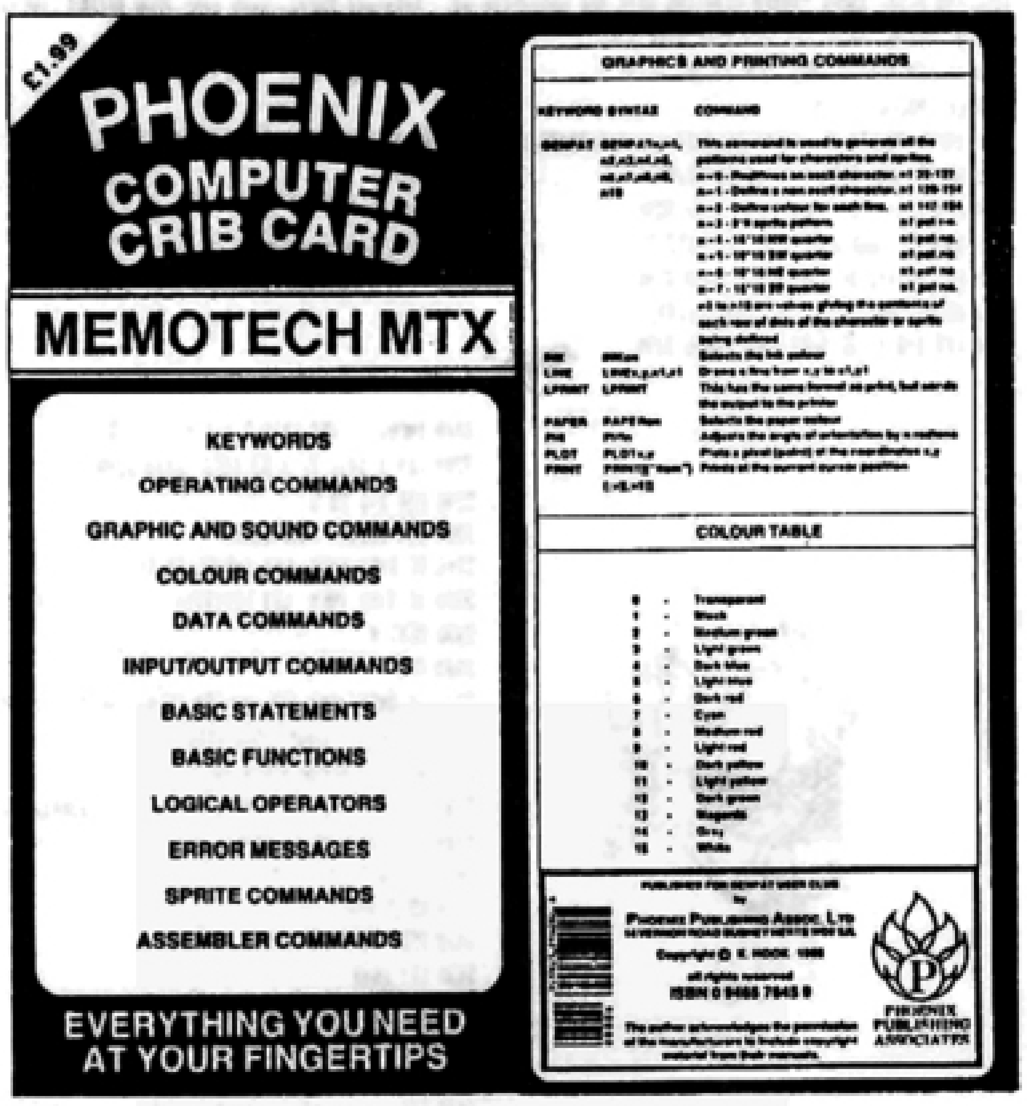
```
               ******************************************************************************************

               ; This is the bit that does all the work
               ; Entry: HL => Character definition in VRAM
               ;        B  => Bottom left Pixel: Row Value
               ;        C  => Bottom left Pixel: Column value
               ; Pixel refers to where to print the character.
               ;

               ******************************************************************************************
               ;
40A8 1608               LD   D,8         ; Set counter: 8 Lines
40AA F3        LOOP:    DI               ; Turn off the Interups
40AB 7D                 LD   A,L         ; Get LSB of VRAM location
40AC D302               OUT  (2),A       ; Tell it to the VDP
40AE 7C                 LD   A,H         ; Get MSB of VRAM location
40AF D302               OUT  (2),A       ; Tell the VDP, Read mode
40B1 2B                 DEC  HL          ; Move up a line (of patten)
40B2 E5                 PUSH HL          ; Delay for a bit waiting for the
40B3 E1                 POP  HL          ;      VDP to catch up
40B4 DB01               IN   A,(1)       ; Read the patten value
40B6 FB                 EI               ; Can have the interupts back on now
               ;
```

```
                ;A = Patten Line (Same as in GENPAT statment)
40B7 C5                 PUSH BC             ; Save Pixal location
40B8 1E06               LD   E,6            ; Set counter: 6 pixals acrose
40BA CB17       BLOOP:  RL   A              ; Do I plot the pixal or not
40BC 3008               JR   NC,UNPLOT      ; Not => Set ATTR to Unplot
                ;Set ATTRibuts to Plot the pixal
40BE D7                 RST  10             ; Yes => Call the Video code
40BF 84                 DB   084            ;     Print the next 4 characters
40C0 1B41               DB   01B,"A"        ;     ESCape A (is ATTR function)
40C2 3230               DB   "20"           ;     Pramiters 2,0 (Plot ink)
40C4 1806               JR   PLOT           ; Plot the pixal
                ;Set ATTRibuts to Unplot what might be under the character
40C6 D7         UNPLOT: RST  10             ; Call the Video code
40C7 84                 DB   084            ;     Print the next 4 characters
40C8 1B41               DB   01B,"A"        ;     ESCape A (is ATTR function)
40CA 3231               DB   "21"           ;     Pramites 2,1 (Plot paper)
                ;
40CC D7         PLOT:   RST  10             ; Call the Vido code
40CD 21                 DB   021            ; Function 1: PLOT
40CE C0                 DB   0C0            ; Pramiters:  C = Column
                ;                           ;             B = Row
40CF 0C                 INC  C              ; Move to next Column
40D0 1D                 DEC  E              ; Decrement pixal counter
40D1 20E7               JR   NZ,BLOOP       ; Repeat until out of pixals
40D3 C1                 POP  BC             ; Recover pixal column
40D4 04                 INC  B              ; Move up a line
40D5 15                 DEC  D              ; Decrement line counter
40D6 20D2               JR   NZ,LOOP        ; Repeat until out of lines
                ;
                ;move on to next print location
40D8 DD7E01             LD   A,(IX+1)       ; Get current column location
40DB 3C                 INC  A              ; Move right one character
40DC FE28               CP   028            ; Is it at column 40 ?
40DE 200E               JR   NZ,LF          ; No => Set new column
40E0 DD7E02             LD   A,(IX+2)       ; Yes => Get the Row number
40E3 3C                 INC  A              ;     Move to next line
40E4 FE18               CP   018            ;     Is it last line ?
40E6 2002               JR   NZ,ALLOK       ;     No => Set new line
40E8 3E01               LD   A,1            ;     Yes => start at line 1
                ;
40EA DD7702     ALLOK:  LD   (IX+2),A       ; Set current line/row value
40ED AF                 XOR  A              ; Reset to column 0 of line
                ;
40EE DD7701     LF:     LD   (IX+1),A       ; Set current column value
40F1 F1                 POP  AF             ;)
40F2 C1                 POP  BC             ; )
40F3 D1                 POP  DE             ;  ) Recover all registers
40F4 E1                 POP  HL             ; )
40F5 DDE1               POP  IX             ;)
                ;
40F7 C9                 RET                 ; Return to CALLer
                ;
                ;**********************************************************
                ;
                ;End of program
                ;
```

DUE TO THE POPULARITY WE HAVE REPRINTED THE CRIB CARD... SO WE NOW HAVE MORE STOCKS AVAILABLE OF THIS HANDY REFERENCE CARD.

£2.16p

Including Postage & Packing



Shared Bank Account *Bob Robinson* Part 2

```
2610 CSR 22,22: PRINT NA$
2620 IF NA$(I)=NM$(I) THEN  CSR 22,18: PRINT "       ": CSR 22,21: PRINT "       ": GOTO 2550
2630 PAUSE 1000: CLS : CSR 10,0: PRINT "STARTING DATES": CSR 10,1: PRINT "--------------"
2640 CSR 2,4: INPUT "Enter YEAR for start of entries      (4 Numerals only):  ";YEAR: LET YR=YEAR
2650 IF LEN (STR$(YEAR))<>5 THEN  GOTO 2630
2660 IF YEAR/4=INT(YEAR/4) THEN  LET LP=1 ELSE  LET LP=0
2670 LET LP2=LP
2680 CSR 2,8: INPUT "Enter name of month (first 3 letters only in Capitals) for start of earliestentries:           ";STMON$
2690 IF LEN (STMON$)<>3 THEN  GOTO 2680
2700 FOR I=1 TO 12
2710 IF LP=1 THEN  GOTO 2740
2720 IF MON$(I,1,3)=STMON$ THEN  LET STMON$=MON$(I): GOTO 2760
2730 GOTO 2750
2740 IF NOL$(I,1,3)=STMON$ THEN  LET STMON$=NOL$(I): GOTO 2760
2750 NEXT I
2760 LET MONU=I
2770 IF P<VAL(STMON$(4,3)) THEN  LET P=P+1: LET PP=P: GOTO 2770
2780 CLS : CSR 10,0: PRINT "INTEREST RATES": CSR 10,1: PRINT "--------------"
2790 CSR 2,4: PRINT "If Interest Rates do not apply to   Account, then Enter '0', otherwise    Press (RET)"
2800 INPUT ONT$: IF ONT$="0" THEN  GOTO 2820
2810 GOSUB 6110
2820 CLS : PRINT "  Credit entries by cheque usually     require to be cleared over a period of about 7 days."
2830 INPUT "  Enter the period that applies to thisaccount (in days):-  ";CL: LET CL=INT(CL+.5)
2840 IF CL=0 THEN  LET CLL=1 ELSE  LET CLL=CL
2850 CSR 0,10: PRINT " Interest starts to be paid immediatelyon entering credit cheques or only     after the clearance delay time."
2860 PRINT " If immediately, press 'I'.": PRINT " If delayed,     press 'D'.": PRINT : PRINT " (If no interest is paid,": PRINT " press either 'I' or 'D')"
2870 IF INKEY$="I" THEN  GOTO 2900
2880 IF INKEY$="D" THEN  GOTO 2900
2890 GOTO 2870
2900 IF INKEY$="I" THEN  LET IG=0
2910 IF INKEY$="D" THEN  LET IG=1
2920 GOTO 3380
2930 REM ## (FROM 7840,8230) #####
2940 REM  ## REVIEW OR CHANGING OF        INTEREST RATES   #####
2950 CLS
2960 IF ONT$="0" AND AP$="0" THEN  CSR 4,3: PRINT "NO AUTO-PAYMENTS NOR INTEREST": CSR 4,5: PRINT "INSTRUCTIONS HAVE BEEN ENTERED": PAUSE 4000: GOTO 1160
2970 IF ONT$="0" AND AP$<>"0" THEN  CSR 2,3: PRINT "NO INTEREST INSTRUCTIONS HAVE BEEN                 ENTERED": PAUSE 4000: GOTO 1160
2980 GOTO 6600
2990 REM ## (FROM 6670) ###
3000 IF ZZ<=6 THEN  LET Y=1 ELSE  LET Y=ZZ-5
3010 LET C=0
3020 FOR I=Y TO ZZ: LET C=C+1
3030 FOR W=1 TO 5
3040 LET RATE$(W)="     "
3050 LET D$=RIGHT$(STR$(RATE(I,W)),LEN (STR$(RATE(I,W)))-1): LET D$=LEFT$(D$,5)
3060 LET RATE$(W)=D$
3070 NEXT W
3080 PRINT ":";DATE$(I): CSR 9,(6+C): PRINT ":";RATE$(1): CSR 15,(6+C): PRINT ":";RATE$(2): CSR 21,(6+C): PRINT ":";RATE$(3): CSR 27,(6+C): PRINT ":";RATE$(4)
3090 CSR 33,(6+C): PRINT ":";RATE$(5)
3100 NEXT I
3110 PRINT "---------------------------------------"
3120 FOR I=1 TO 5: LET RATE$(I)="     ": NEXT I
3130 LET ZZ=ZZ+1
3140 CSR 0,14: PRINT "New interest rates apply immediately   after the last date which is entered inthe Account Display."
3150 PRINT "If no change is made, enter Chr.'@'."
3160 CSR 0,18: INPUT "ENTER STARTING DATE for operation of   Interest Rates, with year (FOR RECORD  ONLY, NOT >8 CHR$):      ";D$
3170 IF D$="@" THEN  LET ZZ=ZZ-1: GOTO 3340
3180 IF LEN (D$)>8 THEN  CSR 25,20: PRINT "             ": GOTO 3160
3190 LET DATE$(ZZ)=D$
3200 INPUT "ENTER NEW VALUE OF RATE 1:  ";RATE(ZZ,1): LET I=1: IF I=N2 THEN  GOTO 3290
3210 INPUT "AND RATE 2 ?:";RATE(ZZ,2);
3220 LET I=I+1: IF I=N2 THEN  GOTO 3290
3230 INPUT "  AND RATE 3 ?:";RATE(ZZ,3)
3240 LET I=I+1: IF I=N2 THEN  GOTO 3290
3250 INPUT "AND RATE 4 ?:";RATE(ZZ,4);
3260 LET I=I+1: IF I=N2 THEN  GOTO 3290
```

```
3270 INPUT "  AND RATE 5 ?:";RATE(ZZ,5)
3280 LET I=I+1: IF I=N2 THEN  GOTO 3290
3290 FOR I=1 TO 5
3300 LET LEV(I)=RATE(ZZ,I)
3310 IF I=N2 THEN  LET Z=RATE(ZZ,I)
3320 IF I>N2 THEN  LET LEV(I)=Z
3330 NEXT I
3340 CLS
3350 IF ONT$<>"0" AND AP$="0" THEN  CSR 2,3: PRINT "NO AUTO-PAYMENT INSTRUCTIONS HAVE            BEEN ENTERED": PAUSE 4000: GOTO 1160
3360 IF K$="C" THEN  GOTO 1160
3370 REM ### (FROM 2920) ###
3380 CLS : PRINT "  If Bank Charges are applied below a  specific Balance level, enter the levelat which this operates.If none applies,enter '0',";
3390 INPUT "in which case a caution      appears when the balance becomes       overdrawn:-  £";CH
3400 GOSUB 6950
3410 GOTO 5700
3420 REM  ## (FROM 4330)  ##
3430 LET FL=1
3440 REM ## (FROM 2010) ###
3450 LET LP4=LP2
3460 IF LP4=1 THEN  GOTO 3500
3470 FOR I=1 TO 13: IF R$(1,3,3)=MON$(I,1,3) THEN  LET L=VAL(MON$(I,4,3)): GOTO 3520
```

```
3480 NEXT I
3490 GOTO 3520
3500 FOR I=1 TO 13: IF R$(1,3,3)=MOL$(I,1,3) THEN  LET L=VAL(MOL$(I,4,3)): GOTO 3520
3510 NEXT I
3520 REM *** (FROM 3470,3490,3500) **
3530 LET L=VAL(R$(1,1,2))+L
3540 IF FL=1 THEN  LET I=L: RETURN
3550 IF FL=2 OR L>=PREL THEN  LET I=L+II: LET PREL=L: RETURN
3560 IF L<PREL THEN  LET II=II+365+LP4: LET AN=AN+1
3570 IF (YEAR+AN)/4=INT((YEAR+AN)/4) THEN  LET LP4=1 ELSE  LET LP4=0
3580 LET FL=2: GOTO 3460
3590 IF P<>(A(H)+CLL) THEN  GOTO 3670
3600 LET B5=B3
3610 REM ** (FROM 2350) ***
3620 IF F$(2)=MM$(1) THEN  LET G1=VAL(RIGHT$(F$,(LEN (F$))-2))
3630 IF F$(2)=NA$(1) THEN  LET G2=VAL(RIGHT$(F$,(LEN (F$))-2))
3640 IF F$(2)=MM$(1) THEN  LET B1=B1-G1
3650 IF F$(2)=MM$(1) THEN  LET B3=B3-G1
3660 IF F$(2)=NA$(1) THEN  LET B2=B2-G2
3670 IF F$(2)=NA$(1) THEN  LET B3=B3-G2
3680 LET B1=INT(B1*100+SGN(B1)*0.49)/100: LET B2=INT(B2*100+SGN(B2)*0.49)/100: LET B3=INT(B3*100+SGN(B3)*0.49)/100
3690 IF F$(2)=MM$(1) THEN  CSR 1,20: PRINT ":": CSR 7,20: PRINT ":";MM$;":";RIGHT$(F$,(LEN (F$))-2): CSR 24,20: PRINT ":";LEFT$(STR$(B1),9);
3700 CSR 38,20: PRINT ":"
3710 IF F$(2)=NA$(1) THEN  CSR 1,20: PRINT ":": CSR 7,20: PRINT ":";NA$;":";RIGHT$(F$,(LEN (F$))-2): CSR 24,20: PRINT ":";LEFT$(STR$(B2),9);
3720 CSR 38,20: PRINT ":"
3730 CSR 1,21: PRINT ":": CSR 7,21: PRINT ":JNT.:  :        :   :";LEFT$(STR$(B3),9): CSR 38,21: PRINT ":"
3740 GOSUB 5610
3750 REM ** (FROM 8370) **
3760 IF B3<CH AND B5<CH THEN  GOTO 3790
3770 IF B3<CH THEN  GOSUB 3950: GOSUB 4020
3780 IF B3<CH THEN  CSR 1,22: PRINT ":     :  ** BANK CHARGE ? **        :": CSR 0,23: PRINT
3790 RETURN
3800 REM  ** (FROM 2340,2440) ***
3810 IF F$(2)=MM$(1) THEN  LET G1=VAL(RIGHT$(F$,(LEN (F$))-2))
3820 IF F$(2)=NA$(1) THEN  LET G2=VAL(RIGHT$(F$,(LEN (F$))-2))
3830 IF F$(2)=MM$(1) THEN  LET B1=B1+G1
3840 IF F$(2)=NA$(1) THEN  LET B2=B2+G2
3850 RETURN
3860 REM ** (FROM 2340,2440) **
3870 LET B1=INT(B1*100+SGN(B1)*0.49)/100: LET B2=INT(B2*100+SGN(B2)*0.49)/100: LET B3=INT(B3*100+SGN(B3)*0.49)/100
3880 GOSUB 5610
3890 IF F$(2)=MM$(1) THEN  CSR 1,20: PRINT ":": CSR 7,20: PRINT ":";MM$;":  :";RIGHT$(F$,(LEN (F$))-2): CSR 24,20: PRINT ":";LEFT$(STR$(B1),9);
3900 CSR 38,20: PRINT ":"
3910 IF F$(2)=NA$(1) THEN  CSR 1,20: PRINT ":": CSR 7,20: PRINT ":";NA$;":  :";RIGHT$(F$,(LEN (F$))-2): CSR 24,20: PRINT ":";LEFT$(STR$(B2),9);
3920 CSR 38,20: PRINT ":"
3930 CSR 1,21: PRINT ":": CSR 7,21: PRIN: ":JNT.:  :        :   ]";LEFT$(STR$(B3),9): CSR 38,2;: PRINT ":"
3940 RETURN
3950 REM  ** DISPLAY HEADING (FROM 1850   2360,2480,3770,5950) ***
3960 FOR I=0 TO 7
3970 CSR 0,I: PRINT "                                        "
3980 NEXT I
3990 CSR 1,0: PRINT " ENTER DATE(as,e.g.,03JAN)+5 REF.CHR$."
4000 PRINT "  END WITH <RET>"
4010 RETURN
4020 REM  ** (FROM 1850,2130,2360,2480,   3770,4480,4760,5950) ***
4030 PRINT " ---------------------------------------"
4040 PRINT " :DATE :AC. :  £ ENTRY. :  £ BALANCE. :"
4050 PRINT " : +   :";MM$;":          :INDIVIDUAL.  :"
4060 PRINT " :5 REF: OR :WITHDRAWAL.:   :AVAILABLE:"
4070 PRINT " :CHR$ :";NA$;":  :CREDIT  :   : (JOINT) :"
4080 PRINT " +-----+----+--+--------+---+---------+"
4090 RETURN
4100 REM ** (FROM 2130) **
4110 FOR I=0 TO 7
4120 CSR 0,I: PRINT "                                        "
4130 NEXT I
4140 CSR 0,0: PRINT "ENTER SUM WITH LEADING "C"(CREDIT)OR"W"(WITHDRAW) AND "";MM$(1);""(";MM$;") OR "";NA$(1);""(";NA$;")"
4150 RETURN
4160 REM ** (FROM 1330,1340) **
4170 CLS : IF K$="L" THEN  GOTO 4200
4180 CSR 2,2: PRINT "If it is preferred to include a line space between each entry then": CSR 0,5: INPUT "Enter '1' otherwise Enter '0':  ";SP$
4190 IF SP$<>"0" AND SP$<>"1" THEN  CSR 32,5: PRINT "   ": GOTO 4180
4200 CLS : CSR 6,0: PRINT "LIST OF EARLIER ENTRIES"
4210 CSR 3,3: PRINT "EARLIEST STORED DATE IS ";C$(3,1,5);YEAR
4220 PRINT "   DATE OF LATEST ENTRY IS ";LEFT$(C$(E),5);YR
4230 CSR 2,11: PRINT "Enter required starting date in the    same form as shown above."
4240 CSR 2,16: PRINT " To continue a screen display, press    SPACE BAR: To finish, hold <RET>."
4250 PRINT
4260 PRINT "  IF FILE IS EMPTY, NO MONTHS OR DAYS  WILL BE SHOWN IN THE DATE.  PRESS <RET>"
4270 INPUT O$
4280 IF O$="" THEN  GOTO 1160
4290 IF LEN (O$)<9 OR LEN (O$)>10 THEN  GOTO 4270
4300 LET O=VAL(RIGHT$(O$,LEN (O$)-5))
4310 IF O<YEAR THEN  LET R$(1)="01"+STMON$(1,3)+STR$(YEAR): LET O=YEAR: GOTO 4330
4320 LET R$(1)="          ": LET R$(1)=O$
4330 LET LP3=LP2: LET LP2=LP: GOSUB 3420
4340 LET LP2=LP3: LET I=LP: LET Y=O-YEAR: LET Z=-1: LET W=0
4350 IF Y=0 THEN  GOTO 4400
4360 LET Z=Z+1
4370 IF (YEAR+Z)/4=INT((YEAR+Z)/4) THEN  LET I=1 ELSE  LET I=0
4380 LET W=W+365+I: LET Y=Y-1
4390 GOTO 4350
4400 LET D1=W+L
4410 LET Y=3
4420 IF D1>A(0) THEN  GOTO 4270
```

```
4430 IF D1>A(Y) THEN  LET Y=Y+1
4440 IF D1<=A(Y) THEN  GOTO 4460
4450 IF D1>A(Y) THEN  GOTO 4430
4460 CLS
4470 IF K$="P" THEN  GOTO 4810
4480 GOSUB 4770: GOSUB 4020
4490 FOR Z=Y TO E
4500 IF INKEY$=" " THEN  GOTO 4740
4510 REM **** (FROM 4760) ****
4520 LET KK$=J$(Z): LET Q$="": LET QQ$="": LET II=0: FOR I=1 TO 10
4530 IF ASC(KK$(I))=0 THEN  LET II=II+1
4540 NEXT I
4550 FOR I=1 TO 13-LEN (STR$(L(Z))): LET QQ$=QQ$+" ": NEXT I
4560 IF II=0 THEN  GOTO 4580
4570 FOR I=1 TO II: LET Q$=Q$+" ": NEXT I
4580 IF LEFT$(J$(Z),2)="C"+MM$(1) THEN  PRINT " :";LEFT$(C$(Z),5);":";MM$;":  :";RIGHT$(J$(Z),8)+Q$;":";L(Z);QQ$;":"
```

Cont....

Disc Mania & All About It by Len Clark

Disc storage from ASSEMBLY

One of the major attractions of the MEMOTECH computers is their built-in Z80 ASSEMBLY language. Another that has become available in recent months is the range of cheap disc drives that compete economically with any rival machines. The problem is that these are controlled wholly from BASIC and there is no guidance in the manual as to how they might be used from ASSEMBLY.
I recently bought a 250K SDX drive and have found a reasonably easy way of using them. It is not elegant and other readers may be able to suggest better approaches but, for the time being it works.

The basic idea

The disc drive is always controlled via the BASIC USER command, which has allowed MEMOTECH to add the extra commands required. What this does is jump to a short machine code routine starting at the hex address F5B3H. This resets the ROM page to page 3 and then calls the controlling routine stored on ROM in the separate disc controller unit. At this point the register pair DE contains the address of the first item after the USER command. The other registers do not affect the working of the disc drive, although they will be changed during the procedure and, if they contain values the programmer will want afterwards.

The basic idea is illustrated by the simple program in example 1. This mimics the BASIC command:

```
                                        USER DIR
```

Needless to say, it's hardly a useful program since, if you want to know what files are on disc, you're not going to spend time entering an ASSEMBLY program (however simple) when a direct command exists. The point is that it serves well as an initial illustration.

The line starting at the label DIR (address 4009h) mimics the direct command, except that USER is omitted. This is because we are fooling the computer to assume that it has just read the USER command and is now going on to the next byte. Thus:

```
          LD DE,DIR
```

will make the computer work from the program line in example 1. The command ends with #FF (FFh) which seems to be the usual End of Line marker and I'm fairly sure is required.
Try running this simple program on your system disc or one with a number of files.

Theoretically, the same principle applies to any of the disc commands and it certainly does work with simple commands like STAT and ERA. You can try them by changing the line labelled DIR in example 1 (make sure you don't want the file you erase!) You can even OPEN and CLOSE files from ASSEMBLY but, unfortunately, using most of the actual access commands is difficult because they also involve manipulating BASIC variables from ASSEMBLY. However, it is at this point that WRITE and READ come to our rescue.

Writing a fixed-length file

Example 2a allows the creation of a file of a fixed length - i.e. set by the program and only changeable by changing the program itself.

The command WRITE involves the simple transfer of a block of data from a specified address to the disc. All we need to know (and be able to declare to the disc control program) is:
(a) the location of the first byte of the block of data:
(b) the length of the data block (i.e. how many bytes in it).

I have called the file to be created TSTFILE.DMY but you can replace it with a more inspired name of your own. This program mimics the direct command:

```
        USER WRITE "TSTFILE.DMY",40960,17
```

Note that in the ASSEMBLY command line (in our case, WRITE starting at address 4009h) there must be no blanks. If you try writing a disc command in a numbered BASIC line you will see that the MTX monitor removes all spaces after the USER command. If you put a space in the ASSEMBLY line, you will get an error message.

In example 2a I have made sure that the file is closed by an EOF (End Of File) marker (1AH,26 decimal).

This is not strictly necessary: by the nature of the command, the discs will only save the number of bytes you specify. However, it does make for tidy disc handling. The EOF marker explains why we save 17 bytes even though we only handle 16 (10h) in the program. The 17th is the EOF marker itself. The value 40960 is A000h where the data block starts. Note that the values are given in decimal.

Once you have run the program you can see the effects of what you have done by entering directly:

```
USER TYPE <filename>
```

The EOF marker has the effect of erasing the preceding data byte on the screen, but be assured it is there. (You can check by entering PANEL and typing D (Display>) A000.) Alternatively it is a simple matter to devise a BASIC program to read the file.

READing a fixed length file

Example 2b reads the file we have created back in and displays its contents on the screen. Unfortunately, it is not simply a matter of reversing the process. If you use a READ line in the same way as DIR or WRITE in the previous examples, you get an error. I presume that, unlike DIR and WRITE, READ is a command that is known to the MTX monitor and so it automatically changes the word into a single byte key-word. (This will also be true of such commands as PRINT, INPUT and LINE INPUT.) Fortunately we can solve the problem by calling the READ control routine directly.

The bulk of the program is to do with calling this routine. The ROM page when we run ASSEMBLY is normally set to page 1 while the routine we want to call is on ROM page 3. Starting from address 4024h we:

save the present page configuration
change it to ROM page 3 while preserving the present RAM page
actually change the page configuration
call the routine
retreive the original page value (one return from the subroutine)
restore affairs to the original configuration.

Because we are calling the READ routine directly, the READ line starting at address 4009h is even more truncated than usual. We now omit both USER and READ. However, the first byte is a dummy value since DE is incremented in the subroutine. The READ control routine is located at 2D63h in ROM 3.

Files of variable length

The programs in example 2 are extremely simple and, where the amount of data to be saved is known, will prove very valuable. They have two major limitations.

1. Fixed Length

In many cases we do not know the size of the data we want to save or read until it is created. We need to have files of variable length.

2.Fixed Filename

Some applications might make good use of files of fixed length. An example that comes to mind is a directory of items with a known maximum length. In this case, however,we would need to be able to store an indeterminate number of such files. We would want the program to be able to create new file names such as DIRECTORY.0001 : DIRECTORY.0002 ETC ....

I think it is easy to see that this is relatively simple to do in principle by allowing the program to alter the command lines between each call of the disc control routine.

Examples 3a and 3b WRITE and READ variable files respectively. That I tackled variable file length rather than multiple files was just a matter of which task occurred to me first. Once the principle of one is grasped, the other ought to be fairly easy.

Note that example 3a involves the creation of two files. The first (LENFILE.DMY) stores the length of the main file in a format that makes future use easy. As before, TSTFILE.DMY stores the actual file. Example 3b therefore reads the length from LENFILE first and adjusts the command line accordingly so as to be able to read TSTFILE. A brief summary of the main parts of the programs follows:

1. <u>Example 3a</u>

4048h - 4068h Almost entirely to do with neat screen handling - the only significant matter is that DE is initially set at A000h in the subroutine SETSCR and address 405Eh stores the input character in memory.

406Ah - 407Ch The main part of the program, but it doesn't give much away.

4070h - 4082h Speaks for itself.

4083h - 4096h Avoids that pesky left-hand location on screen.

4097h - 40A2h Adds EOF marker to data block then calculates its length.

40A3h - 40C6h In conjunction with the first part of CNVDEC changes the actual value of the file length into individual decimal digits.

40C7h - 40D6h Controls CNVDIG, converts each decimal value returned into ASCII form and stores the length backwards at NUMSTR.

40D8H - 40F1H Tidies up and moves the decimal length to the WRITE control line.

40F2h - 4104h Sets the length in decimal as data for LENFILE.

4106h - 410Fh Fills any unused leading spaces with blanks.

2. <u>Example 3b</u>

403Eh - 4041h Read file length into 9FFAh.

4044h - 4056h Retrieve file length (skipping leading blanks) and store in read command line.

4058h - 405Dh Reads the actual file into A000h.

4060h - 4086h Displays the file just read.

4087h - 4090 Subroutine to handle pageing and call read control routine.

Postscript

These example routines do not do all the work for you. They show you what can be done and it is up to you to adapt them as is best suited to your needs.

I presume that it is possible to store information directly on disc and that this would save the ASSEMBLY programmer some of the precious kbytes swallowed by formatting. If that's the case, I would be grateful if some knowledgable computer-buff would give me the low-down. However, I think that the sort of approach I've outlined will still be of value partly because it involves relatively little actual programming (the controller ROM has done all the hard work for us) and partly because the resulting files are in a format that is compatable with BASIC manipulation. ★

```
10 REM EXAMPLE 1
20 REM A simple illustration of
30 REM   of accessing discs
40 CODE

4051           JR START
4053 DIR:      DB "DIR",£FF
4057 START:    LD DE,DIR
405A           CALL £F5B3
405D           RET
```

```
10 REM EXAMPLE 2A
20 REM WRITEing a file of set length
30 CODE

403C           JR START
403E WRITE:    DB "WRITE",£22,"TSTFILE.DMY",£22,",40960,17",£FF    ;The disc command.
405A START:    LD B,£10    ;Controls input
405C           LD HL,£A000         ;Stored at 40960
405F           RST 10      ;Provide a screen message
4060           DB £9F,"Type in a 16 letter message",£3A,£0D,£0A,£1E
4080 SETFLE:   CALL £0079 ;Read keyboard
4083           JR Z,SETFLE         ;Wait for keypress
4085           LD (HL),A  ;Store the value
4086           CALL £00BC ;Echo it to the screen
4089 DELAY:    CALL £0079 ;Wait for the key
408C           JR NZ,DELAY         ;  to be released
408E           INC HL
408F           DJNZ SETFLE
4091           LD (HL),£1A         ;Add End of File marker
4093           LD DE,WRITE         ;Set DE to command line
4096           CALL £F5B3 ;Disc control routine
4099           RET
```

```
10 REM EXAMPLE 2b
20 REM READing a file of set length
30 CODE

403B           JR START
403D READ:     DB £FF,£22,"TSTFILE.DMY",£22,",40960,17",£FF        ;Command line - first byte dummy
4055 START:    LD DE,READ ;Set DE to command line
4058           LD A,(£FAD2)
405B           PUSH AF     ;Save current page configuration
405C           AND £0F
405E           OR £30      ;Set for ROM 3
4060           LD (£FAD2),A
4063           OUT (£00),A         ;Activate ROM 3
4065           CALL £2D63 ;Call ROM 3 routine
4068           POP AF      ;Restore original page
4069           LD (£FAD2),A
406C           OUT (£00),A
406E           LD B,£10    ;Display 16 characters
4070           LD HL,£A000         ;  from 40960
4073 DISPLY:   LD A,(HL)
4074           CALL £00BC
4077           INC HL
4078           DJNZ DISPLY
407A           RET
```

```
10 REM EXAMPLE 3a
20 REM WRITEing a file of unknown length
30 CODE

4040          JR START
4042 WRITE:   DB "WRITE",£22,"TSTFILE.DMY",£22,",40960," ;Disc command
405B WRTEND:  DB "00000",£FF     ;Separate to make change easy
4061 LENGTH:  DB "WRITE",£22,"LENFILE.DMY",£22,",40954,6",£FF
407C NUMSTR:  DS 5         ;Work space
4081 VDULNE:  DW £5C01     ;Start of screen display
4083 VDUWTH:  DB £28       ;Length of screen line
4084 START:   CALL SETCSR
4087 NEWLNE:  CALL SETLNE
408A READIN:  DEC C        ;This is really just to give
408B          JR Z,NEWLNE          ;  a fairly neat screen display
408D INPUT:   CALL £0079
4090          JR Z,INPUT
4092          CP £0D       ;Your message (and file) will end with <RET>
4094          JR Z,SAVE    ;  rather like the BASIC command DSI
4096          LD B,A
4097          LD (DE),A    ;Store the character
4098          INC DE
4099          CALL £00BC
409C KEYUP:   CALL £0079
409F          JR NZ,KEYUP
40A1          JR READIN
40A3 SAVE:    CALL SETSTR
40A6          CALL CNVDEC
40A9          LD DE,LENGTH
40AC          CALL £F5B3
40AF          LD DE,WRITE          ;This is where the file
40B2          CALL £F5B3 ;  is actually loaded onto disc
40B5          RET
40B6 SETCSR:  RST 10
40B7          DB £1E       ;Gives a flashing cursor
40B8          LD DE,£A000          ;Start of store address
40BB          RET          ;  which could be anywhere in RAM
40BC SETLNE:  LD HL,(VDULNE)       ;This just controls screen display
40BF          LD A,L
40C0          OUT (£02),A
40C2          LD A,H
40C3          OUT (£02),A
40C5          LD A,(VDUWTH)
40C8          LD C,A
40C9          LD B,£00
40CB          ADD HL,BC
40CC          LD (VDULNE),HL
40CF          RET
40D0 SETSTR:  LD A,£1A     ;Adds an EOF to the
40D2          LD (DE),A    ;  file to be saved
40D3          LD HL,£A000          ;Start address for file
40D6          EX DE,HL
40D7          XOR A
40D8          SBC HL,DE
40DA          INC HL       ;Length of file to be saved
40DB          RET
40DC CNVDIG:  XOR A        ;Finds the decimal value
40DD          LD E,A       ;  of the remainder of HL divided
40DE          LD D,A       ;  by ten
40DF          LD B,£03
40E1 FSTBTS:  RL L
40E3          RL H
40E5          RLA
40E6          DJNZ FSTBTS
40E8          LD B,£0D
40EA RESTRP:  AND A
40EB          RL L
40ED          RL H
40EF          RLA
40F0          CP £0A
40F2          PUSH AF
40F3          CCF
40F4          RL E
40F6          RL D
40F8          POP AF
40F9          JR C,NEXBIT
40FB          SUB £0A
40FD NEXBIT:  DJNZ RESTRP
40FF          RET
4100 CNVDEC:  LD BC,NUMSTR         ;Location of ASCII digit
4103 DECDIG:  PUSH BC
4104          CALL CNVDIG
4107          POP BC
4108          ADD A,£30    ;Convert to ASCII form
410A          LD (BC),A
410B          INC BC
410C          EX DE,HL
410D          LD A,L       ;Have we finished the
410E          OR H         ;  decimal conversion?
410F          JR NZ,DECDIG
4111          DEC BC       ;Store the whole number at WRTEND
4112          PUSH BC
4113          PUSH BC
4114          POP HL
4115          LD BC,NUMSTR
4118          XOR A
4119          SBC HL,BC    ;How many digits?
411B          LD B,L
411C          INC B
411D          POP HL       ;Top of work space
411E          LD DE,WRTEND         ;Final location of number
4121          PUSH BC
4122 MVESTR:  LD A,(HL)
4123          LD (DE),A
4124          INC DE
4125          DEC HL
4126          DJNZ MVESTR
4128          LD A,£FF     ;The disc command must end
412A          LD (DE),A    ;  with hex FF
412B          LD DE,£9FFF
412E          LD A,£1A
4130          LD (DE),A
4131          LD HL,NUMSTR
4134          POP BC
4135          LD A,£05
4137          SUB B
4138          PUSH AF
4139 LDLNTH:  DEC DE
413A          LD A,(HL)
413B          LD (DE),A
413C          INC HL
413D          DJNZ LDLNTH
413F          POP AF
4140          RET Z
4141          LD B,A
4142          LD A,£20
4144 FILLIN:  DEC DE
4145          LD (DE),A
4146          DJNZ FILLIN
4148          RET
```

```
10 REM EXAMPLE 3b
20 REM READing the file created in example 3a
30 CODE

4045          JR START
4047 READ:    DB £FF,£22,"TSTFILE.DMY",£22,",40960,"      ;First byte is dummy
405C READND:  DB "00000",£FF
4062 SIZE:    DB £FF,£22,"LENFILE.DMY",£22,",40954,6",£FF          ;Note READ command word is omitt
ed
4079 VDULNE:  DW £5C01
407B VDUWTH:  DB £28
407C START:   LD DE,SIZE
407F          CALL JPOUT ;Calls relevant routine
4082          LD DE,£9FFA         ;Location of file length
4085          LD HL,READND
4088 LDSIZE:  LD A,(DE)  ;Transfers file length to command line
4089          CP £1A
408B          JR Z,NWREAD
408D          INC DE
408E          CP £20      ;Skips leading blanks
4090          JR Z,LDSIZE
4092          LD (HL),A
4093          INC HL
4094          JR LDSIZE
4096 NWREAD:  LD (HL),£FF         ;Sets EOL marker
4098          LD DE,READ
409B          CALL JPOUT ;Call disc routine
409E          LD DE,£A000
40A1 NXLINE:  CALL SETLNE
40A4 PRINT:   DEC C       ;Displays file just read
40A5          JR Z,NXLINE
40A7          LD A,(DE)
40AB          CP £1A      ;EOF marker
40AA          RET Z
40AB          INC DE
40AC          CALL £00BC
40AF          JR PRINT
40B1 SETLNE:  LD HL,(VDULNE)
40B4          LD A,L
40B5          OUT (£02),A
40B7          LD A,H
40B8          OUT (£02),A
40BA          LD A,(VDUWTH)
40BD          LD C,A
40BE          LD B,£00
40C0          ADD HL,BC
40C1          LD (VDULNE),HL
40C4          RET
40C5 JPOUT:   LD A,(£FAD2)
40C8          PUSH AF     ;Save present page configuration
40C9          AND £0F     ;Preserve RAM setting
40CB          OR £30      ;Set to ROM page 30
40CD          LD (£FAD2),A
40D0          OUT (£00),A
40D2          CALL £2D63 ;READ disc control routine
40D5          POP AF      ;Restore page setting
40D6          LD (£FAD2),A
40D9          OUT (£00),A
40DB          RET
```

Helpline

"CAVES OF THE ORB" HELPLINE

Here's a hint or two for the adventure.
Having trouble with the rope bridge. Don't go across it. You can get from the east side to the west side by going N,N,W,S,S through the slab room, junction, and thin crack. For those of you who have surpassed this obstacle and who have got to the plant room what you do is GET THE BOTTLE from the house, go to the tap TURN ON THE TAP, FILL THE BOTTLE WITH WATER and TURN OFF TAP (to be tidy). Then you WATER THE PLANT in the plant room. All this info is easily deducable from the description of the plant (EXAMINE PLANT or X PLANT for short) and the help you can get.
Got to the Dragon yet? Tried to KILL DRAGON WITH SWORD and failed. EXAMINE SWORD should show why. Perhaps we will work this one out one day! Met the Dwarf yet? The Dwarf is into buying and selling.
Thats all for now, more mext time. Yours in despair in the Caves of the Orb.
THE GOD OF UNAMBIGUITY

Going Forth Keith Jones

Before launching into this month's routines I'd like to thank everyone who sent subroutines to me over the last month and for the warm reception the article generated, thanks, and keep those routines coming.

One point to note please, if the routines you send are of any appreciable length then it helps me greatly if the code is sent on cassette. I don't have unlimited time to key in your routines as my full time job keeps me rather busy (I'm a lecturer in microelectronics at a local college).

This month's routines show the variation in interests amongst MTX users, but I'll let the routines speak for themselves. Where a routine needs some BASIC to show how it is incorporated into the system that too has been given.

We start with some routines and macros written using EDASM to transfer data between the Z80 and the Texas VDP chip .

Macros:

```
DEFMAC  ("VRAMWR")  ;writes data held in the A register to
        OUT (1),A   ;VRAM  after setting up a write address
        END.        ;using one of the routines given below

DEFMAC  ("VRAMRD")  ;reads data from VRAM into the A
        IN A,(1)    ;register after setting up with one of
        END.        ;the read address routines.
```

Neither macro alters the registers or stack. Each requires two bytes every time macro is used.

Trace:

This routine will print the current BASIC line number when the ' BRK' key is used to stop a BASIC program. The number is printed at the top right hand corner of VS1 (the list screen) between chevrons, when READY appears.

There are three methods of implementing this routine:

1) Load your (troublesome ?) BASIC program and then assemble this code into line zero. The routine must be at the start of memory as it stands.
Then enter the following POKES directly from BASIC.

```
POKE 64154,64  (128 for the MTX500)
POKE 64153,7
POKE 64152,195
```

This will point a USER interrupt at the code in line zero.
Then POKE 64862,31 will turn on the trace facility and POKE 64862,15 will turn off the routine.

2) For the adventurous TRACE could be merged with your program, again to line zero. To do this, assemble TRACE into line zero, with NO comments, then save it to tape. When needed, load TRACE first, then use PANEL to add hex FB to system variable "VAZERO". Then load your program. Now use PANEL again to add hex FB to the following system variables, "NBTOP", "BASTOP", and "BASTPO". Put the original value back into "VAZERO", i.e. subtract hex FB from it. Exit PANEL. You should now have the programs merged and it's a good idea to save them to tape at this point.
Again the pokes given above are necessary.

3) If you look back to issues 7 and 8 of MEMOPAD you will see some articles by Eric Roy on some "utilities". You can use the concepts contained in these articles to,
first move the code produced in TRACE to a high position in memory
and secondly to turn the TRACE function on and off from one of the function keys .

That's all for this month, see you again next month with routines to copy VRAM pattern tables to normal ram for use in sprites! and code to drive your printer directly from your assembly language program.

This month it's back to normal with me droning on about yet another fascinating aspect of FORTH. I want to look at how we can manipulate values on the stack and I'd recommend that you load the definition of .S that I gave in issue 10.

The first operator I want to look at is "DUP", pronounced "dupe" and short for duplicate. Any value which is at the top of the stack will have a copy of itself placed below it when this word is used. Try this to see what I mean

```
5 DUP . .
```

and you'll see that although we only placed one value on the stack there were two values to come off it. The most frequent use of DUP is before a conditional test, no doubt you've already seen

```
DUP IF
```

in a lot of words. The reason for this is that the IF will destroy the value and we may need it later on.
One obvious use for DUP is in a word to produce the square of a number and this is easily coded as

```
: ^2
DUP * ;
```

Another form of DUP is -DUP which will only duplicate the top value so long as it's not zero.
The next word which we'll examine is SWAP. This word exchanges the position of the top two values on the stack.
Yet another word is OVER. This will copy the second value to the top of the stack. So if you do the following then you'll see the difference between SWAP and OVER

```
5 4 SWAP . .
5 4 OVER . . .
```

DROP will discard the top item off the stack.
ROT rotates the top three values, for instance if we typed

```
1 2 3
```

then the top value would be 3 and the bottom would be 1. Now typing ROT would make 1 the top value and 2 the bottom.

Contained in last month's article was the word PICK. As I'm sure you remember this word copied the nth value down from the top of the stack onto the top. I now present a word called ROLL which does the same as PICK but it removes the value which it copied. For example ;

STACK BEFORE	User types 4 ROLL	STACK AFTER
1		4
2		1
3		2
4		3
5		5

Here's the word ;

```
: ROLL
2 * SP@ +
DUP SO @ <
 IF DUP @ SWAP SP@ 2+ SWAP
  DO 1 2 - @ I ! -2 +LOOP DROP
 ELSE CR ."Below stack depth"
DROP QUIT THEN ;
```

I've received some correspondence from Dr. B. Houghton who rightly points out that the extra screens given in the article in issue 1, Volume 2, can not be saved to tape . In order to make amends I set my tired brain into operation and came up with a word which will set up all the necessary addresses so that all you need do is say how many screens you want, FORTH will then sort everything out so that you can save and load these screens as usual. The word is ;

```
HEX

: EXTRAS   ( n --- )

DUP 9 < IF
400 * D800 +
DUP
5990 !
D400 -
DUP
5ED9 ! 5EB9 !
 ELSE CR
 ."Not enough room "
DROP QUIT
ENDIF ;
```

If you've got VERIFY in (as given in issue 11) then the necessary steps to ensure that this routine will verify all the screens are as follows ;

```
HEX

n 400 * 400 - 'VERIFY 5 + !
```

Where n is the number of screens.

That's all for this month. Next time we'll look at some words sent to me and also yet another fascinating aspect of FORTH.★

Pascal Tip

Member R.A. Brooks has reminded us that the function keys on the MTX are programmed to insert pre-defined functions into a program when using the PASCAL rom. They are as follows:-

Function Key		Function Key [Shifted]	
1	_	1	_
2	PROGRAM	2	AND
3	DIV	3	MOD
4	CONST	4	VAR
5	PROCEDURE	5	OF
6	FUNCTION	6	TO
7	NOT	7	DOWNTO
8	OR	8	UNTIL

The underline symbol _ is used -so I believe- to pad out certain file names used in the $F command.

We would be interested to hear from any member who has discovered other method of using shortened key sequences.

Incidently,any member who is experiencing trouble with their PASCAL ROM when using the $F command to chain tape files, can return their board to HI-SOFT who will replace the Roms. This only applies to the earlier rom boards, as the bug was corrected in the latter half of last year.

Picking Pears S Aizlewood

Picking Pears is a program for either the MEMOTECH MTX 500 or 512 and is a test of memory. If you are familiar with the card game 'Pelmanism' then playing this version will be straight forward. Instead of having playing cards turned face down on a table, the computer displays an 8x8 grid with fruit randomly hidden within it. By selecting squares using grid co-ordinates the hidden fruit will be revealed for 10 seconds allowing you time to memorise their positions. When a pair is found they remain on the screen and play continues until all 32 pairs are found during which time running totals of choices made and pairs found are updated.

The program demonstrates a couple of the MEMOTECH'S SPECIAL FEATURES - NODDYfor display of instructions and multicolour user defined graphics.

Finally, the program should be saved to tape with GOTO 780 for auto-running when loaded .

PROGRAM NOTES

5	RUN 'NODDY' instructions.
10	Set up Hi-Res screen.
20-55	Define 8 fruits and assign ink colours.
70	Set up array for holding fruit
80-90	Intro screen.
120	Place fruit in A$.
145-180	Randomise positions of fruit in A$ with 100 passes.
200-335	Draw 8x8 grid et c.
400	Check for fruit already paired
475	Check for input error.
480	Check for fruit already paired
490-500	Print fruit in chosen squares.
510	Counter for number of attempts
530	Check for a pair.
540-560	Display fruit for 10 seconds then blank out squares.
580	Counter for number of pairs found.
600-610	Change character code of pair.
620	Check if all pairs have been found.
630-660	End of game routine.
700-770	Subroutine for input of grid positions.

MAIN VARIABLES

B	Array of 4 random numbers for shuffling A$.
A$	Holds graphic characters.
I,J	Loop counters.
X,Y	Loop counters.
L	Loop counter.
P	Pairs counter.
Q	Tries counter.
L,C	Cursor position.
X1,Y1	Position of square 1.
X2,Y2	Position of square 2.
K$,K	Value of key pressed during input routine.

NODDY PAGES FOR 'PICKING PEARS'

PROG

```
* D  INSTRUCTIONS
* E
* D  PAGE 2
* E
* R
```

PICKING PEARS is based on the popular card game 'PELMANISM' and is a test of memory. The Computer displays an 8 x 8 grid with 8 different types of fruit r andomly hidden within it. The fruits are BANANA, APPLE, PEAR, CHERRY, PLUM, LEMON, ORANGE and STRAWBERRY

Your task is to find the pairs by selecting two squares within the grid. The contents of the selected squares will then be displayed for 10 seconds. Memorise the positions of the fruit and continue making selections until you have found all 32 pairs. When you find pairs, they remain displayed and the game status is shown on the top right of the screen.

PRESS 'RET' FOR MORE..........

PAGE 2

~~~~~~~~ MAKING YOUR SELECTIONS ~~~~~~~~

Each square of the grid has it's own unique X/Y co-ordinate, therefore you have to input 2 numbers in the range 1 to 8 to select the desired square.

For example, to select the square at 5 across and 6 down:

Key '5' then 'RET'
Key '6' then 'RET'

The screen prompt will then ask you for the co-ordinates of your second choice. Follow the same procedure as described above. The contents of the 2 squares will then be revealed. ★

PRESS 'RET' TO PLAY
~~~~~~~~

```
0 REM ~~~~~ PICKING PEARS ~~~~~
1 REM ~~~~~ S. R. AIZLEWOOD ~~~~~
2 REM ~~~~~ FEB 1985 ~~~~~
3 REM ~~~~~ FOR THE MEMOTECH MTX ~~~~~
5 PLOD "PROG"
10 VS 4: COLOUR 4,1: PAPER 1: CLS
20 GENPAT 1,147,3,4,8,96,246,255,111,6:
 GENPAT 2,147,193,193,193,97,97,97,97,9
7
25 GENPAT 1,148,1,3,7,14,28,120,240,192
: GENPAT 2,148,193,49,177,177,177,177,1
77,49
30 GENPAT 1,149,20,62,127,127,127,62,28
,0: GENPAT 2,149,144,33,33,33,193,193,4
9,144
35 GENPAT 1,150,16,56,56,56,92,108,124,
56: GENPAT 2,150,129,193,193,193,193,19
3,193,193
40 GENPAT 1,151,0,56,124,254,254,124,56
,0: GENPAT 2,151,0,81,209,209,209,209,8
1,0
45 GENPAT 1,152,16,16,40,124,124,124,56
,16: GENPAT 2,152,193,193,97,97,97,97,9
7,97
50 GENPAT 1,153,28,62,127,255,127,62,28
,0: GENPAT 2,153,49,161,161,161,161,161
,48,0
55 GENPAT 1,154,24,60,118,122,126,60,24
,0: GENPAT 2,154,145,145,145,145,145,14
5,145,0
70 DIM B(4),A$(8,8)
80 CSR 10,0: INK 15: PAPER 6: PRINT " P
ICKING PEARS "
90 CSR 10,10: INK 11: PAPER 1: PRINT "S
EITING UP";
120 FOR I=1 TO 8: LET A$(I)=CHR$(147)+C
HR$(148)+CHR$(149)+CHR$(150)+CHR$(151)+
CHR$(152)+CHR$(153)+CHR$(154)
140 NEXT
145 FOR I=1 TO 100: IF I=10*INT(I/10) T
HEN  PRINT ".";: SOUND 0,200-I,15
150 FOR J=1 TO 4: LET B(J)=INT(RND*8+1)
160 NEXT J
165 LET X$=A$(B(1),B(2))
170 LET A$(B(1),B(2))=A$(B(3),B(4))
175 LET A$(B(3),B(4))=X$
180 NEXT I: SOUND 0,0,0
185 COLOUR 0,1: CLS
200 COLOUR 3,4: FOR X=52 TO 180 STEP 16
: LINE 20,X,148,X: NEXT
205 FOR Y=20 TO 152 STEP 16: LINE Y,180
,Y,52: NEXT
215 CSR 3,18: INK 10: PRINT "1 2 3 4 5
6 7 8"
220 CSR 3,0: PRINT "1 2 3 4 5 6 7 8"
230 FOR L=2 TO 16 STEP 2: CSR 0,L
240 PRINT 9-L/2: CSR 19,L: PRINT 9-L/2
250 NEXT
260 LET P=0: LET Q=0
280 CSR 23,3: INK 15: PAPER 6: PRINT "T
RYS ="
290 CSR 23,5: PRINT "PAIRS="
300 CSR 29,10: PAPER 1: PRINT "X Y"
310 CSR 21,11: PAPER 6: PRINT "BLOCK 1"
320 CSR 21,13: PRINT "BLOCK 2"
330 CSR 28,13: PAPER 1: PRINT "     "
335 CSR 28,11: PRINT "     "
340 LET L=11: LET C=28
350 GOSUB 700
360 LET X1=K: LET C=30
380 GOSUB 700
390 LET Y1=K
400 IF ASC(A$(X1,Y1))>154 THEN  GOTO 33
0
410 LET L=13: LET C=28
430 GOSUB 700
440 LET X2=K: LET C=30
460 GOSUB 700
470 LET Y2=K
475 IF X1=X2 AND Y1=Y2 THEN  GOTO 330
480 IF ASC(A$(X2,Y2))>154 THEN  GOTO 33
0
490 CSP (2*X1)+1,18-2*Y1: PRINT A$(X1,Y
1)
500 CSP (2*X2)+1,18-2*Y2: PRINT A$(X2,Y
2)
510 LET Q=Q+1
520 CSR 29,3: PRINT Q
530 IF A$(X1,Y1)=A$(X2,Y2) THEN  GOTO 5
80
540 PAUSE 10000
550 CSR (2*X1)+1,18-2*Y1: COLOUR 1,1: P
RINT " "
560 CSR (2*X2)+1,18-2*Y2: PRINT " "
570 GOTO 330
580 LET P=P+1: CSR 29,5: PRINT P
590 PRINT CHR$(7)
600 LET A$(X1,Y1)=CHR$(ASC(A$(X1,Y1))+8
)
610 LET A$(X2,Y2)=CHR$(ASC(A$(X2,Y2))+8
)
620 IF P<>32 THEN  GOTO 330
630 CSR 23,15: ATTR 0,1: PRINT "WELL DO
NE": ATTR 0,0
640 CSR 2,22: PRINT "PRESS 'SPACE' FOR
ANOTHER GO"
650 LET Z=ASC(INKEY$): IF Z=-1 THEN  GO
TO 650 ELSE  IF Z=32 THEN  RUN
660 STOP
700 CSR C,L: INK 15: INPUT K$
710 SOUND 0,500,15
715 IF LEN (K$)<>1 THEN  GOTO 700
720 IF K$<="8" AND K$>="1" THEN  GOTO 7
50
740 GOTO 700
750 LET K=VAL(K$)
760 CSR C,L: PRINT K: SOUND 0,0,0
770 RETURN
780 SAVE "PEARS"
790 GOTO 5
```

View Letters TO THE EDITOR Point

J. M. Cartney has made the following comments on "Take A Note"
Regarding "Take A Note" (issue 10) by Derek Brown

Each note of the Chromatic scale 12 in all are equally separated. To move any note up or down by 1 semitone you multiply it by 1.0594630945 or divide by this number as it is the $\sqrt[12]{2}$
i.e. 12 times this difference would be 2.

1.059463 is accurate enough for calculations.

Using this value with my Sharp scientific calculator I calculate the chromatic scale from A 55 HZ to A 28160 HZ 10 Octaves although 28160 HZ beyond the human hearing range the actual value was 28160.014 which is a lot more accurate than multiplying a known approximate chromatic scale by 2 each time.

Alberto Bernabe would like to contact other members and has sent in the following.

I became a member of your club about five months ago, and I enjoy the monthly magazine you send me.

I would like to know if there is another member in Spain, as I would be interested in contacting anybody to exchange ideas .

I would appreciate any information that you could give me about this.

Another letter received from Jerry Dring

I enclose a high score for the adventure game 'SNOWBALL'. After many sleepless nights strolling around the S-9 giving things to robots and giving drinks to cold-crew-mates my brother (Kevin) completed the game with a score of 1000.

He wandered around the 10 passenger discs, travelled on the shuttle, visited a hotel with holograms everywhere .

No deal really he just saved a 5 mile long colony starship with 2 million passengers on board for a firery death so could you put his name on the HI-score table, (he likes to be called K-Mander Kev !).

Thanks.

Letter from D.S. Hodgson, 9 Hillbeck Way, Greenford, Middlesex. UB6 8LT.
Tel. 01-578-9001 (WORKS)
18th November 1985

Dear Sirs,

With a business going through a 44% expansion, I have little time for my own pursuits, but a good product is worth a bit of effort, so here goes.

We all seem agreed that the Memotech is the best microcomputer on the market, and that most others are, by comparison, little more than highly sophisticated toys. I cannot, therefore, understand our Amersham member's unwillingness to promote the product among his friends. A recommendation is worth a dozen adverts - and I am speaking from experience. If Memotech wish to make it worth our while, even better - and they are certainly not the first to use such tactics. Readers Digest, window companies and many more know only too well the power of recommendation, and give incentives to customers to introduce new business.

It was nice to read more constructive comments in the latest issue, although much of it was over my head. I cannot understand why Derek Bergin feels that the current design (keyboard?) would not stand up to serious office use. My expanded MTX512 is loaded with a 520-page Noddy file which is on from 8.30a.m. to 5.45 p.m. Mon-Fri., and is constantly being called up. It is also frequently in use as a word-processor, and I find no problems. I have also had the DMX80 printer in fairly constant use for the past 18 months, and it stands up to all I can give it!

So where do we go from here?

It is critical that both Memotech and Genpat identify their market. The lack of software sales must prove that most owners are not games-playing fanatics. If that is the market you want, however, then Memotech must go into the High Street - there is no other way. It is obvious from the Letters column that many owners are extremely advanced. This group are unlikely to buy software, but will buy sophisticated hardware on its specification alone.

I suspect that a large number of users (the majority?) are similar to myself. I have little time or interest for games - and this seems to have rubbed off onto my children. I am capable of writing simple programs - Product Turnover Analysis, Customer Turnover Analysis, etc. I have a 6-year old twin floppy 8k Logabax that does all the invoicing, accounts and stock, but there are various reasons why it is easier to use the MTX for functions it does not cover. I will need to update the business system in the near future . I would like the system to be Memotech-based, but I doubt if Memotech have tied up with a data systems company to offer a fully-housed package. There is no way that I will buy the hardware and software separate. The staff would need personal instruction on a new systemnm for a start, and having cables dangling here, there and everywhere is an absolute no go, no matter what the saving.

I would like to buy a disk drive, but the information printed so far in Genpat has been no help whatsoever. The sort of information I, and possibly others like me, require is:
What sort of criteria (apart from money) does one use to decide whether one needs the 250K SDX 500K, an FDX single or twin system?
What happens to my 500+ page Noddy file? Do I simply load into memory from tape and dump it whole onto disk, or do I need to feed it in in smaller pieces? In other words, describe how, in simple and practical steps, how I change from tape to disc, not bother me with how discs are sectored (I already know that) or delve into the various commands at my disposal (I will soon master them when I buy the system).

Similar comments apply when reviewing serious software. Half a page is fine for games reviews, but useless to me if you are reviewing Management Data base Systems. I need to know, in simple terms, what it does and how to operate it. If that takes half the magazine, then so be it.

So, Memotech and Genpat, you have four basic types of customers. Who are you aiming at? You almost certainly haven't the resouces to enter the games market properly. The advanced user will look after himself as far as programmes are concerned, but will constantly demand the most advanced hardware available. Have you the resources to develop the super-duper 32-bit, no-one-else-can-touch-it computer they are looking forward to? Or will you put together a fully-housed package, including desk with drawers, for the data systems companies? Or will you settle for the likes of me?

Or perhaps you are tired of everybody telling you how to run your business? Maybe you are already achieving the sales and profits you are looking for! If that is the case, congratulations! - but please tell us, so that hours of letter writing by other well-meaning members can be saved in future.

Both of you must decide where you are going - and when you have done so, go for that market with full dedication. Whichever you opt for, I wish you well, as I now have an expanded 512, an RS128 and am considering a further 512 . The kids keep complaining of lack of access!

Yours sincerely,

Denekamper Str.8
4460 Nordhorn
West Germany
15 Nov. 1985

The Editor,
Memopad

Dear Sir,

I have just received my first copy of GENPAT, and I must confess to a feeling of disappointment when I opened it and was faced with a total of seven out of 28 sides devoted to computer games. I thought when I joined the Memotech User Group that I would be associating myself with a better class of computer-owners than moronic joystick jogglers! I had my share of zapping (and being zapped by) aliens forty-odd years ago and have had enough real-life adventures to be able to dispense with the synthetic variety.

My disappointment diminished when I turned the pages further and found that the other articles were more to my taste and that other correspondents shared my opinion of computer-games and those who play them.

I was particularly interested to see that you propose to publish a book on machine code for the MTX since I intended to write to ask you to recommend such a book. I have a number of books on Assembler and Z80 machine code and I am quite well up on how to shuffle numbers, Hex, Binary and BCD, from one register to another and rotate them right and left, but how to turn this knowledge into a program still eludes me. I look forward to seeing whether your effort can enlighten me.

As a counter blast to your correspondent, Paul Schofield of Switzerland, may I tell of my experience of ordering and receiving an SDX system from Memotech? I rang Memotech to discuss ordering an FDX system in July. Having explained my needs, I was advised against the FDX and recommended the SDX, despite its lower price (and smaller profit for Memotech). I ordered the SDX at the end of July and was rung up by the Sales Department who warned me that the firm was about to close down for its summer holidays. Nevertheless, I received my machine three weeks later, delivered by Securicor, on the day specified. As to getting advice on the telephone from Memotech, I have always been put through to the appropriate department and been given helpful and accurate advice which has solved my problem. It is quite clear that a business concern does not want its telephone lines cluttered all day with people who have problems which are not due to faults in the design or construction of its products, and will try to pass them on to an organisation, such as Genpat, whose declared object is to help Memotech owners. If only all computer manufacturers were as obliging as Memotech, the computer magazines' correspondence columns would be half empty.

I look forward to receiving future copies of Memopad in the hope of finding useful an instructive articles (and far less trivia about games).

Yours faithfully,

Alex Smith

John Hodgson has sent the following observations on the RST Articles

I would like to make a few comments on the very good article by John Hudson on the use of the RST28 command. On the blunders page John has put RST28 #8E as 8E 1289 MOVE(DE) to OP11. The address should be 12B9 and was correct the first time. (DE) should be the number pointed to by the contents of reg DE, i.e. RST28#8E is move the 5 byte number pointed to by reg. DE to the address OP11 . If you look at address 12AC and 12B3 there is a LDBC,5 followed by a LD1R. This applies to routine 12A7 MOVE (DE) To ACC1 and may also apply to some of the other ROM routines .

Here are some more ROM routines that may be of interest . I'll use <REG> to denote the "NUMBER POINTED TO BY REG ".

0953	SOUND BELL
0DD0	CONVERT NUMBER IN BC TO A DECIMAL <DE> ENDED WITH FF
0E28	ACC/10 **
0F75	(HL) = (HL) + 2's COMPLEMENT (A)
1163	IF <DE> = <HL> THEN (A) = 0
	IF <DE> GT <HL> THEN (A) = 1
	IF <DE> LT <HL> THEN (A) = -1
119D	Test IF ACC IS AN INTEGER IN THE RANGE 0 TO 255
11B9	CONVERT A REAL NUMBER IN ACC1 TO A NUMBER IN REG A
11EC	2's COMPLEMENT (BC)
11F6	TESTS IF REAL <HL> US BETWEEN 0 AND 65535
121D	ASC II <DE> TO (DE)
200A	INTEGER PART OF ACC
204B	GENERATE A NEW RANDOM NUMBER SEED
208F	TEST THE SIGN OF ACC
	IF 0 THEN ACC = 0
	IF + THEN ACC = 1
	IF - THEN ACC = -1
209A	AND ACC WITH OP1 **
20A6	OR ACC WITH OP1 **
21ED	ACC/10 (NORMALISED) **
220A	ACC*10 (NORMALISED) **
22DF	ASCII <DE> TO A HEX NUMBER IN A, 'HL,HL'

I don't know if John intends to include it in his article but the RST28 command is also used to print out the BASIC error messages. Let me know if you want more information on this .

One of the problems with the MTX is that unlike some other computers you cannot set aside an area of RAM that is protected from the operating system. The following routine will reset the start address of BASIC and will give you an area of RAM that is safe to use without the fear of it being overwritten.

```
LD A, (#FA7A)
OR A
LD HL, #5000 ; NEW START ADDRESS FOR  BASIC
CA LL #22B ; INIT TO NEW START  ADDRESS
RET
```

Now load your program and you will find that it now starts at address #5000. The only problem is that you cannot load programs that have been saved when the MTX has been in this state.

Mr.R. Siddall has a Memotech compatable Tandy 'Radio Shack' colour graphics printer with interface and dust cover for sale at a price of 100.00. Details can be obtained from Mr. Siddall at:-

23, Langtree ave, Old Whittington, Chesterfield, Derbyshire, S41 9HW

Here is a letter from Mr. Crighton of Gravesend.

Congratulations on the first year of Genpat. I hope there are many more to come. I was very pleased to see Assembly line in the latest edition. How about a similar article for Basic?

I think that Paul Wood's idea for a survey would help the club. If you do undertake a survey it would be interesting if you published the results.

ED's comment:We think Mr. Crighton's idea of a "Basic line" is an excellent suggestion. You will find the first article in the series on Basic in this issue. Each month we will print useful Basic listings, so if you have any useful programs or subroutines, please send them to us.

I have recently made a quite interesting discovery concerning the PASCAL (ROM version) which I feel may be useful to others.

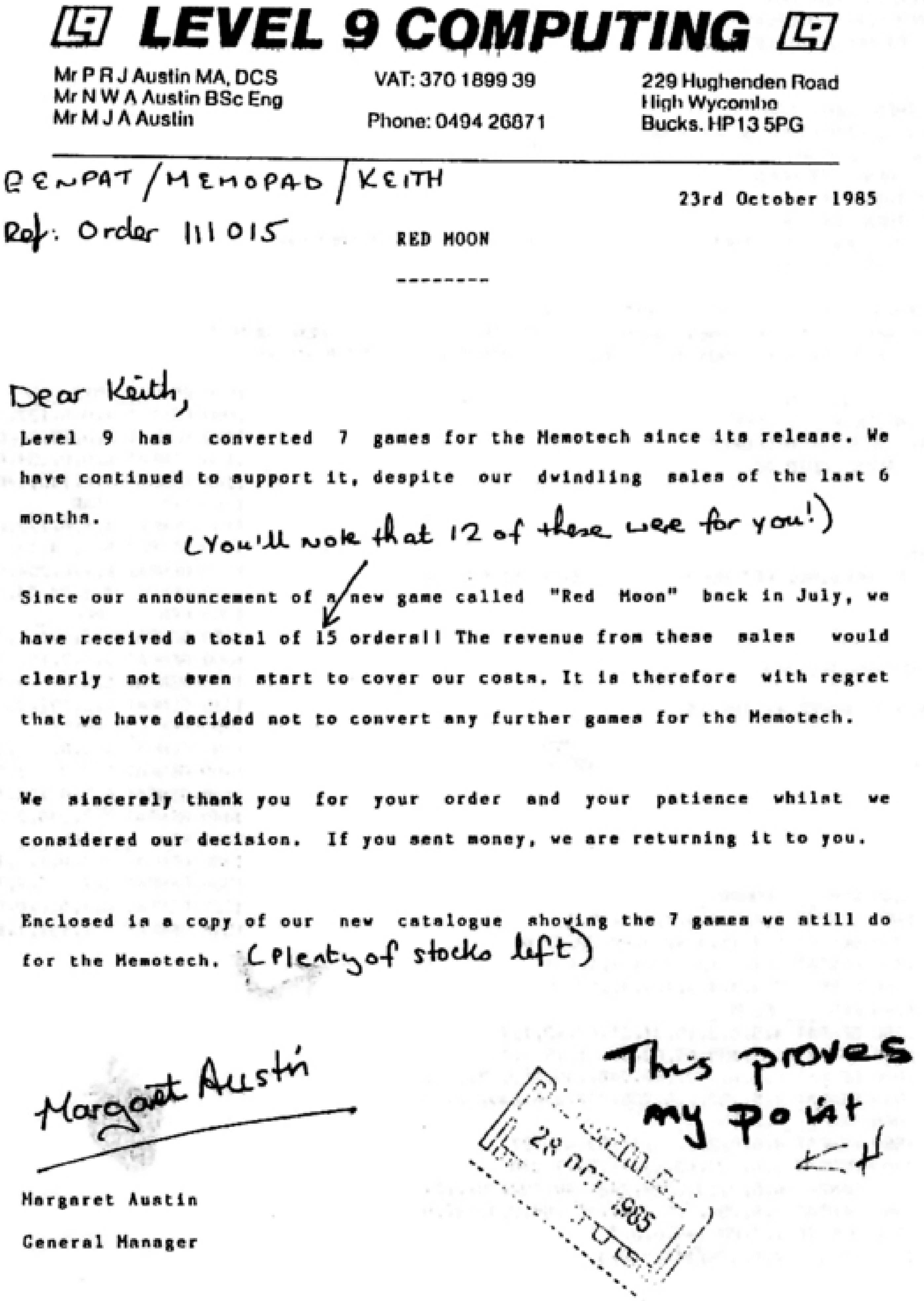
I know that we have all experienced finger produced hang-ups. My discovery is that they are recoverable. In the event of a hang-up the procedure is simply this:

1) Press the two RESET keys.
2) Enter "ROM 2"
3) Select B from the menu. Do not press <RETURN>
4) Press CTRL and X keys (simultaneously)
5) Program is recovered

I don't know if the Procedure works for an unexplained hang-up, like a mains glitch or what have you, because I haven't exerienced one since I discovered this.

The following comments about FORTH have been sent in by Terry Trotter:
After reviewing FORTH I ran the PCW benchmarks for FORTH, the average time is 19 seconds - as fast as all other versions except the one on the IBM PC with 8088 & 8087 co-processor, that version from FORTH INC.

Acornsoft	FORTH	average	21	BBC MICRO
JWB	"	"	24	"
ZX81	"	"	35	
Knights	"	"	57	(MZ80A)
JWB	"	"	38	Epson HX-20



LEVEL 9 COMPUTING

Mr P R J Austin MA, DCS
Mr N W A Austin BSc Eng
Mr M J A Austin

VAT: 370 1899 39

Phone: 0494 26871

229 Hughenden Road
High Wycombe
Bucks. HP13 5PG

GENPAT / MEMOPAD / KEITH
Ref: Order 111015

23rd October 1985

RED MOON

Dear Keith,

Level 9 has converted 7 games for the Memotech since its release. We have continued to support it, despite our dwindling sales of the last 6 months.

(You'll note that 12 of these were for you!)

Since our announcement of a new game called "Red Moon" back in July, we have received a total of 15 orders!! The revenue from these sales would clearly not even start to cover our costs. It is therefore with regret that we have decided not to convert any further games for the Memotech.

We sincerely thank you for your order and your patience whilst we considered our decision. If you sent money, we are returning it to you.

Enclosed is a copy of our new catalogue showing the 7 games we still do for the Memotech. (Plenty of stocks left)

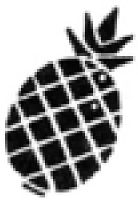
Margaret Austin

Margaret Austin
General Manager

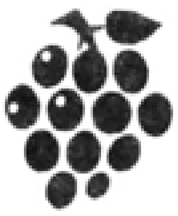
This proves my point
KH

Fruit Machine T.J. Seldon

```
10 REM ******************************
20 REM *                            *
30 REM *       FRUIT MACHINE        *
40 REM *                            *
50 REM *   WRITTEN BY : T.J.SELDON  *
60 REM *                            *
70 REM *      DATE    : 25/11/85    *
80 REM *                            *
90 REM ******************************
100 GOSUB 1000: REM SET UP SPRITES
110 GOSUB 2400: REM SET UP SCREEN
120 GOSUB 2800: REM SET UP REELS
130 GOSUB 4200: REM INITIALISE GAME
140 FOR PUTIN=10 TO 300 STEP 10
145 SOUND 1,1600,7: PAUSE 30: SOUND 1,0,0: PAUSE 30: SOUND 1,600,9: PAUSE 30: SOUND 1,0,0
150 CSR 3,2: INK 11: PRINT PUTIN;"  "
160 GOSUB 4000: REM SPIN REELS
165 GOSUB 5800: REM CLEAR DISPLAY
170 GOSUB 4800: REM CHECK WIN
180 LET EXIT=0
190 IF WIN=0 THEN  GOTO 300
200 IF WIN=300 THEN  GOSUB 10000: GOTO 290
210 GOSUB 5100: REM DISPLAY WIN OPTIONS
220 FOR X=0 TO 16 STEP 2: SOUND 3,3,16-X: LET KEY$=INKEY$: SOUND 2,900-WIN,X: NEXT : IF KEY$=""
THEN  GOTO 220
230 IF KEY$="\" THEN  POKE 64145,128: POKE 64862,143: STOP
240 IF KEY$<>"G" AND KEY$<>"g" THEN  GOTO 250
245 IF RND>.45 THEN  SOUND 1,960,-1,-50,0,19,1: LET WIN=GWIN ELSE  FOR X=15 TO 0 STEP -1: SOUND
3,2,X: PAUSE 40: NEXT : LET WIN=LOSE: LET EXIT=1
250 IF KEY$="C" OR KEY$="c" THEN  FOR X=0 TO 16: SOUND 2,WIN+800,X: PAUSE 20: NEXT : LET EXIT=1
260 IF KEY$="N" OR KEY$="n" THEN  GOSUB 6400: IF WIN=300 THEN  GOSUB 10000
270 IF KEY$="W" OR KEY$="w" THEN  IF INT(WIN/35)>0 THEN  GOSUB 8300
280 IF EXIT=0 THEN  GOTO 200
290 GOSUB 5600: REM DISPLAY OUT
300 GOSUB 5800: REM CLEAR DISPLAY
310 LET HOLD1=0: LET HOLD2=0: LET HOLD3=0
320 LET HOLD=HOLD-1
325 LET FLASH=4
330 IF PUTIN=300 THEN  GOTO 500
335 IF HOLD<1 THEN  GOSUB 6200: GOSUB 6000
340 IF HOLD>=1 THEN  GOTO 500
350 IF INKEY$="1" THEN  LET HOLD1=1
360 IF INKEY$="2" THEN  LET HOLD2=1
370 IF INKEY$="3" THEN  LET HOLD3=1
380 IF INKEY$="C" OR INKEY$="c" THEN  LET HOLD1=0: LET HOLD2=0: LET HOLD3=0
390 GOSUB 6200: REM SHOW HOLDS
500 REM
580 IF PUTIN=290 THEN  CSR 3,22: INK 12: PRINT "LAST GO"
590 IF INKEY$<>" " AND PUTIN<>500 THEN  GOTO 340 ELSE  IF INKEY$<>" " THEN  GOTO 590
600 IF HOLD<1 THEN  LET HOLD=INT(RND*6): LET FLASH=0: GOSUB 6200: GOSUB 10500
620 NEXT PUTIN
630 VS 5: PAPER 6: INK 1: CLS
640 GOSUB 7800: REM CHECK  HISCORE
650 GOSUB 8100: REM DISPLAY HISCORES
660 IF INKEY$<>" " THEN  GOTO 660
670 LET GOTOUT=0
680 VS 4
690 GOSUB 5800
700 CSR 26,2: PRINT "      "
710 LET HOLD1=0: LET HOLD2=0: LET HOLD3=0: LET HOLD=INT(RND*5)
720 PAPER 1
995 GOTO 140
1000 REM
1002 REM [[[ SET UP SPRITES ]]]
1005 REM
1010 VS 4: COLOUR 4,1: PAPER 1: INK 15
1015 CLS
1020 CTLSPR 0,0
1040 CTLSPR 2,9
1060 CTLSPR 6,3
```



```
1070 REM ___ MTX ___
1080 GENPAT 4,0,0,127,0,69,109,124,84,84
1100 GENPAT 5,0,68,68,68,68,68,0,127,0
1120 GENPAT 6,0,0,254,0,210,210,146,146,140
1140 GENPAT 7,0,140,146,146,146,146,0,254,0
1160 REM ___ BAR ___
1180 GENPAT 4,1,0,127,0,113,74,74,74,115
1200 GENPAT 5,1,74,74,74,74,122,0,127,0
1220 GENPAT 6,1,0,254,0,156,82,82,82,220
1240 GENPAT 7,1,84,84,82,82,82,0,254,0
1260 REM ___ BELL ___
1280 GENPAT 4,2,0,1,1,3,7,7,7,7
1300 GENPAT 5,2,7,15,31,30,15,7,3,3
1320 GENPAT 6,2,0,0,0,128,192,192,192,192
1340 GENPAT 7,2,192,224,176,112,224,192,0,0
1360 REM ___ APPLE ___
1380 GENPAT 4,3,0,0,0,6,15,31,31,31
1400 GENPAT 5,3,31,31,31,15,7,7,3,0
1420 GENPAT 6,3,0,128,128,224,240,248,248,248
1440 GENPAT 7,3,248,208,208,176,160,96,192,0
1460 REM ___ STAR ___
1480 GENPAT 4,7,0,1,1,1,3,3,31,15
1500 GENPAT 5,7,7,3,7,7,14,12,8,0
1520 GENPAT 6,7,0,0,0,0,128,128,240,224
1540 GENPAT 7,7,192,128,192,192,224,96,32,0
```



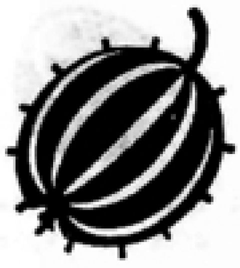
```
1660 REM ___ POUND ___
1680 GENPAT 4,4,0,3,4,4,4,4,15,4
1700 GENPAT 5,4,15,4,4,20,44,38,27,0
1720 GENPAT 6,4,0,132,76,4,4,4,4,4
1740 GENPAT 7,4,4,4,4,4,4,4,238,0
1760 REM ___ PLUM ___
1780 GENPAT 4,5,0,3,15,31,31,63,63,127
1800 GENPAT 5,5,255,63,63,31,31,15,3,0
1820 GENPAT 6,5,0,192,240,248,248,252,252,252
1840 GENPAT 7,5,252,236,236,232,216,240,224,0
1860 REM ___ LEMON ___
1880 GENPAT 4,6,0,3,7,31,31,63,63,127
1900 GENPAT 5,6,127,63,63,31,31,7,3,0
1920 GENPAT 6,6,0,192,224,248,248,252,252,254
1940 GENPAT 7,6,254,228,204,216,184,224,192,0
2020 SPRITE 1,7,88,144,0,0,7
2040 SPRITE 4,0,136,144,0,0,4
```

```
2060 SPRITE 7,2,184,144,0,0,9
2080 SPRITE 2,6,88,112,0,0,10
2100 SPRITE 5,6,136,112,0,0,10
2120 SPRITE 8,6,184,112,0,0,10
2140 SPRITE 3,3,88,80,0,0,2
2160 SPRITE 6,2,136,80,0,0,9
2180 SPRITE 9,5,184,80,0,0,13
2320 RETURN
2400 REM
2420 PEM [[[ SET UP SCREEN ]]]
2440 PEM
2450 LET FLASH=2
2460 CSR 4,0: PRINT "IN"
2480 CSR 28,0: PRINT "OUT"
2490 INK 5
2500 CSR 5,6: PRINT "MTX"
2520 CSR 27,6: PRINT "POUND"
2540 CSR 3,7: PRINT CHR$(27);"B1£3.00"
2560 CSR 27,7: PRINT "£1.00"
2580 CSR 5,9: PRINT "BAR ________________  PLUM    £2.00"
2600 CSR 27,10: PRINT "£0.70"
2620 CSR 4,12: PRINT "BELL"
2640 CSR 27,12: PRINT "LEMON"
2660 CSR 3,13: PRINT "£2.00"
2680 CSR 27,13: PRINT "£0.50"
2700 CSR 3,15: PRINT "APPLE"
2720 CSR 27,15: PRINT "STAR"
2740 CSR 3,16: PRINT "£1.50"
2760 CSR 27,16: PRINT "£0.30"
2770 GOSUB 10600
2780 RETURN
2800 REM
2820 REM [[[ SET UP REELS ]]]
2840 REM
2850 DIM WR2(8),WR3(8)
2860 DIM REEL(3,22),COL(8)
2880 FOR X=1 TO 3
2900 FOR Y=1 TO 22
2920 READ REEL(X,Y)
2940 NEXT Y
2960 NEXT X
2965 FOR X=1 TO 8: READ WR2(X),WR3(X): NEXT X
2980 DATA   4,7,6,3,5,4,1,5,6,4,5,0,6,5,6,3,7,4,2,5,4,7
3000 DATA   7,0,6,2,7,1,6,3,6,2,1,4,0,6,2,5,3,6,1,5,7,0
3020 DATA   7,2,6,5,1,4,6,1,7,2,3,5,0,6,7,2,4,3,6,1,7,2
3030 DATA    1,12,5,4,3,9,7,10,11,5,19,3,2,2,0,0
3040 LET R1=0: LET R2=0: LET R3=0
3060 LET HOLD1=0: LET HOLD2=0: LET HOLD3=0
3100 LET COL(1)=4: LET COL(2)=6: LET COL(3)=9: LET COL(4)=2
3120 LET COL(5)=12: LET COL(6)=13: LET COL(7)=10: LET COL(8)=7
3140 RETURN
3200 REM
3220 REM [[[ SET REEL POSITION ]]]
3240 REM
3260 REM SETS REELS ACCORDING TO VALUE
3270 REM OF R1,R2,R3          RANGE 0-19
3280 ADJSPR 0,1,REEL(1,R1+1)
3300 ADJSPR 1,1,COL(REEL(1,R1+1)+1)
3320 ADJSPR 0,2,REEL(1,R1+2)
3340 ADJSPR 1,2,COL(REEL(1,R1+2)+1)
3360 ADJSPR 0,3,REEL(1,R1+3)
3380 ADJSPR 1,3,COL(REEL(1,R1+3)+1)
3560 ADJSPR 0,4,REEL(2,R2+1)
3580 ADJSPR 1,4,COL(REEL(2,R2+1)+1)
3600 ADJSPR 0,5,REEL(2,R2+2)
3620 ADJSPR 1,5,COL(REEL(2,R2+2)+1)
3640 ADJSPR 0,6,REEL(2,R2+3)
```

```
3660 ADJSPR 1,6,COL(REEL(2,R2+3)+1)
3760 ADJSPR 0,7,REEL(3,R3+1)
3780 ADJSPR 1,7,COL(REEL(3,R3+1)+1)
3800 ADJSPR 0,8,REEL(3,R3+2)
3820 ADJSPR 1,8,COL(REEL(3,R3+2)+1)
3840 ADJSPR 0,9,REEL(3,R3+3)
3860 ADJSPR 1,9,COL(REEL(3,R3+3)+1)
3880 RETURN
4000 REM
4020 REM [[[ SPIN WHEEL ]]]
4040 REM
4045 IF HOLD1=1 AND HOLD2=1 AND HOLD3=1 THEN  RETURN
4046 SOUND 3,7,5
4048 SOUND 2,8000,0,2500,0,200,1
4050 FOR Q=1 TO 1(
4060 IF HOLD1=0 THEN  LET R1=INT(RND*20)
4080 IF HOLD2=0 THEN  LET R2=INT(RND*20)
4100 IF HOLD3=0 THEN  LET R3=INT(RND*20)
4120 GOSUB 3200: REM POSITION REELS
4140 NEXT Q
4145 FUR X=10 TU 0 STEP -2: SOUND 3,4,X: PAUSE 40: NEXT
4150 GOSUB 10500: REM STOP SOUNDS
4160 RETURN
4200 REM
4220 REM [[[ INITIALISE GAME ]]]
4240 REM
4250 SBUF 10
4260 DIM HI$(8,20),HI(8)
4280 POKE 64145,132: REM SET KEYPAD NUMBERS
4290 POKE 64862,13: REM DISABLE BREAK USE "\" WHEN ON A WIN
4300 FOR X=1 TO 8
4320 LET HI$(X)="MEMOTECH": LET HI(X)=300+(8-X)*20
4340 NEXT X
4380 LET GOTOUT=0
4400 LET HOLD=INT(RND*8)
4420 GENPAT 0,97,224,224,224,224,0,0,0,0
4422 GENPAT 0,98,28,28,28,28,0,0,0,0
4424 GENPAT 0,99,252,252,252,252,0,0,0,0
4426 GENPAT 0,100,0,0,0,0,224,224,224,224
4428 GENPAT 0,101,224,224,224,224,224,224,224,224
4430 GENPAT 0,102,28,28,28,28,224,224,224,224
4432 GENPAT 0,103,252,252,252,252,224,224,224,224
4434 GENPAT 0,104,0,0,0,0,28,28,28,28
4436 GENPAT 0,105,224,224,224,224,28,28,28,28
4438 GENPAT 0,106,28,28,28,28,28,28,28,28
4440 GENPAT 0,107,252,252,252,252,28,28,28,28
4442 GENPAT 0,108,0,0,0,0,252,252,252,252
4444 GENPAT 0,109,224,224,224,224,252,252,252,252
4446 GENPAT 0,110,28,28,28,28,252,252,252,252
4448 GENPAT 0,111,252,252,252,252,252,252,252,252
4450 RETURN
4500 REM
4520 REM [[[ CHECK STARS ]]]
4540 REM
4550 LET ST1=0: LET ST2=0: LET ST3=0
4560 IF REEL(1,R1+1)=7 THEN  LET ST1=1
4580 IF REEL(1,R1+2)=7 THEN  LET ST1=1
4600 IF REEL(1,R1+3)=7 THEN  LET ST1=1
4620 IF REEL(2,R2+1)=7 THEN  LET ST2=1
4640 IF REEL(2,R2+2)=7 THEN  LET ST2=1
4660 IF REEL(2,R2+3)=7 THEN  LET ST2=1
4680 IF REEL(3,R3+1)=7 THEN  LET ST3=1
4700 IF REEL(3,R3+2)=7 THEN  LET ST3=1
4720 IF REEL(3,R3+3)=7 THEN  LET ST3=1
4730 RETURN
```

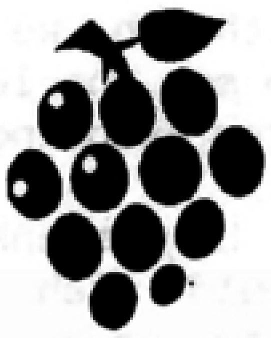
```
4800 REM
4820 REM [[[ CHECK WIN ]]]
4840 REM
4850 LET WIN=0
4860 GOSUB 4500: REM CHECK STARS
4865 IF ST1=0 AND ST1=ST2 AND REEL(1,R1+2)<>REEL(2,R2+2) THEN  RETURN
4880 IF ST1=1 AND ST2=1 AND ST3=1 THEN  LET WIN=30
4900 IF REEL(1,R1+2)=0 AND REEL(2,R2+2)=0 AND REEL(3,R3+2)=0 THEN  LET WIN=300
4920 IF REEL(1,R1+2)=1 AND REEL(2,R2+2)=1 AND REEL(3,R3+2)=1 THEN  LET WIN=200
4940 IF REEL(1,R1+2)=2 AND REEL(2,R2+2)=2 AND REEL(3,R3+2)=2 THEN  LET WIN=200
4960 IF PEEL(1,R1+2)=3 AND REEL(2,R2+2)=3 AND REEL(3,R3+2)=3 THEN  LET WIN=150
4980 IF REEL(1,R1+2)=4 AND REEL(2,R2+2)=4 AND REEL(3,R3+2)=4 THEN  LET WIN=100
5000 IF REEL(1,R1+2)=5 AND REEL(2,R2+2)=5 AND REEL(3,R3+2)=5 THEN  LET WIN=70
5020 IF PEEL(1,R1+2)=6 AND REEL(2,R2+2)=6 AND REEL(3,R3+2)=6 THEN  LET WIN=50
5030 IF WIN=0 THEN  IF REEL(1,R1+2)=REEL(2,R2+2) THEN  LET WIN=20
5035 IF WIN=0 THEN  IF ST1=1 AND ST2=1 THEN  LET WIN=20
5040 RETURN
5100 REM
5120 REM [[[ DISPLAY WIN OPTIONS ]]]
5140 REM
5150 INK 3
5160 CSR 3,18: PRINT "NUDGES:";INT(WIN/10)
5180 CSR 17,18: PRINT "WIN SERIES:";INT(WIN/35)
5190 GOSUB 5380: REM GET WIN/LOSE
5200 INK 7: CSR 13,20: PRINT "CASH"
5220 INK 12: CSR 3,21: PRINT "LOSE"
5240 CSR 12,21: PRINT "COLLECT"
5260 CSR 24,21: PRINT "WIN"
5280 INK 8: CSR 3,22: PRINT LOSE
5300 CSR 23,22: PRINT GWIN
5320 INK 11: CSR 13,22: PRINT WIN
5340 RETURN
```

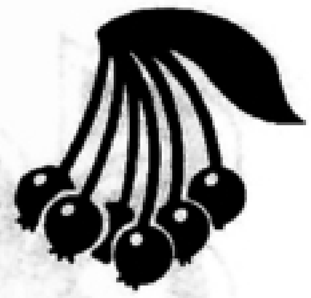
```
5380 REM
5400 FEM [[[ GET WIN/LOSE ]]]
5420 REM
5440 IF WIN=20 THEN  LET LOSE=10: LET GWIN=30
5460 IF WIN=30 THEN  LET LOSE=10: LET GWIN=50
5480 IF WIN=50 THEN  LET LOSE=20: LET GWIN=70
5500 IF WIN=70 THEN  LET LOSE=30: LET GWIN=100
5520 IF WIN=150 THEN  LET LOSE=70: LET GWIN=200
5540 IF WIN=100 THEN  LET LOSE=50: LET GWIN=150
5560 IF WIN=200 THEN  LET LOSE=100: LET GWIN=300
5580 RETURN
5600 FEM
5620 REM [[[ COLLECT WIN ]]]
5640 REM
5645 LET HOLD=HOLD+INT(RND)
5650 IF WIN>100 THEN  LET HOLD=HOLD+INT(RND*6)
5660 FOR X=WIN TO 10 STEP -10
5670 SOUND 3,3,15
5672 SOUND 2,200,0,100,0,2,1
5673 PAUSE 20
5674 SOUND 2,0,0,-50,0,5,0
5675 PAUSE 40: SOUND 2,0,0
5680 LET GOTOUT=GOTOUT+10
5700 INK 11: CSR 27,2: PRINT GOTOUT
5710 NEXT X
5715 PAUSE 60
5720 SOUND 3,0,0
5740 RETURN
5800 REM
5820 REM [[[ CLEAR DISPLAY ]]]
5840 REM
5860 FOR X=18 TO 22: CSR 1,X: PRINT "                                        ": NEXT X
5880 RETURN
6000 REM
6020 REM [[[ AUTO HOLD ]]]
6040 REM
6060 IF REEL(1,R1+2)=REEL(2,R2+2) AND REEL(3,R3+2)=REEL(1,R1+2) THEN  LET HOLD1=1: LET HOLD2=1:
LET HOLD3=1
6080 IF REEL(1,R1+2)=REEL(2,R2+2) THEN  LET HOLD1=1: LET HOLD2=1
6100 IF ST1=1 AND ST2=1 AND ST3=1 THEN  LET HOLD1=1: LET HOLD2=1: LET HOLD3=1
6120 IF ST1=1 AND ST2=1 THEN  LET HOLD1=1: LET HOLD2=1
6140 IF HOLD2<>1 AND REEL(1,R1+2)=REEL(3,R3+2) THEN  LET HOLD1=1: LET HOLD3=1
6150 IF HOLD2<>1 AND ST1=1 AND ST3=1 THEN  LET HOLD1=1: LET HOLD3=1
6152 IF HOLD1<>1 AND REEL(2,R2+2)=REEL(3,R3+2) THEN  LET HOLD2=1: LET HOLD3=1
6154 IF HOLD1<>1 AND ST2=1 AND ST3=1 THEN  LET HOLD2=1: LET HOLD3=1
6160 RETURN
6200 REM
6220 REM [[[ SHOW HOLDS ]]]
6240 REM
6260 SOUND 0,FLASH*10,4
6265 SOUND 1,FLASH*10+10,4
6270 SOUND 2,FLASH*10+20,4
6280 CSR 9,19: IF HOLD1=1 THEN  INK 14: PRINT "HELD" ELSE  INK FLASH: PRINT "HOLD"
6300 CSR 15,19: IF HOLD2=1 THEN  INK 14: PRINT "HELD" ELSE  INK FLASH: PRINT "HOLD"
6320 CSR 21,19: IF HOLD3=1 THEN  INK 14: PRINT "HELD" ELSE  INK FLASH: PRINT "HOLD"
6340 LET FLASH=MOD(FLASH+1,5)
6360 RETURN
6400 REM
6420 REM [[[ NUDGES ]]]
6430 REM
6435 GOSUB 10500: REM STOP SOUNDS
6440 LET HOLD1=0: LET HOLD2=0: LET HOLD3=0
6442 LET Y=WIN
6445 GOSUB 4000: REM SPIN
6450 GOSUB 4800: REM CHECK WIN
6455 IF WIN>0 THEN  LET R2=R2-1-20*(R2=0): GOSUB 3200: GOTO 6450
6457 LET WIN=Y
6460 GOSUB 5000: REM CLEAR DISPLAY
6470 IF WIN=300 THEN  GOTO 6580
6480 GOSUB 7000: REM DISPLAY NUDGE OPTIONS
6500 FOR Y=0 TO 16 STEP 2: SOUND 3,3,Y: LET KEY$=INKEY$: SOUND 2,700-WIN,Y: NEXT : IF KEY$="" TH
EN  GOTO 6500
6520 IF KEY$<>"G" AND KEY$<>"g" THEN  GOTO 6560
6540 IF RND>.45 THEN  LET WIN=GWIN: SOUND 1,960,-1,-50,0,19,1: GOTO 6470 ELSE  LET WIN=LOSE: FOR
 X=15 TO 0 STEP -1: SOUND 3,6,X: PAUSE 40: NEXT : GOTO 6580
6560 IF KEY$<"1" OR KEY$>"6" THEN  GOTO 6470
6580 GOSUB 5800: REM CLEAR DISPLAY
6590 IF WIN=10 THEN  LET EXIT=1: RETURN
6600 CSR 10,20: IF INT(WIN/10)=30 THEN  PRINT "UNLIMITED" ELSE  PRINT INT(WIN/10);" NUDGES"
6620 GOSUB 7500: REM DO NUDGES
6640 LET EXIT=1
6660 RETURN
7000 REM
7020 REM [[[ DISPLAY NUDGE OPTIONS ]]]
7040 REM
7060 GOSUB 5380: REM GET WIN/LOSE
7080 INK 7: CSR 13,19: PRINT "NUDGES"
7100 INK 12: CSR 3,21: PRINT "LOSE"
7120 CSR 12,21: PRINT "COLLECT"
7140 CSR 24,21: PRINT "WIN"
7160 INK 8: CSR 4,22: IF LOSE<20 THEN  PRINT "10p" ELSE  PRINT INT(LOSE/10);"  "
7180 CSR 24,22: IF GWIN=300 THEN  PRINT "UL " ELSE  PRINT INT(GWIN/10);"  "
7200 INK 11: CSR 13,22: PRINT INT(WIN/10)
7220 RETURN
```



```
7500 REM
7505 REM [[[ DO NUDGES ]]]
7510 REM
7512 LET NUDGES=INT(WIN/10)
7520 IF KEY$<"1" OR KEY$>"6" THEN  GOTO 7685
7560 LET R1=R1+(KEY$="1")-(KEY$="4")
7565 IF R1<0 THEN  LET R1=19
7570 IF R1>19 THEN  LET R1=0
7580 LET R2=R2+(KEY$="2")-(KEY$="5")
7585 IF R2<0 THEN  LET R2=19
7590 IF R2>19 THEN  LET R2=0
7600 LET R3=R3+(KEY$="3")-(KEY$="6")
7605 IF R3<0 THEN  LET R3=19
7610 IF R3>19 THEN  LET R3=0
7620 GOSUB 3200: REM SET POSITION
7630 FOR X=16 TO 0 STEP -1: SOUND 3,0,X: NEXT
7640 GOSUB 4800: REM CHECK FOR WIN
7660 IF WIN>0 THEN  RETURN
7600 IF NUDGES<>30 THEN  LET NUDGES=NUDGES-1
```

```
7682 CSR 10,20: IF NUDGES=30 THEN  PRINT "UNLIMITED" ELSE  PRINT NUDGES;" NUDGES"
7683 IF NUDGES<1 THEN  RETURN
7685 LET KEY$=INKEY$: IF KEY$="" THEN  GOTO 7685
7690 IF KEY$<"1" OR KEY$>"6" THEN  GOTO 7685
7700 GOTO 7560
7800 REM
7820 REM [[[ CHECK HISCORE ]]]
7840 REM
7850 LET Y=0
7860 FOR X=1 TO 8
7880 IF GOTOUT>HI(X) AND Y=0 THEN  LET Y=X
7900 NEXT X
7920 IF Y=0 THEN  RETURN
7940 FOR X=8 TO Y+0.9 STEP -1
7960 LET HI(X)=HI(X-1)
7980 LET HI$(X)=HI$(X-1)
8000 NEXT X
8020 CSR 2,10
8030 LET HI$(Y)="                    "
8040 INPUT "HI-SCORE ENTRY: ";A$
8042 IF LEN (A$)>20 THEN  LET HI$(Y)="A VERBOSE PERSON" ELSE  LET HI$(Y)=A$
8050 IF HI$(Y)="                    " THEN  LET HI$(Y)="A VERY LAZY PERSON"
8051 IF HI$(Y)="BILL                " THEN  LET HI$(Y)="BILL - NOT AGAIN !"
8060 LET HI(Y)=GOTOUT
8080 RETURN
8100 REM
8120 REM [[[ DISPLAY HISCORES ]]]
8140 REM
8150 CLS
8160 CSR 10,1: PRINT "TODAYS HIGH SCORES"
8180 FOR X=1 TO 8
8200 CSR 5,X*2+1: PRINT HI$(X)
8210 CSR 30,X*2+1: PRINT HI(X)
8220 NEXT X
8230 PRINT
8240 PRINT "     PRESS SPACE BAR TO PLAY"
8260 RETURN
8300 REM
8320 REM [[[ WINNER SPINS ]]]
8340 REM
8350 GOSUB 5800: REM CLEAR DISPLAY
8360 LET SPINS=INT(WIN/35)
8380 INK 7: CSR 11,19: PRINT "WIN SERIES"
8400 INK 12: CSR 3,21: PRINT "LOSE"
8420 CSR 12,21: PRINT "COLLECT"
8440 CSR 24,21: PRINT "WIN"
8460 IF SPINS>=10 THEN  GOSUB 5800: CSR 10,20: PRINT "10 SPINS": GOTO 8660
8470 INK 8: CSR 4,22: IF SPINS=1 THEN  PRINT "20p" ELSE  PRINT SPINS-1;"  "
8480 CSR 24,22: PRINT SPINS+1
8500 INK 11: CSR 13,22: PRINT SPINS
8520 FOR X=16 TO 0 STEP -2: SOUND 3,3,X: LET KEY$=INKEY$: SOUND 2,900-SPINS*40,X: NEXT : IF KEY$
="" THEN  GOTO 8520
8540 IF KEY$<>"G" AND KEY$<>"g" THEN  GOTO 8620
8560 IF RND>.5 THEN  LET SPINS=SPINS+1: SOUND 1,960,-1,-50,-SPINS,19,1: GOTO 8460 ELSE  FOR X=0
TO 16: SOUND 3,2,X: PAUSE 30: NEXT : LET SPINS=SPINS-1
8600 LET KEY$=" "
8620 IF KEY$<>" " AND SPINS<10 THEN  GOTO 8520
8660 GOSUB 5800: REM CLEAR DISPLAY
8680 GOSUB 8700: REM DO WIN SERIES
8685 RETURN
8700 REM
8720 REM [[[ DO WIN SERIES ]]]
8740 REM
8760 LET EXIT=1
8780 IF SPINS=0 THEN  LET WIN=20: FOR X=32 TO 0 STEP -2: SOUND 3,0,X: NEXT : RETURN
8790 LET Y=0
8800 LET R1=INT(RND*20)
8820 GOSUB 4500: REM CHECK STARS
8840 IF ST1<>1 THEN  GOTO 8880
8850 LET R2=WR2(8)
8860 IF RND>.4 THEN  LET R3=WR3(8) ELSE  LET R3=INT(RND*20)
8870 GOTO 8900
8880 LET R2=WR2(REEL(1,R1+2)+1)-1
8890 IF RND>.6 THEN  LET R3=WR3(REEL(1,R1+2)+1)-1 ELSE  LET R3=INT(RND*20)
8900 SOUND 1,10,-1,2000,0,70,1
8902 GOSUB 3200: REM DISPLAY REELS
8905 PAUSE 60: SOUND 1,0,0,0,0,1,1
8910 GOSUB 4800: REM CHECK WIN
8913 IF WIN=300 THEN  GOSUB 10000
8915 LET Y=Y+WIN: LET SPINS=SPINS-1
8920 CSR 3,19: INK 4: PRINT "SPINS LEFT    = ";SPINS;"  ": CSR 3,21: INK 5: PRINT "TOTAL SO FAR =
 ";Y;"   "
8930 IF INKEY$<>" " THEN  GOTO 8930
8950 IF SPINS<1 THEN  LET WIN=Y: RETURN
8960 GOTO 8800
10000 REM
10020 REM [[[ JACKPOT ]]]
10040 REM
10060 VS 5: INK 0: PAPER 12: CLS
10080 CSR 0,4
10100 PRINT "  cckcc  gk  lcck e  e mgci hcccd cckcc"
10120 PRINT "    j   g  k e    elc  e j e   j   j"
10140 PRINT "    j   mlln e    gl   gllf e   j   j"
10160 PRINT "  j j   e  j e    e cd e    e   j   j"
10180 PRINT "   mn   e  j illn e  j e    billa   j"
10200 CSR 0,13
10220 PRINT "          fcccl  fcci   hcccd hcccd"
10240 PRINT "            e        j   e   j e   j"
10260 PRINT "           cgc      lf l e   j e   j"
10280 PRINT "           cgc        j   e   j e   j"
10300 PRINT "         ll e         j   e   j e   j"
10320 PRINT "        jllobclll  illf   billa billa"
10340 PAPER 4
10360 FOR X=1 TO 20 STEP 4
10380 FOR Q=3 TO 15 STEP 2
10400 INK Q
10405 SOUND 3,MOD(X,7),15
10406 SOUND 2,Q*20,15
10410 FOR A=1 TO 15 STEP 3: SOUND 1,Q*A,A: NEXT A
10420 NEXT Q
10440 NEXT X: SOUND 0,0,0
10441 FOR X=1 TO 9 STEP 3
10442 SOUND 1,5,15,5,3,X*10,1
10443 PAPER X
10444 SOUND 2,1,0,50-X,0,X*10,1
10446 SOUND 3,7,15
10448 PAUSE X*200
10450 NEXT X
10455 SOUND 1,0,0: SOUND 2,0,0: SOUND 3,0,0
10460 LET HOLD=HOLD+INT(RND*2)
10480 VS 4: RETURN
10500 REM
10520 REM [[[ STOP SOUNDS ]]]
10540 REM
10560 FOR X=0 TO 3: SOUND X,0,0: NEXT
10580 RETURN
```





```
10600 REM
10620 REM [[[ MORE GRAPHICS ]]]
10640 REM
10660 FOR X=72 TO 200 STEP 48
10680 INK 14: PLOT X,64
10700 ANGLE 0
10720 DRAW 32
10740 PHI PI/2
10760 DRAW 96
10780 PHI PI/2
10800 DRAW 32
10820 PHI PI/2
10840 DRAW 96
10860 NEXT X
10880 INK 13: CSR 11,1: PRINT "RUIT    ACHINE"
10900 REM ___F___
10920 LET X=3
10930 FOR Y=87 TO 88
10940 PLOT Y,188
10960 ANGLE 3*PI/4: DRAW X
10980 ANGLE PI: DRAW X
11000 ANGLE 5*PI/4: DRAW X
11020 ANGLE 3*PI/2: DRAW X*2
11040 ANGLE 0: DRAW X: ANGLE PI: DRAW 2*X: ANGLE 0: DRAW X
11060 ANGLE 3*PI/2: DRAW X*4
11080 ANGLE 5*PI/4: DRAW X
11100 ANGLE PI: DRAW X
11120 ANGLE 3*PI/4: DRAW X
11140 PLOT Y+46,188
11160 REM ___M___
11180 ANGLE 3*PI/2: DRAW X*6: ANGLE PI/2: DRAW X*6
11200 ANGLE PI/4: DRAW X
11220 ANGLE 0: DRAW X
11240 ANGLE 7*PI/4: DRAW X
11250 ANGLE 3*PI/2: DRAW X*6: ANGLE PI/2: DRAW X*6
11260 ANGLE PI/4: DRAW X
11280 ANGLE 0: DRAW X
11300 ANGLE 7*PI/4: DRAW X
11320 ANGLE 3*PI/2: DRAW X*6
11340 NEXT Y
11360 RETURN
```

Tulips From Amsterdam

I would like to take this opportunity to thank all those marvellous people we met in Holland. The H.C.C show, in Utrecht, was a complete success for Syntaxsoft, and by the second afternoon we had sold out.

However, the best part of the trip was meeting the Memotech users of Holland. They are very nice, friendly people who made us feel at home. I also extend a big thank you to M.U.G.S (THE DUTCH MEMOTECH USER GROUP) who had the adjacent stand.

Although we were in Holland for business purposes the social side was hectic, to say the least. The lager flowed freely, and Michael had a smile on his face from stepping on the boat to returning to Burnley.

A big thank you goes to Rene ter Beek, who is one of the nicest people you could wish to meet and he gave his services willingly without being asked. Pity he has no sense of direction ! He is the only human I know that can get lost simply by blinking his eyes!!

I would also like to thank Nick Passmore for his generous hospitality - it is the only house I have ever been in that flushes the toilet with lager !! Jette, Nick's better half was a gem and apart from being beautiful, she is also a very good cook.

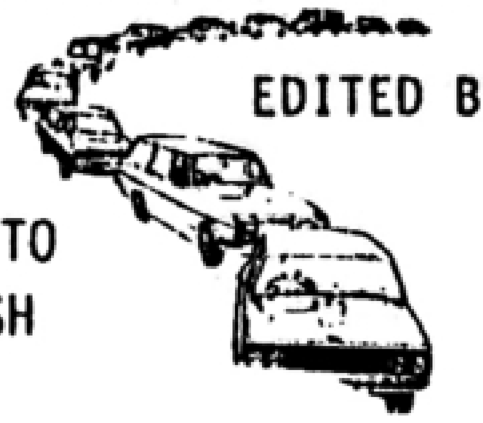
As you can tell, we all enjoyed ourselves, and I personally look forward to a return visit in the new year, and I promise that I shall speak your language by then.

Kh

Sale

NEW RS 232 INTERFACE FOR THE 500 OR 512 - £45 ono. CONTACT MR N DUNCAN ON BASINGSTOKE 59765 ANY EVENING AFTER 7.00 pm.

Assembly Line



EDITED BY TERRY TROTTER

SEND ALL SUBMISSIONS FOR THIS COLUMN TO
TERRY TROTTER, ASSEMBLY LINE, 65 MARSH
HOUSE,BILLINGHAM,CLEVELAND, TS23 2HW

Before launching into this month's routines I'd like to thank everyone who sent subroutines to me over the last month and for the warm reception the article generated, thanks, and keep those routines coming.

One point to note please, if the routines you send are of any appreciable length then it helps me greatly if the code is sent on cassette. I don't have unlimited time to key in your routines as my full time job keeps me rather busy (I'm a lecturer in microelectronics at a local college).

This month's routines show the variation in interests amongst MTX users, but I'll let the routines speak for themselves. Where a routine needs some BASIC to show how it is incorporated into the system that too has been given.

We start with some routines and macros written using EDASM to transfer data between the Z80 and the Texas VDP chip .

Macros:

```
DEFMAC  ("VRAMWR")  ;writes data held in the A register to
        OUT (1),A   ;VRAM  after setting up a write address
        END.        ;using one of the routines given below

DEFMAC  ("VRAMRD")  ;reads data from VRAM into the A
        IN A,(1)    ;register after setting up with one of
        END.        ;the read address routines.
```

Neither macro alters the registers or stack. Each requires two bytes every time macro is used.

Trace:

This routine will print the current BASIC line number when the ' BRK' key is used to stop a BASIC program. The number is printed at the top right hand corner of VS1 (the list screen) between chevrons, when READY appears.

There are three methods of implementing this routine:

1) Load your (troublesome ?) BASIC program and then assemble this code into line zero. The routine must be at the start of memory as it stands.
Then enter the following POKES directly from BASIC.
POKE 64154,64 (128 for the MTX500)
POKE 64153,7
POKE 64152,195
This will point a USER interrupt at the code in line zero.
Then POKE 64862,31 will turn on the trace facility and POKE 64862,15 will turn off the routine.

2) For the adventurous TRACE could be merged with your program, again to line zero. To do this, assemble TRACE into line zero, with NO comments, then save it to tape. When needed, load TRACE first, then use PANEL to add hex FB to system variable "VAZERO". Then load your program. Now use PANEL again to add hex FB to the following system variables, "NBTOP", "BASTOP", and "BASTPO". Put the original value back into "VAZERO", i.e. subtract hex FB from it. Exit PANEL. You should now have the programs merged and it's a good idea to save them to tape at this point.
Again the pokes given above are necessary.

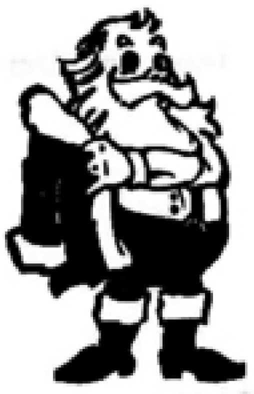
3) If you look back to issues 7 and 8 of MEMOPAD you will see some articles by Eric Roy on some "utilities". You can use the concepts contained in these articles to,
first move the code produced in TRACE to a high position in memory
and secondly to turn the TRACE function on and off from one of the function keys .

That's all for this month, see you again next month with routines to copy VRAM pattern tables to normal ram for use in sprites! and code to drive your printer directly from your assembly language program.

```
0 CODE

4007 TRACE:  PUSH AF      ;save main registers
4008         PUSH BC
4009         PUSH DE
400A         PUSH HL
400B         LD A,£FE     ;scan keyboard for BREAK
400D         OUT (£05),A
400F         IN A,(£06)
4011         BIT 0,A      ;test if set
4013         JR NZ,EXIT ;goto exit if no BREAK
4015         LD HL,(£FD6A)       ;get address of current BASIC token
4018         DEC HL       ;skip back & look for end of previous line
4019 LOOP:   DEC HL
401A         LD A,(HL)   ; FF is end of line
401B         CP £FF
401D         JR NZ,LOOP ;keep looking if multi statement line
401F         INC HL
4020         INC HL       ;skip forward to line number
4021         INC HL
4022         LD C,(HL)   ;put line number in BC
4023         INC HL
4024         LD B,(HL)
4025         CALL £0DD0 ;convert BC to ASCII char. at (DE)
4028         LD A,£EA     ;find number of ascii chars
402A         SUB E
402B         LD B,A       ;put in b
402C         LD A,(£FF58)        ;safe to write to screen ?
402F         OR A
4030         JR NZ,EXIT ;try later if not
4032         LD A,32      ;set screen address at CSR 0,32
4034         OUT (2),A
4036         LD A,£5C
4038         OUT (2),A
403A         LD A,"<"     ;print <
403C         OUT (1),A
403E LOOP1:  LD A,(DE)   ;print ascii chars. (line number)
403F         INC DE
4040         OUT (1),A
4042         DJNZ LOOP1 ;print them all (set in B)
4044         LD A,">"     ;print >
4046         OUT (1),A
4048 EXIT:   POP HL       ;restore registers
4049         POP DE
404A         POP BC
404B         POP AF
404C         RET

Symbols:
TRACE    4007    EXIT    4048
LOOP     4019    LOOP1   403E
```

THIS ROUTINE IS INTERUPT DRIVEN SO THE REGISTERS USED ARE SAVED AT ENTRY AND RESTORED AT EXIT.
THE STACK IS USED BY THE ROM CALL TO #0DD0, WHICH CONVERTS HEX NUMBERS IN BC TO ASCII CHARACTERS AT (DE)

A REG - ACCUMULATOR
BC REG - TEMP STORE THEN B AS A COUNTER
DE REG - POINTER USED BY ROM CALL
HL REG - POINTER



Title : VDPRGWR

Transfers data from register D (Z80) to register number A (VDP).

```
VDPRGWR PUSH BC      ;Save registers used.
        PUSH AF
        LD C,2       ;Select correct VDP mode.
        OUT (C),D   ;D = Data byte to put into register.
        OR 128       ;Ensure required bit is set
        OUT (C),A   ;A = Register to put data byte in.
        POP AF       ;Restore registers.
        POP BC
        RET
```

Register Usage:	Stack Usage:	Length:
A Register	2 words	13 Bytes
C Used to select port		
D Data		

No registers altered.

Title : VMWRADR

Sets up a write address to VRAM. Address to be written to should be in DE, where D contains the MSB's and E contains the LSB's.

```
VMWRADR PUSH AF      ;Store register
        LD A,E       ;Transfer E to A and send to VDP
        OUT (2),A
        LD A,D       ;Transfer D to A, set required bit and send to VDP
        OR 64
        OUT (2),A
        POP AF       ;Restore register
        RET
```

```
Register Usage:              Stack Usage:         Length:

A   Temporary store          1 word               11 bytes
DE  Address to set up
```

No registers altered.

If the E is changed to an L,and the D to an H in the above routine, HL will point to the required address.

Title : VMRDADR

As VMWRADR but sets up a read address.

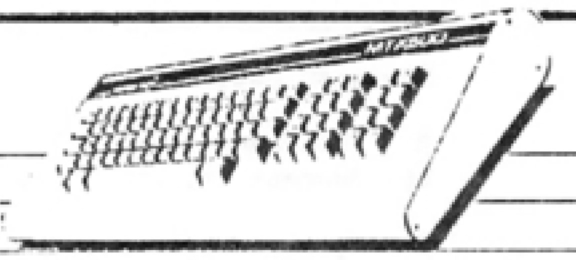
```
VMRDADR PUSH AF      ;Store register
        LD A,E       ;Transfer E and send to VDP
        OUT (2),A
        LD A,D       ;Transfer D, set correct bits and send to VDP
        AND 3FH
        OUT (2),A
        POP AF       ;Restore register
        RET
```

```
Register Usage:              Stack Usage:         Length:

A   Temporary store          1 word               11 bytes
DE  Address .
```

No registers altered.

Again the routine can be altered so as to use HL as the address pointer.

HARDWARE



PRICES EFFECTIVE FROM 1ST DECEMBER 1985

MTX 512	#119.00	MEMBERSHIP EXTRA
MTX 500	# 69.95	MEMBERSHIP EXTRA

32K MEMORY EXPANSION	#36.74
64K MEMORY EXPANSION	#45.43
128K MEMORY EXPANSION	#69.52

**** PLEASE STATE FOR WHICH MODEL ******

NEWWORD 32K ON ROM	#36.74
PASCAL 16K ON ROM	#36.74
SPECULATOR	#36.95

DMX80 PRINTER	#179.95
PRINTER CABLE	# 8.95

SDX 500K DRIVE + INTERFACE	#222.50
SDX 1MEG DRIVE + INTERFACE	#265.83

2ND 250K SDX DRIVE	# 89.00
2ND 500K SDX DRIVE	#169.00
2ND 1MEG SDX DRIVE	#203.00

FDX 2ND DRIVE 500K	#141.00
FDX 2ND 1MEG DRIVE	#163.00

DUST COVER	#3.50
DMX80 PRINTER RIBBON	#8.98

FLOPPY DISCS (BOX 10) GUARANTEED	#18.95
PACE NIGHTINGALE MODEM	#119.00 + #5.00 P&P
250K DISC DRIVE + INTERFACE	#199.00 + #5.00 P&P

DISC CASES HOLD 10 DISC	# 2.55
FLOPPICLENE DISC CLEANING KIT	#17.20
ANTISTATIC SCREEN WIPES ...(10)	# 1.50
DISC CABINETS (LOCKABLE) 110 DISCS	#36.95

CRIB CARD	# 2.16
ROM LISTINGS	#45.00
SDX CONTROLLER LISTINGS	#20.00
ROM CALLS INFO SHEET	50p
RST 10 CALLS INFO SHEET	50p
INTERRUPTS INFO SHEET	80p
DDT INFO BOOK	#2.00
VDP TECHNICAL MANUAL	#5.95
MTX SERVICE MANUAL	#6.95

80 COL CARD + CPM +NW/SC	#180.95
80 COL UPGRADE KIT (RS232)	# 27.00

PACKAGE ONE
SDX 500K DRIVE + INTERFACE
+80 COL BOARD +CPM + NW + SC ....#355.00

PACKAGE TWO
AS ABOVE BUT WITH 1MEG DRIVE ....#395.00

PACKAGE THREE
SDX 500K DRIVE : INTERFACE:
COMPUTER MTX512: 80 COL PCB:CPM ......#449.00

PACKAGE FOUR
AS ABOVE BUT WITH 1MEG DRIVE ....#489.00

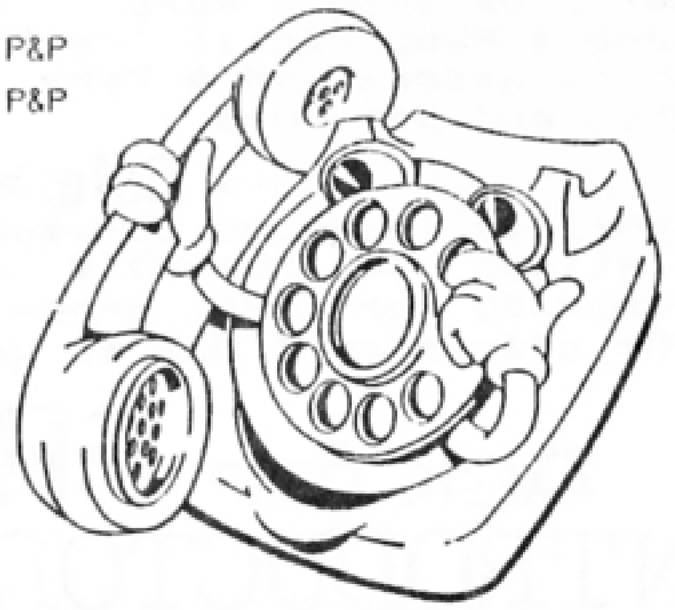
FDX SINGLE 500K CPM SYSTEM	#539.00
FDX SINGLE 1MEG CPM SYSTEM	#675.00

FDX TWIN 500K CPM SYSTEM	#569.00
FDX TWIN 1MEG CPM SYSTEM	#740.00

(REquires RS232 comms board)

SILICON DISCS

500K	#145.90
1MEG	#441.00



THE ABOVE PRICES ONLY APPLY TO U.K SALES & BFPO SALES
BFPO SHOULD ADD AN EXTRA £30.00 TO COVER ADMINISTRATION BY MEMOTECH

UPGRADE PACKAGE 1	£198.00
UPGRADE PACKAGE 2	£223.39

The above require factory fitting so add an extra £25 to cover cost of this service.

DON'T FORGET IF YOU HAVE A DISC DRIVE YOU SHOULD OWN A HIGH QUALITY HEAD CLEANER see FLOPPICLENE .... over half the problems handled by us are due to dirty disc heads.

SOFTWARE

00106	26X26 SPREAD SHEET	UTIL	SYNT	7.95	I	ANY
00057	3D TACHYON FIGHTER	ARC	CONT	6.95	I	ANY
00135	9 ELECTRIC. PROGS	EDUC	SSFT	13.95	I	ANY
00062	ADVENTURE QUEST	ADV	LVL9	8.75	I	ANY
00033	AGROVATOR	ARC	SYNT	5.95	I	512
00125	AIRCRAFT MAGNETISM	FLGT	AVTN	?	I	ANY
00120	AIRCRAFT PAYLOADS	FLGT	AVTN	?	I	ANY
00122	AIRLAW 2	FLGT	AVTN	?	I	ANY
00123	AIRSPEED INDICATOR	FLGT	AVTN	?	I	ANY
00071	ALICE IN WONDER.	ADV	CONT	6.02	I	ANY
00121	ALTIMETER	FLGT	AVTN	?	I	ANY
00008	ASTROMILON	ARC	CONT	6.02	I	ANY
00047	ASTROPAC	ARC	CONT	6.02	I	ANY
00058	BACKGAMMON	BRD	CONT	7.95	I	ANY
00041	BASIC BUSINESS	BS	CONT	5.95	I	ANY
00043	BLOBBO	ARC	CONT	6.02	I	ANY
00073	BOUNCING BILL	ARC	SYNT	4.95	I	ANY
00074	BRIDGE	CARD	CONT	6.95	I	512
00130	BUSINESS GAME	BRD	SSFT	15.95	I	ANY
00077	CANVAS	UTIL	CONT	7.95	I	ANY
00085	CAVES OF ORB	ADV	SYNT	5.95	I	512
00137	CESIL INTERPRETER	LANG	SSFT	5.95	I	ANY
00094	CHAMBEROIDS	ARC	MEGA	5.95	I	ANY
00059	CHESS	BRD	CONT	8.75	I	ANY
00053	COBRA	ARC	CONT	6.02	I	ANY
00025	COLOSSAL ADVENTURE	ADV	LVL9	8.75	I	ANY
00098	COMBAT	ARC	PANS	2.95	I	512
00028	COMPOSER	UTIL	XAV	13.00	I	ANY
00046	CONT RAIDERS	ARC	CONT	6.02	I	ANY
00099	CRIBBAGE	CARD	SCRP	2.95	E	ANY
00110	CRYSTAL	ARC	MEGA	5.95	I	ANY
00050	DEN.GOES BANANAS	ARC	SCRP	2.95	I	ANY
00011	DENNIS & CHICKEN	ARC	SCRP	2.95	I	ANY
00103	DENNIS AND CIRCUS	ARC	SCRP	2.95	I	ANY
00068	DOODLEBUG	ARC	SYNT	4.95	I	ANY
00108	DOWNSTREAM DANGER	ARC	MEGA	5.95	I	ANY
00096	DR. FRANKIE	ARC	SYNT	5.95	I	512
00056	DRAUGHTS	BRD	CONT	6.95	I	ANY
00111	DRIVE THE CEE 5	ARC	MEGA	5.95	I	ANY
00063	DUNGEON ADVENTURE	ADV	LVL9	8.75	I	ANY
00067	EDASM	UTIL	SYNT	7.95	I	512
00066	EMERALD ISLE	ADV	LVL9	5.95	I	ANY
00038	ESCAPE FROM ZARKOS	ARC	MEGA	5.95	I	ANY
00081	EXTENDED BASIC	6.95	SENT	6.95	I	ANY
00082	FATHOMS DEEP	ARC	MEGA	5.95	I	ANY
00090	FIG FORTH	LANG	SYNT	15.75	I	512
00055	FIREHOUSE FREDDIE	ARC	CONT	6.02	I	ANY
00021	FIRST LETTERS 1	EDUC	CONT	8.75	I	ANY
00092	FKEY DEFINER	UTIL	MEMB	6.95	I	ANY
00037	FLUMMOX	ARC	SYNT	5.95	I	512
00132	FRACT, PERCENTAGES	EDUC	SSFT	5.95	I	ANY
00052	GAUNTLET	ARC	CONT	6.02	U	ANY
00102	GHOSTLY CASTLE	ADV	PANS	2.95	I	ANY
00031	GOLDMINE	ARC	CONT	6.02	I	ANY
00069	GRAPHICS	UTIL	CONT	5.95	I	ANY
00087	H & L DUMP	UTIL	MEM	4.95	I	ANY
00072	HAWKWARS	ARC	SYNT	4.95	I	ANY
00065	HELI-MATHS	EDUC	CONT	5.95	I	ANY
00034	HUNCHY	ARC	SYNT	4.95	I	ANY
00083	ICEBERG	ARC	SYNT	4.95	I	ANY
00105	JET SET WILLY	ARC	SPRJ	6.95	E	ANY
00015	JOHNNY REB	WAR	LOTH	6.02	I	ANY
00097	JUMPING JACK FLASH	ARC	SYNT	5.95	I	512
00115	KARATE KING	ARC	MEGA	5.95	I	ANY
00016	KEY TO TIME	ADV	LUMP	6.02	I	ANY
00042	KILOPEDE	ARC	CONT	6.02	I	ANY
00019	KNUCKLES	ARC	CONT	7.95	I	ANY
00078	LES FLICS	ARC	PSS	6.95	E	ANY
00032	LITTLE DEVILS	ARC	SYNT	4.95	I	ANY
00024	LORDS OF TIME	ADV	LVL9	8.75	I	ANY
00035	M COMMAND & ARCAD.	ARC	SYNT	4.95	I	ANY
00070	MAN FROM GRANNY	ADV	SYNT	4.95	I	512
00104	MANIC MINER	ARC	SPRJ	6.95	E	ANY
00119	MAPS AND CHARTS	FLGT	AVTN	?	I	ANY
00126	MAPS AND CHARTS 1	FLGT	AVTN	?	I	ANY
00022	MATHS 1	EDUC	CONT	8.75	I	ANY
00013	MAXIMA	ARC	CONT	6.02	E	ANY
00086	MEMOCHEQUE	UTIL	SYNT	6.95	I	ANY
00075	MEMOSKETCH	UTIL	SYNT	7.95	I	ANY
00089	MINER DICK	ARC	XAV	6.95	I	ANY
00044	MISSION ALPHATRON	ARC	CONT	6.02	I	ANY
00030	MISSION OMEGA	ARC	SYNT	4.95	I	ANY
00054	MURDER AT MANOR	ADV	LUMP	6.02	I	ANY
00010	MUSIC PAD	UTIL	CONT	6.02	I	ANY
00003	NEMO	ARC	CONT	6.00	I	ANY
00131	NETWORK LOADER	UTIL	SSFT	8.95	I	ANY
00112	OBLITERATION ZONE	ARC	MEGA	5.95	I	ANY
00045	OBLOIDS	ARC	CONT	6.02	I	ANY
00129	PAINTBOX	UTIL	SYNT	5.95	I	ANY
00001	PAYROLL	UTIL	CONT	21.25	I	512
00005	PHAID	ARC	CONT	6.02	I	ANY
00061	PHYSICS 1	EDUC	CONT	8.75	I	ANY
00124	PILOT NAVIGATION	FLGT	AVTN	?	I	ANY
00012	PONT & BLACKJACK	CARD	CONT	6.02	I	ANY
00009	POT HOLE PETE	ARC	CONT	6.02	I	ANY
00040	PURCHASE LEDGER	BN	CONT	12.75	I	512
00048	QOGO	ARC	CONT	6.02	I	ANY
00076	QOGO 2	ARC	MEGA	5.95	I	ANY
00095	QUANTUM	ARC	SYNT	5.95	I	ANY
00109	QUAZZIA	ARC	MEGA	5.95	I	ANY
00107	RED MOON	ADV	LVL9	?	U	ANY
00127	RELATIVE VELOCITY	FLGT	AVTN	?	I	ANY
00064	RETURN TO EDEN	ADV	LVL9	8.75	I	ANY
00020	REVERSI	BRD	CONT	7.95	I	ANY
00114	ROLLA BEARING	ARC	MEGA	5.95	I	512
00100	RUTHLESS BASTARD	ARC	LSFT	2.50	I	512
00002	SALES LEDGER	UTIL	SYNT	15.75	I	512
00029	SALTY SAM	ARC	SYNT	4.95	I	ANY
00113	SEPULCRI SCELERATI	ARC	MEGA	5.95	I	512
00101	SLOOPY'S CHRISTMAS	ARC	PANS	2.95	I	ANY
00116	SMG	ARC	MEGA	5.95	I	ANY
00049	SNAPPO	ARC	CONT	6.02	I	ANY
00023	SNOWBALL	ADV	LVL9	8.75	I	ANY
00036	SON OF PETE	ARC	MEGA	5.95	I	ANY
00136	SOUND & RESISTORS	EDUC	SSFT	5.95	I	ANY
00026	SPELLI-COPTER	EDV	CONT	5.95	I	ANY
00080	SPOOLER	UTIL	MEM	4.95	I	ANY
00017	STAR COMMAND	ARC	CONT	6.95	I	ANY
00014	SUPA CODER	UTIL	SYNT	7.95	I	ANY
00084	SUPER BIKE	ARC	SYNT	4.95	I	ANY
00004	SUPER MINEFIELD	ARC	CONT	6.02	I	ANY
00093	SURFACE SCANNER	ARC	MEGA	5.95	I	ANY
00133	SYMMETRY & GLASS	EDUC	SSFT	5.95	I	ANY
00039	TAPE TO DISC	UTIL	MEM	6.95	I	ANY
00007	TAPEWORM	ARC	CONT	6.02	I	ANY
00088	TARGET ZONE	ARC	SYNT	6.95	I	ANY
00118	THE DESIGNER	UTIL	HALT	8.95	I	ANY
00128	THE WALL	ARC	SYNT	4.95	I	512
00051	THE ZOO GAME	ADV	CONT	6.02	I	512
00134	TITRATION,CHROMATO	EDUC	SSFT	5.95	I	ANY
00006	TOADO	ARC	CONT	6.02	I	ANY
00018	TURBO	ARC	CONT	6.95	I	ANY
00117	USER BASIC	UTIL	SYNT	8.95	I	ANY
00079	USER EXTEND	UTIL	MEM	7.95	I	ANY
00027	UTILITIES 1	UTIL	CONT	4.95	I	ANY
00091	VERNON & VAMPIRES	ARC	SYNT	5.95	I	ANY
00138	WOOD SIMULATION	EDUC	SSFT	5.95	I	ANY
00060	WORD & PICTURE	EDUC	CONT	8.75	I	ANY

PLEASE ONLY ORDER THOSE MARKED " I "

KEY:- STOCK NUMBER TITLE TYPE HOUSE MEMBER PRICE STOCK MACHINE

E=EXPECTED SOON I = IN STOCK U = UNAVAILABLE AT PRESENT

SUBSCRIPTIONS

IF YOUR MEMBERSHIP NUMBER IS BETWEEN 0 - 1278 INCLUSIVE YOU ARE NOW DUE TO RENEW YOUR SUBSCRIPTIONS. PLEASE ENSURE THAT YOUR RENEWAL REACHES BEFORE THE 20th JANUARY 1986 TO MAKE SURE OF RECEIVING THE NEXT EDITION OF MEMOPAD.

Raffle

THE DISC DRIVE RAFFLE HAS BEEN WON BY MR P.S.SPALDING OF ROCHDALE WHO HAS NOW RECEIVED THE DRIVE AND SHOULD NOW HAVE A VERY HAPPY CHRISTMAS.

Forth**

If you have a faulty version of Disc Forth, the following line will solve the problem:-

```
POKE 25000,48:USER WRITE "FORTH",16640,9000
```

```
100 REM  USER BASIC CLOSE FILE ERROR
110 REM       MTX 512  VERSION
120 REM---------------------------------
130 REM  Load USER BASIC type in and
140 REM  run the following program to
150 REM  correct the close file error
160 REM---------------------------------
200 RESTORE 400
210 FOR L=16816 TO 16829
220 READ N
230 POKE L,N
240 NEXT L
250 RESTORE 500
260 FOR L=16776 TO 16801
270 READ N
280 POKE L,N
290 NEXT L
300 POKE 16866,213: POKE 16870,175
310 POKE 18943,170: POKE 18959,130
320 STOP
330 REM---------------------------------
340 REM  Close file error pokes.
350 REM---------------------------------
400 DATA 19,205,199,65
410 DATA 195,166,40,0
420 DATA 33,48,74
430 DATA 205,156,64
440 REM---------------------------------
450 REM Following pokes makes version 1
460 REM the same as version 1.2
470 REM---------------------------------
500 DATA 19,247
510 DATA 194,189,73
520 DATA 205,222,6
530 DATA 194,195,73
540 DATA 195,150,40
550 DATA 33,43,74
560 DATA 205,156,64
570 DATA 205,241,65
580 DATA 24,232,0
```

```
100 REM  USER BASIC CLOSE FILE ERROR
110 REM       MTX 500  VERSION
120 REM---------------------------------
130 REM  Load USER BASIC type in and
140 REM  run the following program to
150 REM  correct the close file error
160 REM---------------------------------
200 RESTORE 400
210 FOR L=33200 TO 33213
220 READ N
230 POKE L,N
240 NEXT L
250 RESTORE 500
260 FOR L=33160 TO 33185
270 READ N
280 POKE L,N
290 NEXT L
300 POKE 33250,213: POKE 33254,175
310 POKE 35327,170: POKE 35343,130
320 STOP
330 REM---------------------------------
340 REM  Close file error pokes.
350 REM---------------------------------
400 DATA 19,205,199,129
410 DATA 195,166,40,0
420 DATA 33,48,138
430 DATA 205,156,128
440 REM---------------------------------
450 REM Following pokes makes version 1
460 REM the same as version 1.2
470 REM---------------------------------
500 DATA 19,247
510 DATA 194,189,137
520 DATA 205,222,6
530 DATA 194,195,137
540 DATA 195,150,40
550 DATA 33,43,138
560 DATA 205,156,128
570 DATA 205,241,129
580 DATA 24,232,0
```

If you have any problems with USER BASIC

FRIC ROY contact me at
1,Orchard St.,
Kilmarnock
Ayrshire
KA3 1EB
Tel. 0563 34684

Please enclose SAE, state version 512, 500.