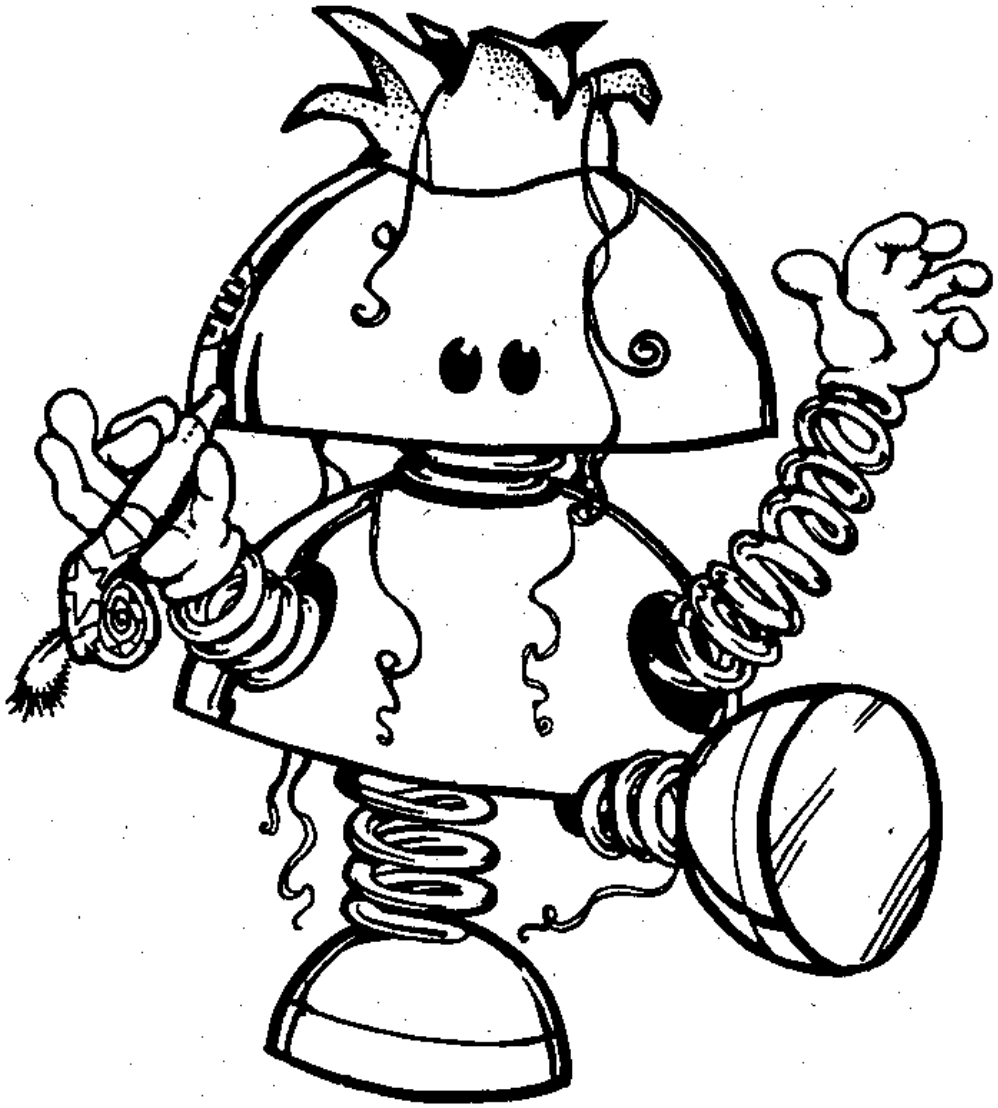
Vol 3 – No 2. October 1989.

FORMAT

FOR SPECTRUM USERS



SAM PEOPLE HAVE ALL THE LUCK

BUSINESS AND PLEASURE

The MGT Lifetime Drive. A Citizen 1Mbyte 3½" drive with its own power supply. Works with your DISCiPLE or PLUS D. Also works with your Q.L., Amiga, ST, BBC or even PC! All you change is the cable and switches. A lifetime of business and pleasure!

Lifetime Drive.....**£129.95** inc VAT
Spectrum/BBC Cable.....**£10.00** inc VAT
All other cables (each).....**£18.50** inc VAT
Shipping charge.....**£4.50**

MGT
MILES GORDON TECHNOLOGY

Miles Gordon Technology plc

Lakeside, Phoenix Way,
Swansea Enterprise Park,
Swansea, SA7 9EH ☎ 0792 791100

CONTENTS

FORMAT is published by FORMAT PUBLICATIONS. 34 Bourton Road, Gloucester, GL4 0LE, England. Telephone 0452-412572. DISCiPLE , PLUS D and SAM Coupe are trade marks of MILES GORDON TECHNOLOGY plc. Lakeside Technology Park, Phoenix Way, Swansea, South Wales, SA7 9EH. Telephone 0792-791100.

Printed by D.S.LITHO. Gloucester. Tel:- (0452) 23198.

NEWS ON 4

SAM's 128 COLOURS.

At one stroke, with a magic wave of his wand, Bruce Gordon has doubled the colour pallet of the new SAM Coupe computer. As if 64 colours weren't enough on a micro Bruce has upped the count to 128. For more details see this months SAM SPOT.

EPSON JOIN THE PRICE WAR.

Epson (UK) is set to join the fast developing price war in the 24 pin dot matrix printer market. October will see the launch of the low cost (£299) LQ400 printer. This takes Epson into direct competition with Citizen who recently launched their Swift 24 and with Star who produce the LC24-15.

The last years fluctuations in the printer industry, caused by European import tariffs, has led to greater competition in the dot matrix field. Most printer manufactures now seem to be aiming most of their efforts at the 24 pin market which can only bring prices down even further.

BBC DROPS TELESOFTWARE.

Following the demise of NEXT (FORMAT Vol 2 No 12), the micro magazine on BBC 2's Ceefax teletext service, the BBC have now dropped telesoftware. Over the last few years the service was used to transmit software (mainly educational) for BBC and IBM computers, they never did get round to the decent micros like the Spectrum.

The service has been dropped to release the pages for new teletext sevices, but it has left many users VERY angry. To download software you needed a special adaptor which didn't come cheap. The adaptors can still be used to store frames of teletext information but little else.

TAMWORTH SHOW.

Tamworth in Staffs is to have its own micro show. Set for the 26th November the show will be held in the Tamworth Arts Centre. For further details contact:- D.Betts, 8 Healey, Lakeside Est, Tamworth, Staffs, B77 2RF.

ZX-GUARANTEED BOWS OUT.

Geoff Bobker, one of the old stalwarts of the Sinclair software scene, has hung up his Speccy for the last time. Well known for his Microdrive transfer software he now plans to spend his time playing with his IBM compatible (and, of course, watching Star Trek).

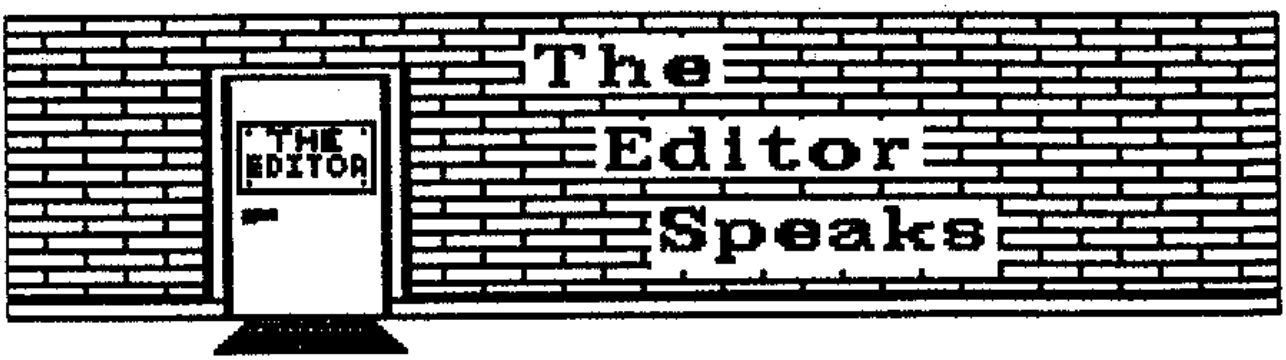
POST DELAYS.

The National Consumer Council has fired a broad-side at the Post Office. It claims that too many first-class letters are not reaching the destination the following day. Even though some improvement had been noted, since their last survey, the NCC still believe the public are getting a raw deal.

PSION LAPTOP

Psion are set to launch an A4 size 'laptop' version of their famous Organiser. Full details of availability and price will be released at the Personal Computer Show in London (21st September to 1st October).

URGENT we need your news. Clubs, Shows, New Releases, anything you think other people should know about. If you have any news items you want to pass on then send them in. Please mark the envelope NEWS in the top corner.



Let's start this month with a progress report on the FORMAT subscription drive. The special forms are still coming in, but a little on the slow side, so I thought you might like to share some of the ideas that have come from other readers.

Three avid FORMAT readers have visited their local computer clubs and bombarded those present with FORMAT subscription forms. Despite the fact that many clubs are now dominated by ST and Amiga owners, a lot of these still use their Spectrums for many things (have you seen the price of an Amiga wordprocessor?). Another reader has inserted an advert in his local free newspaper, most areas have at least one (Gloucester has three). The advert cost nothing and was worded along the lines - "Local Spectrum enthusiast looking to contact other users in the area. Ring Sid on 123456 after 6pm.". Simple, to the point, and it didn't cost him more than the price of a stamp. It's not a bad idea, you make a few friends locally, and if they then take out that subscription to FORMAT you could be our lucky winner.

SAM's coming out party was held in London this week, most of the magazines that count sent along their reporters (well there was champagne and a buffet on offer). A full report will be found in this months SAM SPOT.

Next, a little reminder. Whenever you write to FORMAT, especially if you are renewing your subscription or ordering back-issues etc, you MUST quote your membership number. I've had a couple of rude telephone calls recently, from one person in York, who complained that his back-issues had not been dispatched. On examination I found the order did not quote his membership number. He seemed very put-out and even issued veiled threats of legal action if I didn't send his back-issues immediately. Well it may interest him, and other readers, to note that it took over half an hour of searching through the files before I was able to find his number. Now if I had to do that with every order it would take all month and there would be no time to edit and produce FORMAT. With 1600+ members I think you will agree that it is not unreasonable to expect people to supply their membership number. And for all of you who have lost your original welcome letter, grab this months envelope. Your membership number is the 5 digit one, top left on the label, write it down and keep it safe. Quote your number and things can't go wrong.

This month I would like to welcome a new writer, Ian Cull, to our pages. Ian was writing the Tech-Nech section of CRASH until its recent plummet into the depths of pure juvenility. I look forward to many more articles from Ian and I particularly welcome his expertise on the +3.

Finally this month, I've had an accident! I managed to wipe 2 discs of articles for FORMAT. They all came from readers, but I don't know exactly what was on the discs. If anyone, who is not a regular contributor, sent in an article or program prior to mid August, that's not appeared in print (or been rejected) could you please give me a ring. At the same time I also lost this months SMALL ADS, if your advert hasn't appeared, please send it again.

Bob Brenchley. Editor.

TASCOLOUR

By: Brian Gaff.

The following program will help TASWORD 2 users who are bored with the old black & white screen. You can reset all the screen, border and margin colours and even try them out from within the program. You can also disable the old ZX Printer option in TW2, this avoids problems that can arise on some systems.

Line 10 loads the TW2 code file and line 400 resaves it, I've used microdrive syntax but this could be changed if needed for your system. Please check the start/length values in line 400 as some versions of TW2 are different.

Hints on the Basic program:- You could reset the colours, so that the menu page is different to the text-edit page, by inserting PAPER/INK controls into line 20. If you use TW2 on ANY 128K machine you must take the second RANDOMIZE statement (the one with the FN in it), out of line 250 to enable printing via the internal RS232 port. Also put RETURN in line 3000 to disable the Find/Replace option. If you don't then when you select it the machine will NEW out! TW2 uses the printer buffer to store the data.

```
 1 REM TASWORD 2 Colour modifier
 2 REM P.D. 1989 By Brian Gaff.
 3 REM alter load/save syntax (lines
   10,400 & 9999) to what you use.
 5 STOP
10 CLEAR 31999: PRINT #0;"Loading ta
   sword CODE": LOAD *"M";1;"tasword
   "CODE : INPUT ;
20 PRINT "TASWORD 2 SCREEN colour ch
   anger"
30 PRINT ' "Follow the prompts and y
   our     Tasword could be the envy
    of     everyone!"
50 INPUT "Border Colour? (0 - 7) ";b
   : IF b>7 OR b<0 THEN GOTO 50
60 INPUT  "Paper colour? (0 - 7) ";p
   : IF p>7 OR p<0 THEN GOTO 60
70 INPUT "Ink colour? (0 - 7) ";i: I
   F i>7 OR i<0 OR i =p THEN GOTO 70
80 INPUT "Status lines Paper? (0 - 7
   ) ";sp: IF sp>7 OR sp<0 THEN GOTO
    80
90 INPUT "Status lines Ink? (0 - 7)
   ";si: IF si>7 OR si<0 OR si = sp
    THEN GOTO 90
100 INPUT "Margin Paper? (0 - 7) ";mp
   : IF mp>7 OR mp<0 OR mp =p THEN G
   OTO 100
110 INPUT "Margin Ink? (0 - 7) ";mi:
   IF mi>7 OR mi<0 OR mi =mp THEN GO
   TO 110
115 INPUT "Disable ZX Print option? (
   y - n) "; LINE d$: IF d$(1)="y" T
   HEN POKE 60069,201: GOTO 200: REM
    ZX disable
116 POKE 60069,62: REM ZX enable
200 LET c=8: LET scr=i+p*c: LET statl
   =si+sp*c: LET stat2=sp+si*c: LET
   mar=mi+mp*c: LET ld=54
220 POKE 60641,b: POKE 64516,b: REM B
   ORDER 64/32 displays
230 POKE 58513,scr: POKE 58522,scr: P
   OKE 58551,scr: REM SCREEN 64/32 d
   isplays
240 POKE 64570,statl: POKE 59993,stat
   2: REM status lines
250 POKE 58509,mar: POKE 58518,mar: P
   OKE 58573,mar: POKE 58594,mar: RE
   M MARGINS 64/32 displays
260 POKE 58512,ld: POKE 58521,ld: POK
   E 58508,ld: POKE 58517,ld: POKE 5
   8550,ld: POKE 58572,ld: POKE 5859
   3,ld: REM LD instructions
270 PRINT #0;"Press a key to see effe
   ct": PAUSE 0: LET x=USR 59081: LE
   T x=USR 64330
280 PAPER p: INK i: CLS : PRINT #0;"<
   S>ave  <M>odify  <Q>uit": PAUSE 1
   : PAUSE 0
290 LET a$=INKEY$
300 GOTO (20 AND a$="m")+(400 AND a$=
   "s")+(9000 AND a$="q")
400 SAVE *"M";1;"tasword"CODE 54784,1
   0751
9000 STOP
9999 SAVE *"M";1;"tascol" LINE 10
```

DISCiPLE ANATOMY

By: Dick Guy.

Figure 1 is the circuit for the majority of the DISCiPLE disc system. I haven't included it all as parts are inter-related with other systems in the interface. For example addressing, system selection etc.

Before the circuit description let us consider the purpose of this circuit. To most it will be obvious, it's to enable us to save and reload the data we wish to keep and use. However, as with a lot of things, this is an oversimplification of what happens. To be more precise, the purpose is to convert the parallel data from the computer into a serial format which can be reliably saved, and, in such a manner that we can readily recover from disc the same serial data we put there, for reconvertion to the parallel format that the computer understands.

To enable us to achieve this complicated process using standard discrete components would involve a complex circuit design. It is however one of those processes to which integrated circuits are eminently suited and it wasn't to long before IC manufacturers were bringing them out. An American Company, Western Digital, soon became the leader in this field and the IC used in the DISCiPLE is one of theirs. The device used is a few years old now but is ideal for the type of system in use.

As with everything electronic various disc format standards came along. Fortunately for us in this instance, when the early disc systems came out there was only one major firm using them. This of course was the American giant IBM, who soon laid down its own standard formats for the 8

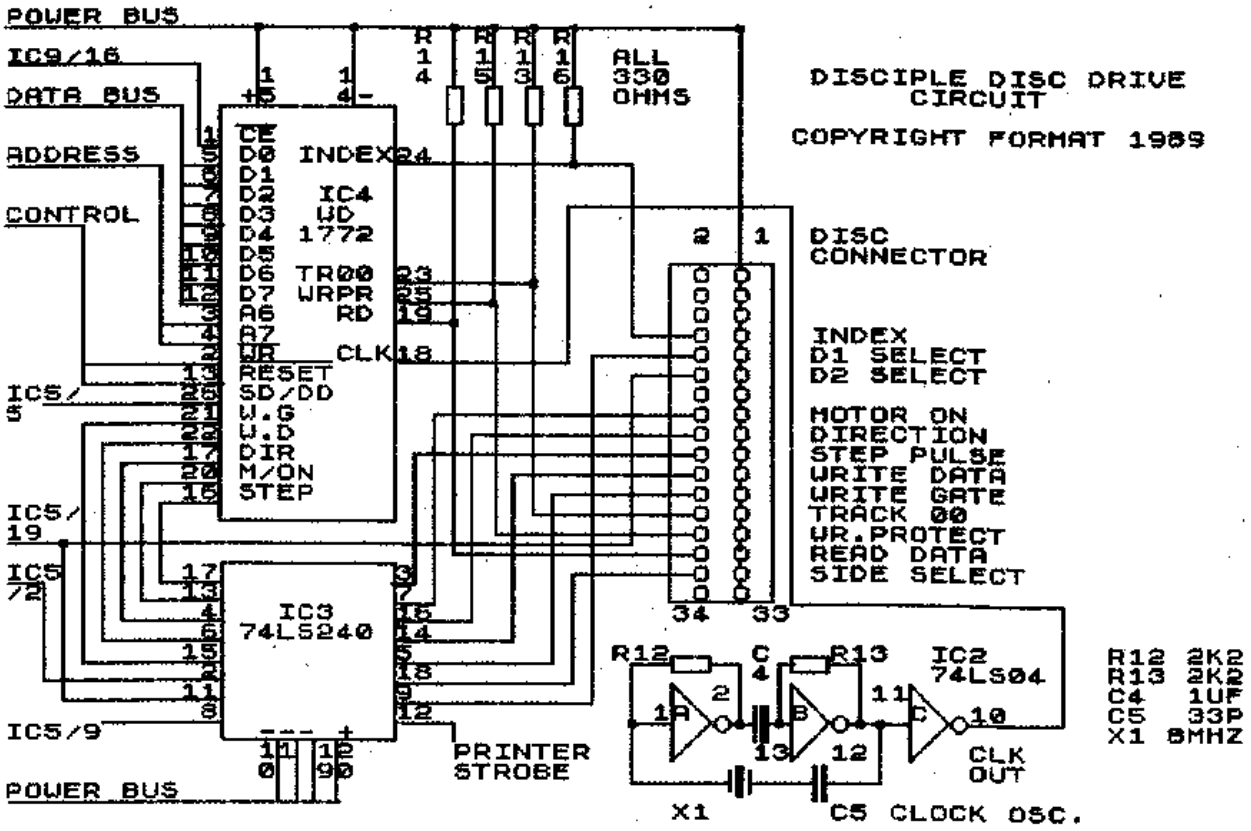


Fig 1

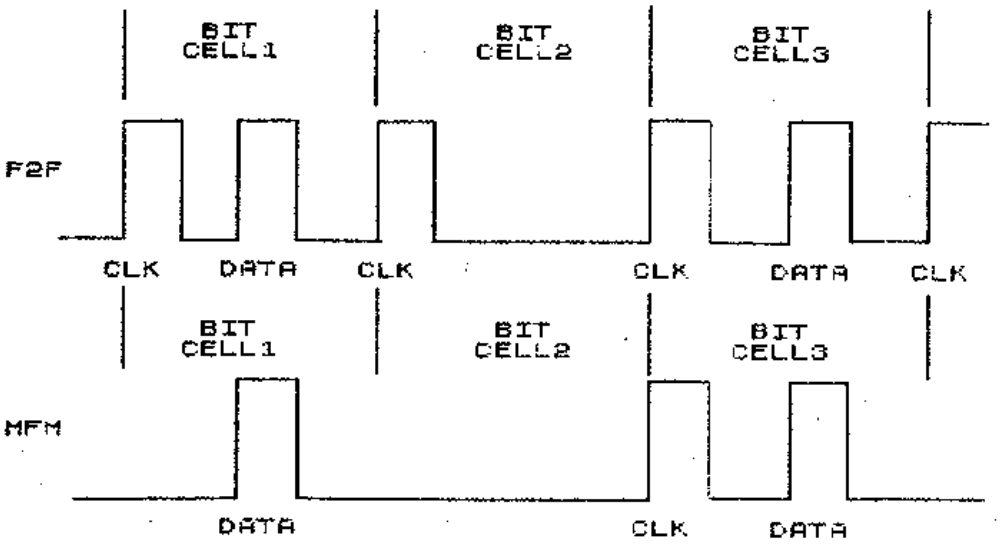


Fig 2

inch disc then in use and these formats have since become the industrial standard for all disc sizes. They aren't the only ones in use, APPLE for instance used their own for a long time.

The early IBM format became known as the 3740 Single Density format. This was also known as the FM (for frequency modulation), or F2F mode. As the name implies this is a single density mode of recording which, those of you that use it will know, wastes a large amount of disc storage space. Following improvements to the materials discs were made from a method of increasing storage density was sought. The solution became IBM standard 34 double density format (Also known as MFM for modified frequency modulation).

Figure 2 demonstrates the difference between the two formats. In brief with F2F format every Bit Cell, as each serial bit to be recorded became known, a bit at the clock frequency is recorded at the start of the cell time period. If a data bit is also to be recorded this is put centrally between two clock pulses. If there is no data bit then nothing appears between the clock pulses. On replay, where a data bit was recorded, the clock frequency is effectively doubled. The replay electronics makes use of the effect to reassemble the recorded data.

It is evident that this approach is wasteful of disc space. With MFM the bit cell start pulse is no longer used unless - that is - the previous bit cell was a zero. Evidently data density can be improved but at the expense of complexity.

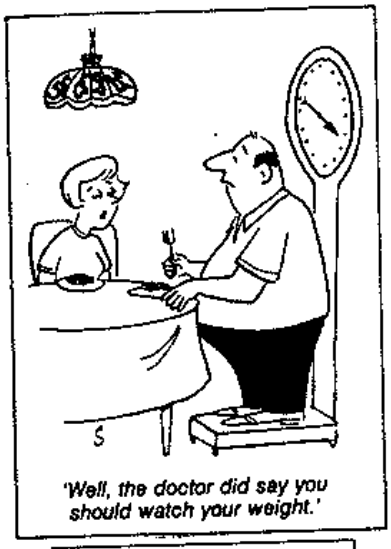
The disc controller in the DISCiPLE makes use of both formats to give single or double density recording. Which is determined by the logic level on pin 26 of IC4.

I do not intend getting into the complexities of IC4 because, as the above may have made clear, its operation is complex and an article the size of FORMAT would be necessary to do it justice. If you wish for the full data I recommend contacting your local component supplier for the address of your nearest Western Digital supplier. As an overview this IC receives the data from the data bus, prepares it for recording in the selected format and after checking a disc is present in the selected drive, using the INDEX input from the drive positions the head at the appropriate track and records the data. The system knows where the head is through use of the TRACK 00 input. This is the outermost track on the disc and forms part of the DISCiPLE catalogue area. If a double sided drive is in use the required side is also selected by IC4. There is obviously a great deal of data flow between the computer and the disc controller. In order to prevent loss of data a number of registers are built into IC4. These are command, control and data registers and their use allows simultaneous operations to be carried out.

IC3 is a Tri-state (meaning 3), inverting buffer. Its main function here is to invert the logic levels required at the disc drive from those produced by IC4 and vice versa, for those produced by the disc. The tri state function is disabled by tying pins 1 and 19 low.

IC2 is a simple crystal oscillator circuit. Resistors R12 and R13 put the inverting gates into their "linear mode", (yes you can use them as audio amplifiers), arranged to give "positive feedback" through crystal X1 to select the required clock frequency. The final stage serves to buffer the output to IC4.

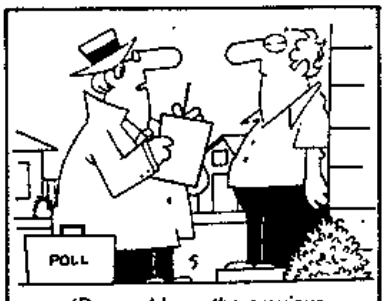
Thats all for this month. I hope you have understood the above descriptions and aren't to confused by what is a fascinating subject. Next month the printer and joystick circuits.



'Well, the doctor did say you should watch your weight.'

'Don't you think I look like a lawyer?'

YOUR
CARTOONS & JOKES
NEEDED TO FILL
THIS SPACE



'Do you blame the previous administration, the present administration or future administrations?'

YOUR LETTERS

STAR*LETTER* *STAR*LETTER

Dear Editor,

I was most interested and impressed by Ken Elston's INSTR$ article in the July'89 issue of FORMAT. It revealed a most ingenious and convenient way of passing parameters to machine code. However, Ken's program lacked one feature - a start number along the string being searched. As presented, all searches for A$ start at the first character of B$. So if A$ occurs again later in B$ the program can't search for it as it always finds the first occurrence.

The following program adds a new parameter, so the function now reads DEF FN I(A$,B$,S). Which can be read as "search for A$ in B$ starting at S". The start parameter should, more conventionally, be first eg (S,A$,B$) as in Beta Basic, but having it as the third simplified a little the amendments necessary to Ken's program.

Type in the program and RUN it, then try the Demo routine.

```
 10 REM Program to amend INSTR$.
 15 LOAD D*"INSTR$" CODE 30000
 20 FOR N=30091 TO 30068 STEP -1: POKE
    N+36,PEEK N: NEXT N: REM make roo
    m for newcode2
 30 FOR N=30067 TO 30022 STEP -1: POKE
    N+28,PEEK N: NEXT N: REM make roo
    m for newcode1
 40 FOR N=30022 TO 30049: READ A: POKE
    N,A: NEXT N: REM insert newcode1
 50 FOR N=30096 TO 30103: READ A: POKE
    N,A: NEXT N: REM insert newcode2
 60 REM data for newcode1
 70 DATA 221,78,22,221,70,23,11,167
 80 DATA 237,66,56,88,229,221,110,13
 90 DATA 221,102,14,167,9,221,117,13
100 DATA 221,116,14,225
110 REM newcode2 data
120 DATA 221,78,22,221,70,23,11,9
130 REM adjust some relative jumps
140 POKE 30015,106: POKE 30054,67: POK
    E 30082,27: POKE 30120,205
150 SAVE d*"IN$CODE" CODE 30000,128
  5 REM * * DEMO * *
 10 DEF FN I(A$,B$,S)= USR 30000
 20 LET Y$="12345678901234567890123456
    7890": PRINT Y$
 30 INPUT "Enter Start","Target$"'N,X$
 40 PRINT N;TAB 5;X$,"="; FN I(X$,Y$,N
    ): GOTO 30
```

Yours Sincerely, Albert Olivera.

*Well done Albert, you must have worked hard to patch the routine so cleverly. **Ed.***

Dear Editor,

I have been reading FORMAT since issue 1, I have looked forward to each months copy with much anticipation. However, I do feel I need to voice a small complaint - why do you ignore the DISCiPLE these days, it's all PLUS D this and PLUS D that. Come on FORMAT, the DISCiPLE is the more powerful interface, lets see more coverage.

Yours Sincerely, Andrew Morris.

I can't agree Andrew, we have not been neglecting the DISCiPLE. The PLUS D is the more popular but, with the exception of DOS updates and the INSIDE G+DOS series nearly all the articles we publish (that are disc related) apply to the DISCiPLE as well as the PLUS D.

*Even allowing for the PLUS D specific articles there are now, due to the larger size, more pages of interest to DISCiPLE owners than there were in the early issues. I do admit that we no longer use the **DISCiPLE / PLUS D** form so much these days as it takes up to much room, but if you see PLUS D then you can read DISCiPLE unless told otherwise. **Ed.***

Letters printed may be edited for length or clarity. The writer of each months STAR LETTER wins an EXTRA 6 months subscription to FORMAT.

..SCROLLER.....

By: Terry Simpson.

This small machine code routine was written to jazz-up a display at a local scout fair. The routine scrolls a message across the screen at a chosen line. The Paper/Ink colours are set and the characters can be printed in any width (although widths greater than 4 are not really much use).

I've given the assembler listing for those of you who might want to modify it, but I have also included a byte listing. As you will see if you look at the demo program, I have used the DEF FN method of calling the machine code. This was explained in the excellent INSTR$ article by Ken Elston (FORMAT Vol 2 No 11). It makes passing data to machine code routines so much easier.

In the DEF FN line, A$ is the string you want to print, L is the line number (0-23) you want to print on, W is the width used to print each character, and C is the colour attribute calculated in the usual way by **Flash*128+Bright*64+Paper*8+INK.**

If the string end with CHR$ 255 then a return to basic is made, otherwise a return is made when any key is pressed.

The only part of the source code that needs a little explanation is the line 0160. This stores BC, why? Well when you call a routine with the Basic USR function the BC register pair hold the address of the routine. As the routine is too long to allow a jump-relative (JR) instruction to loop back to the start, I store BC then load it into HL (at line 1110) so I can do a JP (HL). As the storage area used is the ROM calculators area (MEMBOT) this makes the SCROLLER routine completely relocatable. It will even work in 128k mode, unlike several others I have seen published.

SCROLLER - THE SOURCE CODE

```
0010 ; Screen message scroller
0020 ; V1.14  July 1989.
0030 ; (c) FORMAT.
0040
0050 DEFADD EQU  23563
0060 KEYSCN EQU  654
0070 CHARS  EQU  23606
0080
0090 STHL   EQU  23698
0100 STBC   EQU  STHL+2
0110 WORK   EQU  STHL+4
0120
0130
0140        ORG  60000
0150
0160 START  LD   (STBC),BC
0170        LD   IX,(DEFADD)
0180        LD   C,(IX+13) ;LINE No
0190        LD   B,0
0200        LD   A,C
0210        RRCA
0220        RRCA
0230        RRCA
0240        LD   C,A
0250        AND  224
0260        XOR  B
0270        LD   L,A
0280        LD   A,C
0290        AND  3
0300        XOR  88
0310        LD   H,A
0320        LD   A,(IX+29) ;ATTR byte
0330        LD   B,32
0340 ATRFIL LD   (HL),A
0350        INC  HL
0360        DJNZ ATRFIL
0370        LD   C,(IX+13) ;LINE No
0380        LD   B,31
0390        LD   A,C
0400        RRCA
0410        RRCA
0420        RRCA
0430        AND  224
0440        XOR  B
0450        LD   L,A
0460        LD   A,C
0470        AND  24
0480        XOR  64
```

```
0490          LD   H,A
0500          LD   (STHL),HL
0510          LD   L,(IX+4)   ; STRING
0520          LD   H,(IX+5)   ; address
0530          LD   C,(IX+6)
0540          LD   B,(IX+7)
0550 MAIN     LD   A,(HL)     ; get next
0560          CP   255        ; if 255
0570          RET  Z          ; then end
0580          PUSH BC
0590          PUSH HL
0600          LD   L,A
0610          LD   H,0
0620          ADD  HL,HL
0630          ADD  HL,HL
0640          ADD  HL,HL
0650          LD   DE,(CHARS)
0660          ADD  HL,DE
0670          LD   DE,WORK
0680          LD   BC,8
0690          LDIR
0700          LD   B,8
0710 LOOP1    PUSH BC
0720          LD   HL,(STHL)
0730          LD   DE,WORK
0740          LD   B,8
0750 LOOP2    PUSH BC
0760          PUSH HL
0770          LD   A,(DE)
0780          RLA
0790          LD   (DE),A
0800          LD   B,(IX+21) ; WIDTH
0810 LOOP3    PUSH AF
0820          PUSH BC
0830          PUSH HL
0840          LD   B,32
0850 MOV1     RL   (HL)
0860          DEC  HL
0870          DJNZ MOV1
0880          POP  HL
0890          POP  BC
0900          POP  AF
0910          DJNZ LOOP3
0920          POP  HL
0930          POP  BC
0940          INC  H
0950          INC  DE
0960          DJNZ LOOP2
0970          HALT
0980          POP  BC
0990          DJNZ LOOP1
1000          CALL KEYSCN
1010          POP  HL
1020          POP  BC
1030          LD   A,E
1040          CP   255 ; Test for any
1050          RET  NZ  ; key pressed
1060          INC  HL
1070          DEC  BC
1080          LD   A,B
1090          OR   C
1100          JR   NZ,MAIN
1110          LD   HL,(STBC)
1120          JP   (HL)
1130
1140 END      EQU  $
1150 LENGTH EQU  END-START
```

SCROLLER - THE POKER

```
  1 REM Byte Poker.
 10 LET TOTAL=0
 20 FOR I=1 TO 159
 30 READ BYTE
 40 POKE 59999+I,BYTE
 50 LET TOTAL=TOTAL+BYTE
 60 NEXT I
 70 READ CHECK
 80 IF CHECK<>TOTAL THEN PRINT "ERROR
    IN DATA": STOP
 90 SAVE D1;"SCROLL_C" CODE 60000,159
100 DATA 237,67,148,92,221,42,11,92,2
    21,78,13,6,0,121,15,15,15,79,230,
    224
110 DATA 168,111,121,230,3,238,88,103
    ,221,126,29,6,32,119,35,16,252,22
    1,78,13
120 DATA 6,31,121,15,15,15,230,224,16
    8,111,121,230,24,238,64,103,34,14
    6,92,221
130 DATA 110,4,221,102,5,221,78,6,221
    ,70,7,126,254,255,200,197,229,111
    ,38,0
140 DATA 41,41,41,237,91,54,92,25,17,
    150,92,1,8,0,237,176,6,8,197,42
150 DATA 146,92,17,150,92,6,8,197,229
    ,26,23,18,221,70,21,245,197,229,6
    ,32
160 DATA 203,22,43,16,251,225,193,241
    ,16,241,225,193,36,19,16,227,118,
    193,16,214
170 DATA 205,142,2,225,193,123,254,25
    5,192,35,11,120,177,32,172,42,148
    ,92,233,17743
```

DEMO PROGRAM

```
 10 DEF FN X(A$,L,W,C)=USR 60000
 20 LOAD d1"SCROLL_C"CODE 60000
 30 LET T$="Read FORMAT the best Spec
    trum magazine in the WORLD.......
    ...."+ CHR$ 255
 40 LET T=FN X(T$,12,2,23)
 50 LET T$="This message should repea
    t untill you press a key........"
 60 LET T=FN X(T$,20,1,112)
```

ADVENTURE CORNER

By: Paul Rigby.

I thought this month that a little change would go down well with my loyal band of readers. A little while ago I talked to Walter Pooley, an established and respected adventure author for Spectrum. Walter has produced many adventuring gems, all of which I have played and thoroughly enjoyed. It was nice, therefore, to talk to the man behind the game.

FORMAT - So when did you begin adventuring?
WALTER - It was in 1981 when I borrowed a Texas TI-99/A. I started, but never completed, Scott Adam's Pirate Adventure on cartridge. Although I wasn't discouraged. The Dragon 32 was my first computer. I bought a game called Black Sanctum, from Dixons, and then Scott Adam's Mission Impossible, which was later changed to Secret Mission because of copyright problems with the TV series which was popular at the time. Later, I became tempted by Spectrum adventures such as those produced by Brian Howarth, Level 9 and so on.

FORMAT - Staying with 'firsts', what were the first adventures you produced?
WALTER - I think the first was Desert Island, then Castle Adventure, Mansion Quest and Mission X. More recently they've been followed by The Pyramid.

FORMAT - They were all produced via the Quill weren't they?
WALTER - Yes. The first ones with an early version with no Ramsave or Character Sets. But I've updated them since with Ramsave/load and so on.

FORMAT - Why the Quill?
WALTER - Well, I only know a smattering of Machine Code and Basic. I've never tried programming in anything else bar The Quill.

FORMAT - What is your opinion of the three utilities?
WALTER - The PAW is excellent. I rate the Quill very highly, it is best for the beginner. I don't think much of the GAC, though. It is difficult for the novice, even with the GACPAC there is no real improvement.

FORMAT - How do you choose subjects for your adventures?
WALTER - For Desert Island, I lifted the basic story from a type-in program published in a Tandy magazine. As far as the others, well, I've always wanted to do a castle adventure and a SF adventure and so on.

FORMAT - How do you go about selling your adventures?
WALTER - Mainly via private ads, which are not lucrative. They barely cover the initial outlay after I have bought the bulk tapes and jiffy bags - there is not a lot of profit at the end of the day. They were also sold via H&D's "What Now" adventure magazine as H&D adventures until H&D went bust.

FORMAT - How long does it take to write an adventure? Is there a fixed time?
WALTER - How long is a piece of string? There's just no answer.

FORMAT - I just wondered if you, as some authors do when they write a book, force yourself to sit down for so many hours per day. For example, four hours per day writing, coding, mapping and so on.
WALTER - Oh, no, no. At my age I'm a free agent. I can do them when I want to. However, Castle Adventure was up and running within a week. I got a couple of pieces of A4 and stuck them

together, drew the shape of the Castle, what I wanted to do with all the commands and then did it - and then (with and ironic laugh) about six weeks later, it was debugged!

FORMAT - So it takes quite a while to debug it?
WALTER - Oh, yes it does. Well, you're forever finding all kinds of stupid things so you go back to it and redo it and then put a couple of hours play in and then start again.

FORMAT - Do you find that there are many in the final version? Do you manage to eradicate all of them?
WALTER - Well there was one that nearly got through in Desert Island. While standing on a ladder, in one part, if you had accidentally put one more "UP" in you would never have got down the ladder again! But I managed to catch that one just before it was released. There are other routines which I, and other authors, put into an adventure which helps to get around the adventure quickly which can be left in accidently. Pressing "123" or "ZZZ" , for example, can take you to the middle of the adventure or the last location. It depends on what the author has rigged up.

FORMAT - Do you have many playtesters? or are you it?
WALTER - I throw them around to different people. (laughing) I've given more games away than I've ever sold, quite honestly! Basically anyone whose fool enough to have a go at playing them!

FORMAT - Do many people come back having found bugs?
WALTER - Not many, no. Except for that Desert Island one I mentioned. You'll get a few spelling mistakes, though.

FORMAT - Did the fact of mapping Castle Adventure and preparing the commands, etc speed up the work?
WALTER - No. Castle was mapped first but the others were not. Either of two things happen. You come from a complete idea of what you want to do, like Castle, or you start with a basic idea and add to it as you go along.

FORMAT - So you're basically sitting in front of a white screen and making it up as you go along?
WALTER - Yes. Most of them - I've done like that. I don't know if authors of books do a similar thing but you find that ideas normally come off the top of your head. You'll be clacking away and you get a scene in front of your mind, add it in and something else pops up. Once you get going it flows.

FORMAT - I've read many well-meaning journalists who say don't sit down in front of a screen and utility and start typing away. You should plan it all out, do maps, the plot and so on before you type in anything.
WALTER - Do what suits you best. Both methods work and I've done them both. Off the top of your head works a treat as long as you have a printer connected up so you can print out bits as you go, especially with things such as flag usage and just what your doing in the adventure.

FORMAT - But you have no preference either way?
WALTER - No. Either system works.

FORMAT - The new game you are working on now, Forgotten City, was that programmed off the top of your head?
WALTER - Straight of the top of my head, yes.

FORMAT - So, then, that means that the majority of your games were developed in that way. Castle Adventure was the only one which was planned. Why?
WALTER - I don't know. I was probably sitting there one night with a cup of coffee in one hand and a fag in the other, and out came the pencil and paper.

FORMAT - Do you, having used the Quill extensively along with any tricks and routines you may have picked up along the way, intend to still use them for any future adventures?
WALTER - Yes. Up to now I've just used the complete Gilsoft package: The Quill and The Illustrator even though I tend not to produce graphics I still use Illustrator so that I can use The Quill with The Patch. You have to do

it in stages, you Quill the adventure. Once you're happy with that you Illustrate it, Patch it and that's the final product. The Press is just a glorified Patch. The only problem with The Patch is that you have to keep two flags clear for it in The Quill which can be a bit of a nuisance because you're already short of flags.

FORMAT - What is your opinion of adventure reviewers? Do you have any strong feelings at all?
WALTER - Well, without naming names, there are a few characters who are obviously playing with a help sheet alongside them. They should never review a game in a million years! For example, a BBC adventure "Blood of the Mutineers, was reviewed recently. The reviewer said that it was simple and no adventurer would have any trouble whatsoever with it. That game has got one of the hardest starts I've ever seen! I've typed out a whole A4 page just as a "get you started"! Never in a million years could you call it easy. So all the reviewer has done is work off the crib sheet. To review an adventure you have to play through it. Until you have played it end to end there is no way you can write a fair review.

FORMAT - Is there anything in adventuring, as a hobby, which dismays or angers you.
WALTER - Since the advent of adventure magazines such as H&D adventuring, to a great extent, has been spoiled as a lot of people just pick a book up and read through the solution and then they cross it off the list and say "I've finished that". But they will not have got that wonderful feeling when you get that final message up and you've done it all by yourself with, maybe, just one phone call to a friend asking for help. The nearest I've got to sitting down with a solution was with "Jack The Ripper" (CRL) part 3 which was heavily bugged. I checked through a C64 version, which wasn't bugged, to see where the bugs were or were not in the Spectrum version.

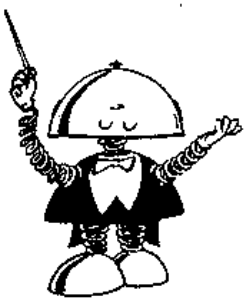
FORMAT - Do you have any all time favourite adventures?
WALTER - Lords of Time must be high on my list.

FORMAT - Any particular reason?
WALTER - Yes, because I enjoyed it! I thought it fitted together very well. Strangely enough The Snow Queen is another favourite. Blizzard's Pass was ruddy marvellous. It must qualify as the adventure to raise the jaded spirits of any adventurer. I thoroughly enjoyed that because it excepted all the old, short commands plus a full sentence parser. A very pleasant adventure overall, really. I don't know what lateral thinking means but you certainly had to think sideways in that one.

FORMAT - Finally, do you have any favourite "bad" adventures?
WALTER - The Hobbit. Which I don't think is a proper adventure. It was only as a result of hype that it got off the ground at all. Everywhere you go you have to constantly wait for something to happen, you don't need to visit half the locations. Its also bug-ridden. I have similar comments for Sherlock. Also The Island, by Crystal, and Woods of Winter.

I would like to thank Walter Pooley for his time and for putting up with me firing questions at him left, right and centre. I have play tested part one of his latest adventure, FORGOTTEN CITY, and have thoroughly enjoyed it, as I have his other adventures. They may not have the wordy prose of Magnetic Scrolls or Infocom but they give you all the information you need. His adventures are text only, to save memory. The puzzles are all logical, which is a nice change and I recommend any, or all of them, to beginners and experienced players alike. If you think you may like to try them then Walter has given me some details.

Mansion Quest, Castle Adventure, Desert Island and Mission X can be bought, on one tape, for £5.00 inc. of P+P. While the later Pyramid can be obtained for just £2.00 inc. from:- Walter Pooley, 46 Exeter Road, Bootle, Liverpool, L20 7BL. Please mention Format when you write.



THE SAM SPOT

SAM coupé

By Bob Brenchley.

Bruce Gordon & Alan Miles with the new SAM Coupe computer

SAM IS HERE! Well almost, but its had its Coming Out Party.

Tuesday 12th September was the day the press got to see SAM for the first time. The gathering, in London, was well attended. All the major computer magazines, many of the electronic mags, some newspapers, and (of course) FORMAT were represented. In many ways it was a typical press occasion, lots of wine, plenty of gossip etc. But there was also a great eagerness to see SAM. The technical reporters were keen to see what was under the wraps.

All the MGT team were there for the presentation. Bruce Gordon (THE MASTER), Alan Miles, Rob Collins, Andy Wright (author of SAM Basic), Bo Jangeborg (creater of SAM's graphic package), Bruce Everiss (publicity), David Whittaker (author of sound development software), Keith Turner (MIDI sequencer software), Mel Croucher (author of SAM's manual), Robin Evans (SAM artist),

Alan Miles started the presentation by talking about cars - cars were originally made for enthusiasts, then Henry Ford introduced Model 'T' and gave birth to the mass market. Style and fuctions were gradually introduced - different things gradually became important. Alan's Dad talked about a car's horse-power - now few people do. Todays models are differing sorts of

car for different purposes.

Similarly, Sinclair believed in the tenet that 'people didn't mind about the colour as long as it was black'. The mass market for home computers arrived. But you try and find people who will give you advice. It's there for PC's, but PC's are for businesses. When people bring out home computers, we're still concerned about the techie bits. MGT feels it's not as relevant in the field in which they're selling. Alan believes that the sorts of criteria which are being applied are positively dangerous. The danger is that the industry has stopped producing machines which are actually of use to people in the market.

Because MGT is a mail order company, they have tried to listen to what the customer wants. Also the level of service provided - to be there when the customer has a problem - is considered very important. That's why the presentation is not only an introduction to a computer - it's an introduction to SAM - an introduction to quality of Service.

Alan went on to give a history of the company and its range of products. He covered the growth that had been necessary to bring MGT to the point where they could cope with the launch of a mass-market product like SAM. He explained how the DISCiPLE and PLUS D had tested many key design ideas that were needed for a full computer.

Bruce Gordon then spoke about the technical aspects of the SAM Coupe. The most important component in SAM is the ASIC. Most of the computer is inside this customised gate array which he has designed. Two years ago SAM was to have been a 64k computer costing £99. But market research showed that the customer wanted something better. A lot of hard work, incorporated all the peripheral chips; four screen modes; extra speed; extra memory addressing; and lots more, produced the ASIC that now provides the power of SAM.

The ASIC itself also produces a low cost machine. Low chip count = less to go wrong in production = lower cost in production. It also makes for a more reliable product when in the hands of customers. Another aspect of the low chip count is that overseas production can be considered to meet local demand in countries like India.

The latest ASIC design had taken about 3 months to move from the 143 chip design board to the single ASIC chip. To start with, you only have computer aids to tell you if this works, but finally you plug it in and IT WORKS. Then there's CELEBRATION. And then... the supply Company tells you it can't meet your demand. So, off to another manufacturer and wait another three months.

The actual disadvantage of one manufacturer saying that MGT couldn't have what they wanted gave the opportuntiy of making a minor design change that leads to a MAJOR new feature in SAM. The colour palette now has 128 colours instead of 64. SAM was shown displaying a selection of art screens that had been copied from an Atari ST. Demos were also given of its advanced sound features and of the very fast Basic.

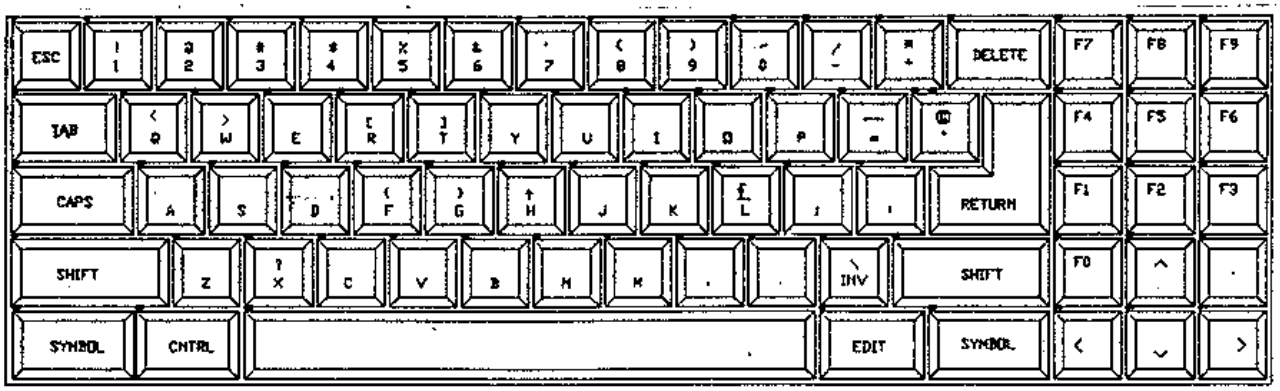
The magic of the Spectrum was for a customer to buy a basic machine and then add on, as and when they could afford it. In SAM it's nice to have good sound, good graphics, good memory etc. But it's also nice to have expansibility so that the Coupe is not the end - it's just the beginning. The Coupe is the first of the SAM family, it wont be the last, and everyone will be able to upgrade their existing Coupe to the later specs. For once it wont pay you to wait for the all-singing, all-dancing computer thats just round the corner. The SAM Coupe will grow as your needs grow.

The Party also saw the launch of a new cartoon character (see front cover and these pages) which will feature in the Coupe's manual, advertising and publicity. SAM - the Character - is not just a childish gimmic. They say a picture is worth a thousand words, the

SAM character will help to make the manual more readable and interesting. In advertising he will help to give SAM – the computer – an identity that will be recognisable – even among people with no interest in computers. Bruce Everiss, publicity consultant for MGT and the man for whome the term Industry Veteran was almost certainly invented, believes that computers now have to stand their own against other calls on peoples money like McDonalds (yuk), Kylie Minogue or the latest hit video. For SAM to succeed, outside the initial enthusiasts market, people must know of its existence and be able to identify the product. How many poor, unsuspecting, kids have said in the past (while thinking of their friends Spectrum) "Mum, I want a computer for Christmas" only to get a Commodore C16 on the day.

A brochure for SAM, together with a firm release date, should be sent out by MGT during the first half of October. If you haven't registered your interest in SAM and put your name on their mailing list, then write to MGT at the address on page 3.

Next month we will start to look at the hardware of the SAM Coupe, until then I will leave you with a little taster, a plan of SAM's keyboard. 72 keys including 10 function keys and a cursor cluster, full travel, and EVERY key redefinable from Basic.



NEV'S HELP PAGE

By: Nev Young.

Losing track of the stack is causing problems for A L Vernon of Looe Cornwall. He is trying to use the pokes given by Shimon Young on page 28 of FORMAT Vol 2 £11 to switch screens on his 128+2. The problem being that as soon as he pokes 23388 with 23 the machine crashes. Quite right too. What he has forgotten, and this was also missed from the article, was to move RAMTOP to below 49152.

The reason you have to do this is that the Z80 processor uses the memory just below RAMTOP for its machine stack. So if you page in bank 7 into the top 16K of memory the machine stack get paged out along with the rest of the original 16K.

Why does this matter you may ask. Simply put, as the Z80 interprets each line of the basic program it has to make many calls to other rom procedures. This is a machine code version of a GOSUB. When it has finished this call it does a RET and goes back to wherever it was when the call happened. Just like a RETURN in basic goes back to just after the GOSUB. The stack is used to store where the Z80 has to return to. If the stack is paged out it just doesn't know where to go and will either crash or hang when it reads whatever garbage happens to be where the stack used to be.

Everything will be fine, therefore, if you do CLEAR 49151 before starting.

M R Perry of Kidderminster is having problems with FLAGS. Not the ones you put up a pole but those named in Section 25 of the Spectrum manual. He is puzzled by the fact that although he POKEs a value of 0 or 8 to FLAGS2 at 23658 to set and reset the caps lock flag. When he comes to PEEK that value from within a BASIC program it holds 16 or 24. The answer is that each of the FLAGS is a byte of memory and as we all know a byte has 8 bits. Bit 3 of FLAGS2 is the caps lock flag. The other bits are used for other flags by the BASIC rom program. In this case the bits are:-

0 Screen needs to be cleared of an auto listing.
1 The ZX Printer buffer is not empty.
2 Inside quotes while scanning through an expression.
3 Caps Lock
4 Using K channel.
5 Not used
6 Not used
7 Not used

When you are running a basic program at the start of each line bit 4 will be set to indicate that input will come from the keyboard. This can change if the BASIC line changes channel. To a disc stream for example. When you are not running a basic program and enter commands directly you are using an R channel. This is why you get different results.

If you want to know the state of the caps lock flag and toggle that flag then use something like this:

```
LET A = PEEK 23658
LET A = INT (A/8)
LET A = 2*(A/2 - INT (A/2))
REM  A  now = 0 or 1 depending on CAPS LOCK.
POKE 23658, 8*(NOT A)
```

Now for an easy way to toggle the caps lock call USR 4317 and let the rom do it for you. Each call will toggle the state of the caps lock flag.

A few people are still having problems with their keyboards hanging

up PLUS D users can get around this by using the PLUS D interrupt patch published in Vol 2 £10. Which will unlock a keyboard by using the snapshot button. Except for A Vernon who can not get it to work. (Sorry can't see why. It works for me.) Hopefully somebody will work out a similar patch for us DISCiPLE users.

The Rev Chris Benyon of Guildford is pleased to report that the FIXER from MGT has solved a lot of his problems on his +3, although a bit of fiddling is still required. However he is under a miscomprehension of how save and load work. Some of his programs LOAD and SAVE to tape from machine code calls and he would like to know what to POKE to direct then to disc.

Although you can re-direct input and output on the spectrum by changing channels or streams often by POKEing some where into the program, LOAD and SAVE do not work in the same way. Much of what I said about converting from PLUS D to +3 also applies to converting from tape to disc loaders. Those programs written in machine code can be converted but the effort to do so is often excessive as often quite large sections of the program need to be changed.

Thats all for this month. Remember If you don't write to me I can't write this page. I also have to point out that I can not answer questions personally so DO NOT send me return postage etc. I will attempt to answer as many queries as possible but only through the magazine.

Write to FORMAT or directly to me at:-

FORMAT HELP PAGE.
3, Mitchell Place,
Falkirk,
Stirlingshire,
Scotland,
FK1 5PJ.

FASTER THAN BASIC

By: Ian Cull.

The Sinclair ZX Spectrum computer is, like almost all computers, built around a Central Processing Unit - the Z80 microprocessor chip. This chip is a very small piece of silicon packaged into a case about 3 inches by 1, and can execute (carry out) up to four million operations EVERY second!

The Spectrum does not quite run the Z80 at full speed, but can nevertheless carry out hundreds of thousands of operations every second. So why does your Basic program run so slowly? The answer is that the Z80 chip can only understand very simple instructions, that mean nothing to you or I. For example, the following instructions add 2 and 7 :-

```
00111110
00000010
11000110
00000111
```

and the Spectrum can do this about 250,000 times every second!

Unfortunately, when you write the following Basic program:-

```
10 LET A=2+7
```

the Spectrum can only manage to perform the sum about 385 times each second. The problem is that the Basic, which we can immediately understand, means nothing to the Z80. Special programs in the Spectrum ROM converts our Basic into instructions the Z80 can understand, and that is how our program is run. Of course, the converting takes a VERY long time for the Z80 to do, since it can only do very simple things. That's why the Basic version takes 500+ times longer to give results than the Z80 program.

There are, however, a number of ways that we can get our programs to work more quickly, and this series will look at doing things 'faster than (ZX) Basic'.

Tweaks

ZX Basic is a very 'user-friendly' Basic - anyone who has used any non-Sinclair computer will confirm this. ZX Basic checks each line of program as you type it in. You can also stop a program, alter it and then continue running it (most versions of Basic lose all the variables whenever you change something). Unfortunately, ZX Basic is also VERY slow. Most identically written programs will run quite a lot slower on the Spectrum than on many other computers. The reasons behind this are complex, but are caused by the way that ZX Basic was written (mainly by bolting extras on to the early ZX80 & ZX81 versions).

If you know the ways of ZX Basic, however, you can change your programs so that they run more quickly. Here are three examples :-

a) Define variables that are used a lot first. When the program references them, ZX Basic will find them first.

b) Put often-used subroutines and loops at the start of your program (and use a GOTO around them) - ZX Basic always searches for subroutines from the beginning of the program.

c) Avoid 'difficult' instructions if simple ones will do. For instance, multiplying by two is much slower for the Z80 to do than adding something to itself.

These and other 'tweaks' can be used in any program, and will have more or less effect depending on what the program is doing (for example, a very

short program won't be sped up much by 'b').

Alternative Languages

BASIC, Beginners All-purpose Symbolic Instruction Code, was originally developed to teach Fortran and was designed to be easy to learn, rather than efficient to use. Basic is very good at some tasks (in particular, string handling) but is not good at many things (try to handle more data than can be held in memory, for instance).

There are many other computer languages designed for many different reasons, and you will find that writing a program in a language other than Basic will often give much faster results. Of course, it may take much longer to write - this is why Basic is still popular as a 'quick and dirty' solution to many tasks; if the computer takes a long time to run the program that took much less time for you to write, then it must be time for another coffee!

In this series we will be looking at other languages available for the Spectrum, in particular how much faster they are than Basic compared with how much more difficult they are to write.

Machine Code.

The way to get your programs to run REALLY fast is to actually write them in the language that the Z80 understands - this language is called Machine Code, and is really difficult to learn and write. The example I gave earlier is an extreme way of writing machine code - the actual Binary representations of the instructions. Although those 1s and 0s are all that the Z80 understands, the task is made easier for us if we use an Assembler - this is a program that carries out the conversion from Assembler programs that we write, into machine code.

The earlier program in assembler looks like :-

```
LD A,2
ADD A,7
```

obviously this is easier for us to understand than the actual machine code, but it doesn't solve the problem that the Z80 can only do very simple things - we have to break down our program into hundreds of very simple instructions.

Compilers.

Here, in theory, is the answer to everyones dreams (and what this series is truly about). A Compiler is a little like an Assembler in that it takes a program and produces machine code. The difference is that the program can be written in a language that you already know - Basic, for example!

Compilers are very simple or very complicated, depending on how many tricks are programmed into them. At the simplest level, each line of the program is converted into an exactly equivalent set of machine code instructions - the increase in speed is due purely to the translation from program to machine code being done by the compiler, rather than as the program is run.

At the other extreme, Compilers can spend ages hunting through your program, deleting unnecessary code, re-arranging bits and generally producing machine code which can run almost as quickly as if you had written the program in assembler.

In order to get some idea of what improvements can be expected when using a different language, or a compiler, we use 'benchmarks'. These are simple programs which can be written in any computer language and times when run - the time gives an idea of how useful the language or compiler would be for real tasks. The problem with benchmarks is that they only test what they are written to test - which is NOT a real task. Some compilers can even spot common benchmark test programs, and 'cheat' by rewriting the test. Nevertheless, I am going to use some 'really useful' benchmarks in these tests - the programs will calculate the first one hundred Prime Numbers (numbers which

are only divided exactly by themselves or one - really useful!).

There are two programs, which do the same thing in two completely different ways. PRIMES1 calculates each prime by checking all the divisions (this is a very slow & difficult task for the Z80). PRIMES2 uses a 'sieve' which is like crossing off a list all multiples of each number; any numbers not crossed off must be prime (try it if you are not sure).

This month, we finish by presenting both listings in ZX Basic. Try them on your computer (and on other computers if you can) and try tweaking them - see how fast you can get them to run. Note that deleting the PRINT line (160 in PRIMES1, 130 in PRIMES2) is a fairer comparison, since it then does not time the displaying, only the calculating (The Spectrum is VERY slow at displaying things).

As a taster to keep you interested, a leter presentation will run PRIMES2 (without the PRINTing) in about half a second - thats about one hundred times Faster Than Basic.

```
 10 REM PRIMES1 in Basic.
 20 DIM P(1000)
 30 LET P(1)=2: LET PTOP=1
 40 LET PP=1: LET PPS=P(PP)*P(PP)
 50 LET P=3
100 IF PPS<P THEN LET PP=PP+1: LET PP
    S=P(PP)*P(PP): GOTO 100
110 FOR X=1 TO PP
120 IF INT(P/P(X)+.5)*P(X)=P THEN LET
     P=P+2:GOTO 100
130 NEXT X
140 LET PTOP=PTOP+1
150 LET P(PTOP)=P
160 PRINT PTOP,P: INPUT; :REM Remov t
    hi fo speed
170 IF PTOP<100 THEN LET P=P+2: GOTO
    100
180 PRINT "Prime 100 is ";P: STOP
```

* - * - * - *

WRITING FOR FORMAT

FORMAT is a magazine written by enthusiasts and for enthusiasts. We are always on the look-out for articles and programs to publish in FORMAT. Articles can be on any computer-related subject. They can be half a page or a long series or any length in between.

Don't worry too much about spelling and things like that (the Editor can't spell either) we will sort things out. Just put it down as clearly as you can. It is best if you send your article as a word processor file, on disc or tape, but please include a printed copy so we can look at it straight away. Pack any pictures flat or better still include SCREEN$ files so we can print them out here.

We are urgently looking for writers to produce articles on the following subjects:-

THE 128K ROM - EDUCATION

PASCAL - COMPUTER ART

We also require REVIEWS of software and hardware, both new and old. If you want to do a review please ring first to confirm no-one else is reviewing the same product.

Send your work to our address on page 3 or give us a ring to talk about it.

DOS COMMAND CODES

By:Bob Brenchley.

In this article I want to look at using OPENTYPE files from machine code. OTFOC - Command Code 70 (46 hex) is a DOS Command code available on DISCiPLE version 3c and above and PLUS D versions 1a/2a and above.

OTFOC stands for Open Type File Open and Close, the one Command Code does both jobs. The original GDOS and G+DOS did not contain this code so OPENTYPE files had to be opened from Basic, not a very good idea if you needed to use them from within a commercial program.

So it was FORMAT to the rescue (no, not on a white charger) with the updates we did last May and June (Vol 1 Nos 10 & 11). Now using files from machine code is really easy.

First you will need to set up the User File Information Area (see FORMAT Vol 2 No 4) like this:-

```
DSTR1   DEFB 1 ;drive number - 1 or 2
FSTR1   DEFB 223 ;OUT-223 OR IN-191
SSTR1   DEFB 4 ;stream number 4 - 15
LSTR1   DEFB 'd' ;disc type file
NSTR1   DEFB 10 ;file type
NSTR2   DEFM 'FILENAME  ' ;Length=10
HD00    DEFB 0 ;N/A
HD0B    DEFW 0 ;N/A
HD0D    DEFW 0 ;N/A
HD0F    DEFW 0 ;N/A
HD11    DEFW 0 ;N/A
```

Now use the Command HXFER to move UFIA into DRAM then OTFOC to open file.

```
OPEN    LD IX,DSTR1 ;point to UFIA
        RST 8
        DEFB 51 ;HXFER (33HEX)
        XOR A ;Zero A to flag OPEN
        RST 8
        DEFB 70 ;OTFOC
        JR C,ERROR ;carry set = error
        RET ;exit
```

I have left it to you to provide your own error routine, If an error has occurred then, after solving the problem, you should jump to OPEN again if you want to retry. Remember that the fact that a file already exists is treated as an error if you are trying to OPEN an OUTput file, you dont get the OVERWRITE Y or N message that you get from Basic.

The code in FSTR1 in the UFIA tells the DOS system weither you wont to open an **OUT**put file or an **IN**put file, the values are in fact the CODEs for the Sinclair Basic tokens IN and OUT (what a clever idea Bruce).

To write to an OPENTYPE file, make current channel same as output file by:-

```
WRITE   PUSH AF ;save A (the byte to
                ;print)
        LD A,SSTR1 ;A=stream number
        CALL 5633 ;1601hex - make
                  ;stream current
        POP AF ;recover A
        RST 16 ;print byte to stream
        RET ;return.
```

Of course you may need to re-allocate the default output stream (the main screen is stream #2) before returning from the WRITE routine.

To read a byte from a file (opened as an **IN**put file of course) the routine is similar to writing. A call is made to the main ROM routine at 5606 (15E6hex) which reads a bytes from the current stream.

```
READ    LD A,SSTR1 ;stream number
        CALL 5633 ;select stream
        CALL TEOF ;test end of file
        CALL 5606 ;read byte into A
        RET ;return
```

A count of characters to be read is stored in the channels area. The routine TEOF tests the three bytes of the count and jumps to END when the file is empty. It is assumed that IX still points to the channel, I leave you to write the END routine.

```
TEOF   LD A,(IX+31) ;Low order byte
       OR (IX+32)   ;Mid order byte
       OR (IX+18)   ;High order byte
       RET NZ ;return if bytes still
              ;left to read.
       JP END ;jump to EOF routine
```

Now all thats left to do is to close the file when you have finished. In the case of a write file, the file buffer will be written to disc (even if it is empty), the directory entry will be written and the channel area will be recovered. When CLOSEing a read file the channel area used is recovered and the stream is freed for future use. OTFOC is used to do this job but this time with the A register holding the stream number (4-15) you want to close.

```
CLOSE  LD A,SSTR1 ;Stream number
       RST 8
       DEFB 70 ;OTFOC to close file
```

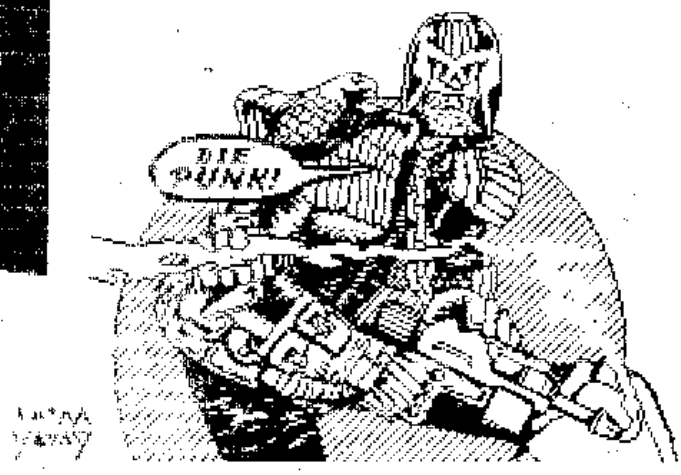
If you want to know more about streams and channels then I recommend you read the articles by Nev Young in FORMAT Vol.2 Issues 6 & 7.

Before I finish with OPENTYPE files there is just one small point that should be born in mind. You can have, subject to the number of streams available, any number of INput files open at once. These can be spread over both drives. But, while you can have more than one OUTput file open at a time they MUST be on the SAME drive. This is because there is only one 'File Sector Map' (the bit-map of free space on a disc used to find the next sector when writting a file) allowed in shadow memory at once.

Well thats all for this time. I have tried, over this irregular series of articles, to give you the building blocks that enable you to control the very powerful DISCiPLE & PLUS D disc systems from machine code. I don't pretend to have covered every Command Code in detail (its difficult to know what people need to know about) but if I've missed anything important I'm sure someone will let me know.

If you write any good routines using command codes then send them in, we will be pleased to print them in FORMAT.

ART SPOT



At Last! someone has sent in some computer art. Our thanks to Shim Young for these two excellent pictures. Now lets see lots more from readers.

THE SECRETS OF WORD MANAGER

SPECTRUM MACHINE CODE MADE EASY

Part 6. **By: Francis Miles.**

LOOPS. - Part 2.

The triple loop in the PAGE subroutine goes like this:-

```
1930 ;Prepare for triple loop.
1940 ;Put "screen zero" in HL'.
```

[HL' is the alternate HL register, reached by the command EXX, see last month's notes.]

```
1950        LD HL,(S0)
1960        CALL HLM
```

[S0 holds the "screen zero" (the character just before the first to be printed on screen) as a text address, ie counted from "text zero". HLM is a subroutine which converts text addresses into memory addresses, ie counted from zero in the memory. HLM and its opposite, HLT which does the reverse operation, must be the most commonly called subroutines in "WordManager".]

```
1970        EXX
```

[Now the memory address of "screen zero" is in the alternate HL register.]

```
1980 ;Is the print line 64 chars?
1990 ;Put F'=Z if yes.
2000        LD A,(SCRFL+1)
```

[The high byte of system variable SCRFL is zero for a 64-character line.]

```
2010        AND A
2020        PUSH AF
2030        EX AF,AF'
2040        POP AF
2050        JR Z,PG.3LP
```

[Exchanging AF with its alternate is quite independent of EXX. Now we have the zero flag in F' as a signal that the print line is 64 characters.]

```
...
2380 ;Put the screen location (scr)
2385 ;on the stack.
2390 ;Start with the 2nd pixel byte -
2400 ;all the character forms have
2405 ;zero top rows
2410 PG.3LP LD HL,16640
```

[16640 is 4100 hex, the first screen location. Now we enter the triple loop: both the outer loop and the second loop, the "thirds loop" and the "character loop", return to PG.64 with a new value of scr.]

```
2420 ;64-character lines?
2430 PG.64  PUSH HL ;scr
2440        EX AF,AF'
2450        JR Z,PG.6NO
```

[The jump is on the alternate flag F', which signals 64-character print lines.]

```
...
2550 ;Print two characters from HL'.
2560 ;Load A with the next char code.
2570 PG.6NO EX AF,AF'
```

[Putting the "64-char" flag back in the alternate F'.]

```
2580        EXX
2590        INC HL
2600        LD A,(HL)
2610        EXX
```

[Getting a character code from the alternate HL'. HL' was "screen zero" to start with, so the first A is "screen first", as it should be.]

```
2620        LD L,A
2630        LD H,0
```

```
2640 ;HL now holds the character code
2650 ;The character forms each take
2660 ;up 8 bytes and the first form
2670 ;is character 32, so the address
2680 ;of the second byte of char no
2690 ;HL is base-8x32+8xHL + 1 (for
2695 ;the second byte).
2700 ;Find the character form in the
2710 ;left font, and put its address
2715 ;in DE.
2720        ADD HL,HL
2730        ADD HL,HL
2740        ADD HL,HL ;HL is now x 8
2750        LD DE,CHR.L-255 ;8x32=256
```

[CHR.L and CHR.R are the base addresses of the "left font" character forms and the "right font" forms respectively.]

```
2760        ADD HL,DE
2770 ;Keep this address in DE.
2780        EX DE,HL
```

[It is the address of the second byte of the character form of the letter which is to go in the left half of the screen position which is now on the stack.]

```
2790 ;Load A with the next char code,
2800 ;find the form in right font,
2810 ;and put its address in HL.
```

[Repetition of the same procedure.]

```
2820 ;64-char lines?
2830        EX AF,AF'
2840        JR Z,PG.4NO
...
3000 ;Yes.
3010 PG.4NO EX AF,AF'
3020        EXX
3030        INC HL
3040        LD A,(HL)
3050        EXX
3060        LD L,A
3070        LD H,0
3080 ;HL holds the character code.
3090 ;The calculation to find the
3100 ;address is almost the same.
3110        ADD HL,HL
3120        ADD HL,HL
3130        ADD HL,HL
3140        LD BC,CHR.R-255
3150        ADD HL,BC
```

[But this time we leave the character form address in HL.]

```
3160 ;Paint the 7 elements of the
3170 ;double character into scr.
```

[This is the inner loop, the "pixel row loop".

```
3180        POP BC
3190        PUSH BC ; scr
```

[Lines 3210-3330 form the action of the inner loop.]

```
3200 ;Left form from DE,
3210        LD A,(DE)
3220 ;added to right form from HL.
3230        ADD A,(HL)
3240 ;The composite goes into BC.
3250        LD (BC),A
3260 ;The next byte of the same
3270 ;position in the screen buffer
3280 ;is reached by increasing the
3290 ;high byte of the address:
3295 ;4100 hex to 4200 hex, etc.
3300        INC B
3310 ;Get next byte of char forms.
3320        INC DE
3330        INC HL
```

[These six lines are repeated six more times. This is a sort of loop, but its high speed is so critical - the 6-line operation is repeated 10,752 times for each key stroke! - that I thought it better to repeat the commands rather than using any loop mechanism - which is bound to slow it down.]

```
3340        LD A,(DE)
3350        ADD A,(HL)
3360        LD (BC),A
3370        INC B
3380        INC DE
3390        INC HL
...
etc

3450 ;Next screen position.
3460        POP HL
3470        INC HL
3480 ;If L not zero, next character.
```

[See the screen number table displayed last month. Scr has been incremented from 4100 hex (or 4900 hex or 5100 hex) once per character; its low byte will not reach zero till it has gone

through all 100 hex bytes of the second pixel row of the screen third.]

```
3490        XOR A
3500        OR L
3510        JR NZ,PG.64
```

[This is the return from the next-to-outer loop, the character loop.]

```
3520 ;L is zero: next third.
3530        LD A,7
3540        ADD A,H
```

[Why 7? If we have just finished the first third, scr is now 4200 hex, having just moved from 41FF hex, the eighth byte of the last character of the eighth line in the first third. Adding 7 to H puts it on 4900 hex, the second byte of the first character in the first line of the second third.]

```
3550        LD H,A
3560 ;Finished?
```

[Scr is either 4900 hex, starting the second third, or 5100hex starting the third third, or 5900 hex, in which case it's all over.]

```
3570        LD A,59H
3580        CP H
3590        RET Z
3600        JP PG.64
```

[The return of the outermost loop, the thirds loop.]

This triple loop actually computes and pokes into the screen buffer 43,008 pixel dots (not counting the top pixel row of each character, which is left blank) on every key stroke; it is nevertheless so fast that it was necessary to include a short delay (not shown in these excerpts) to prevent it overtaking the typist and printing double characters.

Next Month we turn our attention to interrupts.

SHORT • SPOT

By: John Wase.

Oh dear! Last months SHORT SPOT caused a few problems for readers, if Bob's telephone is anything to go by. Daniel Neidle's Disc Backup Routine for the 128k Spectrum. There were two small typos, both in line 180 (but I bet most of you spotted them). First there should be no space between the 6 and the 5 in 65000, then there should be a colon (:) before the last GOTO in the line. The next two errors where caused when Daniel tried to renumber the program before sending it in, he forgot to test it after... The 128K's renumber routine will not alter 'computed' GOTOs (and it doesn't tell you so you could alter them by hand - it just ignores them). Two lines are affected by this, line 180 where the last GOTO should be GOTO 180 (not 130) and line 250 which should read GOTO 250 (not 200). One final problem in line 240. This should have read as follows:- 240 BORDER 0: LET I=PEEK 29998: LET J=PEEK 29999: LET M=30000

Everything will now work fine. However, Bob has had some reports of difficulty in entering some line of the routine. These have been traced to bugs in the 128K editor, if the line wont go in try adding spaces before and after all variable names (it takes up more memory but that can't be helped). The alternative is to type in all but the SAVE ! and LOAD ! lines in 48k mode (where everything will go in as printed), save to disc, then reload into 128k mode and finish off the routine.

Right, now on with the first of this months offerings. Derek Crabtree of Weybridge, a recent user of the PLUS D, writes that he uses his Spectrum (as a lot of us do) for a variety of technical purposes. In one of these he draws graphs on-screen. Whilst these look best as white on a black background, a dump to the printer gives disappointing results - the background is a mottled grey and the printer ribbon tends to die. The problem is easily overcome by resetting the paper to white, achieved by poking the attribute file with the code for white paper (eight times the number on the keyboard - number 7 for white: 8x7=56). So poking the attributes between addresses 22528 and 23295 with 56 should do nicely. The appropriate system variable should simultaneously be changed: Derek first tried the variable at address 23728, but this did nothing; poking 23296 with 56 did the trick. Derek wonders if anyone can explain why.

Derek first tried Basic which was far too slow, so he then wrote it in code which is satisfactory. Program 1 is a Basic program to provide the code: although this starts at 65000 (length 17 bytes, call with RANDOMIZE USR 65000), it can, in fact be located anywhere.

PROGRAM 1.

```
 1 REM **PAPER ATTRIBUTE SWOP*
 2 REM ***POKE 23296 WITH*****
 3 REM ***(8*PAPER COLOUR)***
 4 REM ******CALL WITH********
 5 REM *** RANDOMIZE USR 65000
 6 REM ======================
10 CLEAR 64999: LET TOT=0
20 FOR F=65000 TO 65016: READ A: POK
   E F,A: LET TOT=TOT+A: NEXT F
30 DATA 1,0,3,33,0,88,58,0,91,119,35
   ,11,120,177,32,246,201
40 IF TOT<>1215 THEN PRINT "ERROR IN
    DATA": STOP
50 STOP
60 SAVE d1"paper swop" LINE 1
```

Program 2 is an assembly listing, if you already have an assembler, and program 3 is a short demonstration program which draws an enclosed polygon with any number of coordinates. The paper colour is

selected (in Derek's case, white for the printer) and the routine then pokes the attribute file appropriately. Finally, just for completion, the paper is returned to black for the monitor display.

PROGRAM 2.
ASSEMBLY LISTING FOR ATTRIBUTE SWOP

```
00000          ORG 65000
00000          LD BC,768
00000          LD HL,22528
00000 START    LD A,(23296)
00000          LD (HL),A
00000          INC HL
00000          DEC BC
00000          LD A,B
00000          JR NZ,START
00000          RET
```

PROGRAM 3.

```
  1 REM ***PAPER ATTRIBUTE SWAP***
  2 REM *******DEMO PROGRAM*******
  3 REM *DRAWING WITH COORDINATES*
 30 CLEAR 64999: LOAD D*"paper swop"
    CODE
 40 INPUT "How many coordinates? ";N
 50 PRINT "NUMBER OF CORDINATES= ";N
 60 DIM C(N,2)
 70 FOR F=1 TO N 80 PRINT F;
 90 INPUT "x cor (<=255) =";X: IF X<
    0 OR X>255 THEN BEEP .1,05: GOTO
    90
100 PRINT TAB 4;"XCOR= ";X;
110 LET C(F,1)=X
120 INPUT "Y COR (<=175) = ";Y: IF Y<
    0 OR Y>175 THEN BEEP .1,05: GOTO
    120
130 LET C(F,2)=Y
140 PRINT TAB 18;"YCOR= ";Y
150 NEXT F
160 INPUT "PAPER COLOUR (0 TO 9) ";P:
     IF P<0 OR P>9 THEN BEEP .1,05: G
    OTO 160
170 POKE 23296,P*8: CLS: RANDOMIZE US
    R 65000
180 PLOT C(1,1),C(1,2)
190 FOR F=1 TO N
200 IF F<N THEN DRAW C(F+1,1)-C(F,1),
     C(F+1,2)-C(F,2): GOTO 220
210 IF N>2 THEN DRAW C(1,1)-C(N,1),C(
     1,2)-C(N,2)
220 NEXT F
230 PAUSE 100
240 PRINT £1;" PRESS ANY KEY FOR WHIT
    E PAPER ": PAUSE 0
250 POKE 23296,56: RANDOMIZE USR 6500
    0: PAUSE 100: PRINT AT 10,0;" PRE
    SS ANY KEY TO CONTINUE ": PAUSE 0
    : RUN
```

Finally, here's a piece for Basic beginners from Trevor Wright's booklet which I mentioned earlier this year. Basic logic can often be speeded and simplified by use of Boolean logic. For example, the statement:-

```
10 IF A=ALPHA THEN GOSUB 1000
```

is evaluated by treating the expression A=ALPHA as a number. (A=ALPHA) = 1 when this is true and zero when it is false. This allows you to rewrite lengthy and convoluted IF statements in one line. For instance:

```
100 IF A=ALPHA THEN LET X=X-1
```

can be written:

```
100 LET X=X-(A=ALPHA)
```

If A is not equal to ALPHA, then X remains unchanged, but if A and ALPHA are equal, then the value of the expression (A=ALPHA) is 1 and is subtracted from X.

Already, there is some saving in time, but because IF is not used, you can now combine a load of IF statements together. For instance:

```
 10 LET X=1000
100 REM MAIN LOOP
110 IF A=ALPHA THEN LET X=X+1
120 IF A>ALPHA THEN LET X=X-1
130 IF A<ALPHA THEN LET X=X+7
140 GOSUB X
```

can be rewritten:

```
 10 LET X=1000
100 REM MAIN LOOP
110 LET X=X+(A=ALPHA)-(A>ALPHA)+(A<AL
    PHA)*7: GOSUB X
```

Well that's all for this month. All the hairy vest. John Wase.

- - - - -

Contributions to SHORT SPOT should be sent to John Wase, Green Leys Cottage, Bishampton, Pershaw, Worcs, WR10 2LX.

FORMAT READERS SERVICES

By popular demand IT'S BACK. After a break of over a year FORMAT SOFTWARE returns to bring you those really useful items that you can't get elsewhere. Now combined with the Back-Issue service so it's easier to place orders. And there's lots more to come in the near future.

TAPE SOFTWARE

LCOPY 2

A set of routines which merge into your system file, replacing the existing printer dump routines. Enables Epson compatible printers that understand the Esc L code to do both SCREEN$ (1&2) and SNAPSHOT prints.

Order Code FST-01 **Price** £3.95

ARTCON 48

Converts the 48k OCP ART STUDIO to full DISCiPLE / PLUS D operation, includes printer driver.

Order Code FST-02 **Price** £3.95

TASCON 128

Converts TASWORD 128 for use with DISCiPLE or PLUS D. All features retained, includes TW2 file converter.

Order Code FST-05 **Price** £3.95

TASCON +2

Converts TASWORD +2 for use with DISCiPLE or PLUS D. All features retained, includes TW2 file converter.

Order Code FST-06 **Price** £3.95

G+DOS 2a

Latest version of the PLUS D system file. Works with ROM version 1a.

Order Code FST-90 **Price** £3.95

GDOS 3d

Up to date system file for DISCiPLEs with ROM version 3. Includes new set-up program, far easier to use.

Order Code FST-91 **Price** £3.95

Software codes starting FST are supplied on tape for use with any format/size of disc. Items coded FSD are supplied on 80 track disc (works with single or double sided) - please state 5.25" or 3.5".

- + - + - + -

FORMAT-BACK-ISSUES

For readers who have missed past issues of FORMAT (or perhaps worn theirs out through constant use) ALL back issues are still in print.

Volume 1	Issues #1 (Aug'87) To - #12 (Jul'88).
Volume 2	Issues #1 (Aug'88) To - #12 (Aug'89).
Volume 3	Issues #1 (Sep'88).

Order Code BKI + Vol & Issue Numbers.
Price £1.00 per copy

POSTAGE and ORDERING

Add the following charges, for postage and packing, to your total order.

	UK	OVERSEAS	AIR
Under £10	£1.00	£1.50	£2.50
£10 to £25	£1.50	£2.25	£3.50
Over £25	£FREE	£3.00	£4.50

Please **WRITE YOUR ORDER ON A PIECE OF PLAIN PAPER**, and mark envelope ORDER. Clearly state Order Code; Title; Quantity required and price. Payment in STERLING by Cheque (UK bank), P.O. Euro Cheques or Cash. Please make cheques payable to **FORMAT**.

Remember to quote your membership number or we wont be able to send out your order.

DO NOT include letters with your order as this will cause delays.

COMING SOON.....
Binders for your valuable issues of FORMAT. Disc conversions for TASWORD 3 and The WRITER. +Lots More.

GET THE MOST FROM YOUR DRIVE FACILITIES. *STEP UP TO*

NOT SO SOFT-WARE

SPECIFICALLY WRITTEN FOR PLUS D & DISCIPLE

SPECIFICALLY WRITTEN FOR PLUS D AND DISCIPLE · SPECIFICALLY WRITTEN FOR PLUS D AND DISCIPLE

FASTWORD PLUS

Here is a great programme that rejuvenates an old favourite. How often when using TASWORD TWO, have you been stuck for the right word? FASTWORD PLUS offers a complete reference book of words, waiting, ready for use at the press of a key!

This is the Plus D and Disciple Disc version of the sucessful program FASTWORD, a THESAURUS for the Spectrum. Features include ● Full alphabetical index of over 6,000 words ● A display of almost 600 screens ● Equivalent to a 120,000 word printed Thesaurus ● Sense identifaction for each word ● Select a word from the index using the first 2 letters and displayed on the screen is an average of 20 synonyms and associated words ● 2 way scrolling index ● Fast response ● Overlays Tasword 2 ● Words can be looked up in the Thesaurus as you are working on a text file in Tasword 2!

Also supplied is a NEW TASWORD 2 BASIC PROGRAM, specially written for Disc drive operation, offering new features such as selective catalogue, Wordcount etc. A CONVERTER program tailors your Tasword 2 code to give any combination of screen colours plus a new highly readable Tasword 2 character set.

Complete with instructions these 4 programs represent amazing value. Supplied on Disc. ONLY **£9.95**

FILE MASTER

At last! A comprehensive DataBase system that is **VERY EASY** to use. You will never forget the correct keys to press when using **FILE MASTER**, we guarantee it!

This versatile and powerful program can be used to store virtually any type of data. File Master enables you to create and maintain sophisticated data files, the type normally associated with more expensive computers. For example a file called 'DIARY' could be created. After typing the relevant data, the program can inform you which of your friends have a birthday in the next month, if the TV licence is due and the date of your next dental appointment. Records of money paid into the bank and cheques drawn could also be kept in the diary, and for an up-to-date total you simply press a key! The variety and scope of the files you can create is almost limitless and each file you create can have an infinite number of records (using segmented files).

Send for your **FILE MASTER** disc today. ONLY **£8.95**

THE ORGANISER

Features include ● Organisation of programs on any disc in any order you wish ● Makes finding programs in the directory much easier ● Selected programs load faster ● Checks for faulty sectors ● Automatic or manual Sector Repair ● Recovers erased files ... PLUS many more features to make file handling and disc 'housekeeping' so much EASIER.

This is what the experts said:-

"The ORGANISER has a very friendly user interface.... it's rare to find a package so easy to use.... brings out the best features in GDOS.... MGT should bundle it with SAM.... value for money...."
SID MARTIN in COMPUTER SHOPPER.

"Beautifully constructed and presented.... at £5 a bargain.... worth £10 or more just for the Disc Doctor aspects alone.... clearly to be recommended...."
Dr. JOHN WASE in POPULAR COMPUTING WEEKLY.

The ORGANISER is an essential program for ALL Plus D and Disciple users. Supplied on Cassette. ONLY **£5**

DISC MANAGER

THE DISC MANAGER is the most powerful program ever written for the DISCIPLE/PLUS D.

Designed to take advantage of Disc Drive ownership, the Manager keeps track of all the programs on all your discs, ● Storage of up to 27,000 records on one Disc, or 79,920 total. Random File Access. ● Name & Number Discs with fast Autonumber and user pre-defined titles features. No typing in of Data. Press a key and Discs are automatically added to appropriate catalogue. ● Fastest ever M/Code Search. 2 modes – Search and Load or Search and List all occurances, then select program to load. ● Plus many other unique features. ● Comes with 16 page manual and demonstration catalogues. ● Operates with 48K or 128K Spectrums. **Send for the DISC MANAGER today... and you'll soon wonder how you ever Managed without it!**

SPECIAL INDUG PRICE **£12.95**

NORMALLY £14.95

BETTERBYTES

SPECIAL OFFER! Send us a disc (with system), the return postage and £1 (deductable if you order FMaster) and we will send you a **USABLE FILE MASTER DEMONSTRATION** + Demo files

10 SPITTAL TERRACE · GOSFORTH
NEWCASTLE UPON TYNE NE3 1UT · TEL: (091) 285 6185

IF OUTSIDE EUROPE SEND £1.50 p&p FOR EACH ORDER

SOFTWARE THAT IS SPECIFICALLY WRITTEN FOR PLUS D & DISCIPLE

PLEASE STATE YOUR DRIVE TYPE, SIZE, ETC, INDUG MEMBERSHIP No. AND IF REQUIRED FOR DISCIPLE OR PLUS D