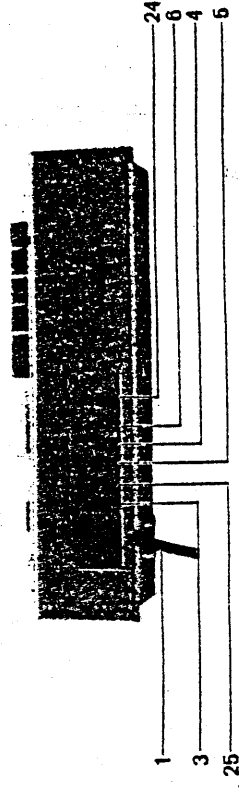
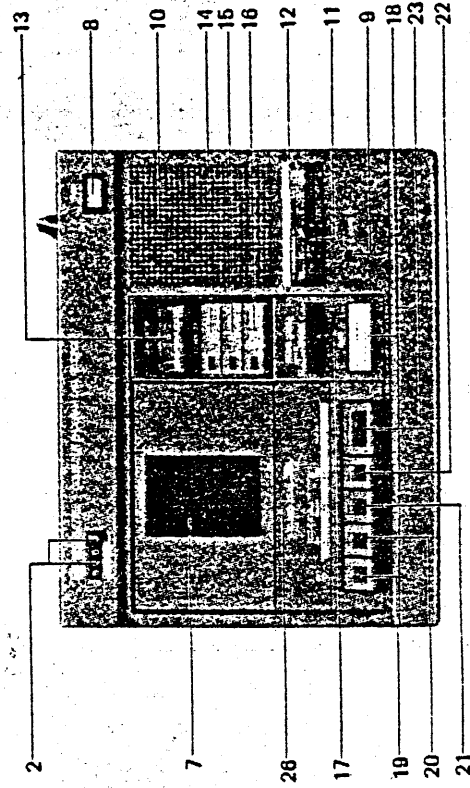


ACE USER

NEWSLETTER OF THE JUPITER ACE USERS CLUB

no. 4: winter 1983



Most of you will know of the crash of Jupiter Cantab by now. The remaining stock is being sold off (I've made a partial bid), and the liquidators are hopeful they can sell the company name. I have my doubts! About 5-6000 Aces were sold; there's about 1500 in stock (mainly French and European), and there's parts for another few hundred. In addition there are a further 800 of the Ace 4000 destined for the US market.

Now that manufacturer's support has gone, the Users Club is even more essential. As you can see, there's plenty of material being written at the moment. In fact about a quarter of no.5 is already written. A healthy state of affairs for any editor! So please do renew for 1984. The subn is being held at £7.

Peripherals: the Fuller Ace keyboard was advertised but never delivered (if you've paid, chase 'em). Stonechip's Racer is still available. Their White Echo sound amplification/tape load-save filter unit is designed, but needs a minimum order of 25 to go into production (let me know if you want one; about £25). Essex Microelectronics have produced their soundboard and have memory available (see their advert). ADS of Portsmouth have produced a Centronics interface which I'm told is good. Club member Roger Liddiard is advertising a direct connection printer in this issue. The Innovonics ZX adaptor is still available. And of course, the key contact pads kit designed by club member Lars Arell in Sweden, is available from Remsoft (as is the Innovonics adaptor). The advice regarding all these peripherals is: **BUY NOW!**

We've been having considerable tape copying problems here at Remsoft. This is a left-over from Doug Bollen's departure, with the copying gear! We've finally had to go to Downsway, but the quantities are so small, that we are well down the priority list. Thanks to all those who have waited patiently for their tapes. A 19K Monitor program is being copied at present, and there are also Database and Home Accounts programs waiting. We've also seen the first version of an interesting Spreadsheet program. We're always keen to see new utilities software, but PLEASE, no more Games!!!

The new Microkey computer is now ready, and production is commencing for a January launch. Looks most impressive. Several of the existing Ace programs will be converted, and others are being specially written. More details next issue.

Tape recorders: there's a new low-cost recorder from WH Smiths designed specifically for the Spectrum, but works well on other makes, including the Ace. It's now our main recorder, and works very well. Recommended! £39.95 from main WH Smiths branches. Ask for the CPD8300.

Finally, do please keep the articles and short items coming in: it's very much a case of sharing all available info. Also, if any one is using Forth on other low-cost micros and is interested in writing an article for a new publication here at Remsoft, do write or phone.

Next ACE USER in March 84. And dont forget to renew!

REMOSFT
Computer software
18 GEORGE STREET
BRIGHTON BN2 1RH
Tel: (0273) 602364

EPROMS FOR THE ACE

We wish to use the Ace to control laboratory instruments, but to be independent of monitors and tape-loading. This required output to the ZX printer, an integral display, RAM extension and above all, programmes on Eprom. A suitable interface was built on a double sided PCB as a direct add-on, Fig.1..

Eprom programming turned out to be remarkably simple and the programmes could be written directly in Forth with machine code embedded if necessary. All that is required is to relocate a block of RAM where the ROM will be eventually and to write the programmes within it, by-passing the on-board RAM. In our system we used a 4K RAM extension normally run at 16-20K with possible Eproms at higher 4K slots. The RAM extension could be relocated at any of these slots.

Eprom programming

- 1) Relocate the 4K RAM
- 2) Top RAM address 15436 1 Resets RAMTOP.
- 3) Bottom RAM address 15415 1 Sets HERE at the base of the RAM extension and bypasses the onboard RAM. This gives an ERROR 2 report, but relocates the stack above HERE and allows normal programming. We do not know whether or not other machines would abort at this stage!
- 4) Write the programme by normal Forth procedures. Edit and redefine may be used with considerable freedom.
- 5) Determine the linkage address of the last word by 15436 @ . and find the new value for HERE.
- 6) Transfer the programme to Eprom via the programmer, Fig.2.
- 7) Relocate the RAM at 16-20K and insert the Eprom.
- 8) Do Top linkage address 15436 1 i.e. link the new vocabulary to Forth. This step could be omitted if a small modification was made to Ace ROM. The new words can be vlisted, listed, edited and used to build further words.

Extending the programme Partially filled ROMs can be extended without erasure by copying back the programme onto relocated RAM and continuing to programme as in 4).

Programming from Tape Tape programmes can be loaded into relocated RAM and then Eprom, but any variables must be allotted to fixed addresses in RAM. As an example it was easy to allot an input buffer for the ZX printer whilst keeping the running programme and the allotting procedure in Eprom.

Summary The Ace is probably the only microcomputer for which Eproms can be written in its own language, with the great advantage for us that this language, Forth, is the ideal one for instrument control. Extension of the Ace with 16K RAM & 32K ROM would be a very simple procedure which would give a machine of considerable sophistication.

A.J. and P.D. Lawrence
Department of Cell Biology
University of Glasgow
Glasgow G12 8QQ

Fig 1 Ace Add-on for Eprom expansion and instrument control.

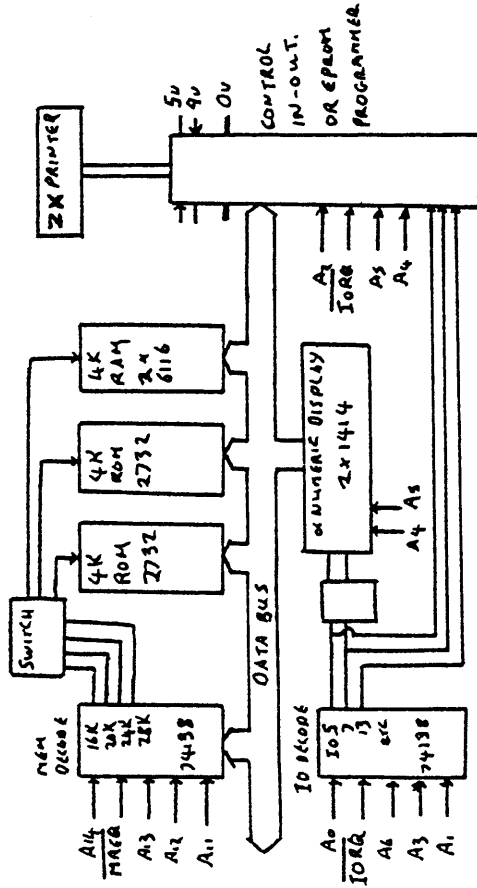
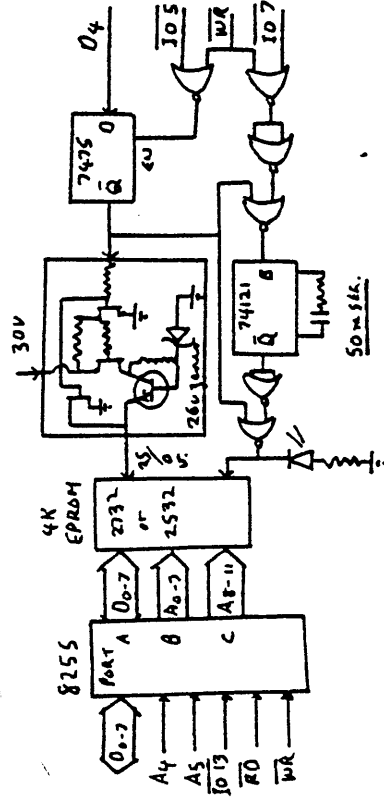


Fig 2 Eprom programmer for the Ace.



An Eprom was written for location at 16-20K to contain words to run the Eprom programmer.

: inen 73 5 out ; Sets the programmer to read and displays I.
: outen 80 5 out ; Sets the programmer to write and displays P.
n : e@ inen 256 /mod 144 61 out 45 out 29 out 13 in ; Reads one byte from the Eprom.
n1 n2 : e1 outen 256 /mod 128 61 out 45 out 29 out 13 out 10 0 do loop
0 7 out 500 0 do loop; Writes a byte to an address in the Eprom.
n1 n2 : pgm do i c@ i e1 loop; Transfers blocks of data to the Eprom.
n1 n2 : everify do i c@ i e@ = 0= if i . then loop ; Verifies programming.
: check 4096 0 do i c@ i e@ emit loop ; Types the contents of the Eprom. Very useful
n1 n2 : copy dup 20480 do i e@ i c1 loop 24575 15384 i swap 15436 i
15415 i ; Copies the programmed portion of an Eprom back to relocated RAM
and sets the conditions to continue programming. Only works for 20-24K in this
form.

DEPARTMENT OF CELL BIOLOGY

THE UNIVERSITY, GLASGOW, G12 8QQ. telephone: 041 339 8855 Ext. 7147.
REF: K/5

11th November 1983

Mr. J. Noyce
Remsoft
18 George Street
Brighton BN2 1RH

Dear Mr. Noyce,

Following our telephone conversation I am submitting
the enclosed article for the Ace User News Letter. I
think the work is of some general interest and I would
welcome enquiries about the details of the circuits. As
you will see these are block diagrams, but there are no
hidden catches. We have double sided PCBs for the
circuits and they all fit together with standard edge
connectors.

Yours sincerely,

Tony Lawrence

Dr. A.J. Lawrence

SOUND GENERATOR BOARD & JOYSTICK PORT

LEX VAN SONDEREN

Here is a design for a sound generator board and joystick port I made myself for the Ace.

Components are:

	Qty
AY-3-8910 PSG	1
79 LS 02	1
79 LS 32	1
LM 386	1
I/O SOCKET to fit ATARI-type joysticks	1
3.5 mm SOCKET (Base and switch)	1
LS. 8 Ohm	1
Resistor 2.7 Ohm	1
" 470 Ohm	2
" 1k Ohm	11
Variable resistor 1k Ohm	1
Capacitor 47uF	1
" 100uF	2
" 0.1uF	5
" 330pF	1
23-Way Edge Connector	1

In Holland this cost me about £18- I think it will cost about £12-15 in the UK.

The programmable sound generator contains 16 registers, and is addressed with the commands

221 OUT
223 OUT
221 IN

Entering N 221 OUT (N=0 to 15) selects the current register, and N 223 OUT puts value N in register last pointed to by N 221 OUT.

I use : REG 221 OUT ;
: SET 223 OUT ;
: REG? REG 221 IN ;

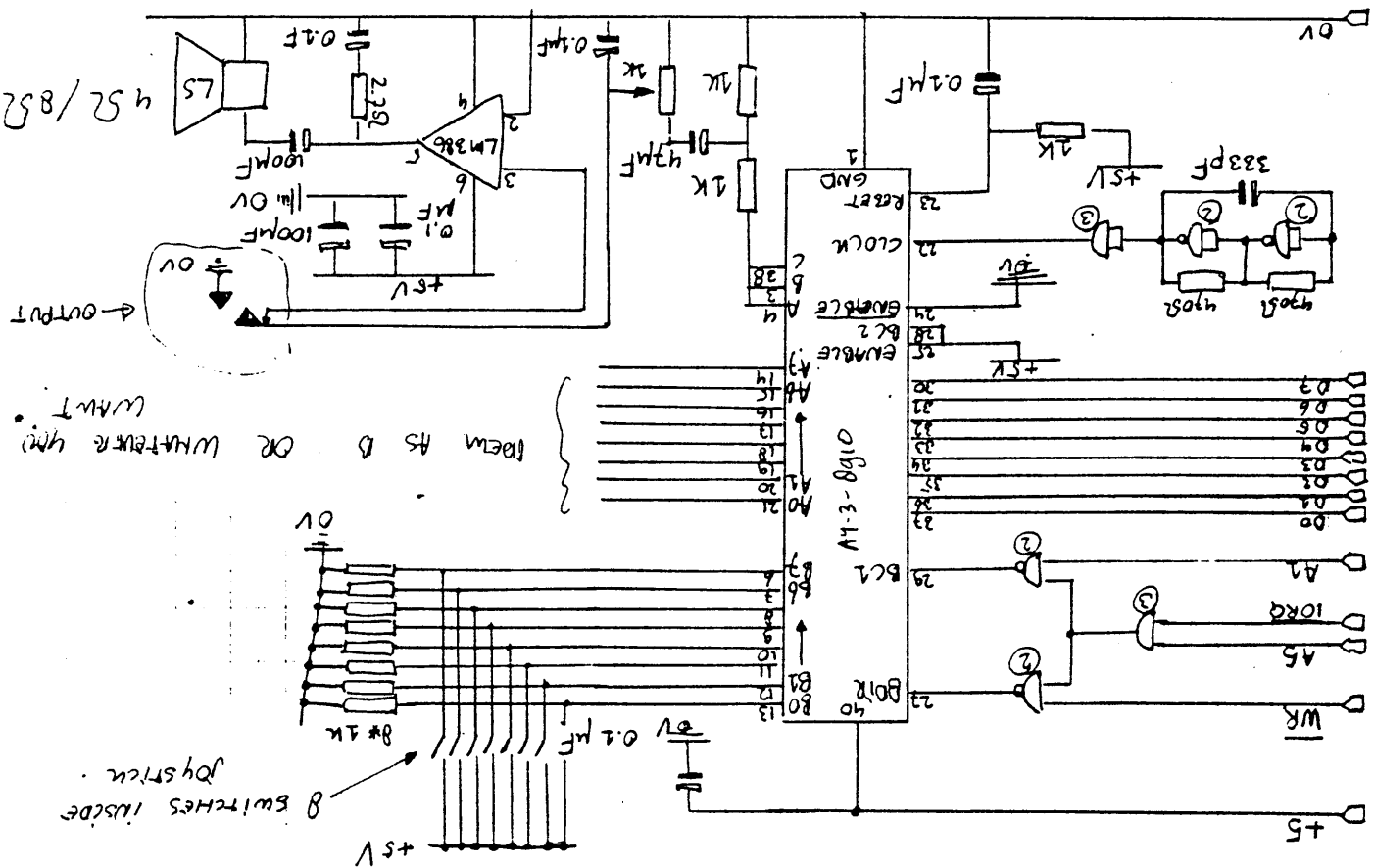
EG. 6 REG points to register 6, and

25 SET puts 25 into register 6.

6 REG? returns the current value of register 6

7

② : 74LS02
③ : 74LS32



The registers have the following uses:

Register #	Use	Val
0 & 1	Low and High bytes of channel A pitch (same value as ACE beeps)	0-255 0-31
2 & 3	Low and High bytes of channel B pitch (same value as ACE beeps)	0-255 0-31
4 & 5	Low and High bytes of channel C pitch (same value as ACE beeps)	0-255 0-31
6	Noise period on all channels	0-31
7	On-Off selection (see below)	
8	Volume of channel A	0-15
9	Volume of channel B	0-15
10	Volume of channel C	0-15
11 & 12	Low and High bytes of Envelope Duration	0-255 0-255
13	Envelope Shape (see below)	
14	I/O port A data register	
15	I/O port B data register	

Register 7:

Decimal	128	64	32	16	8	4	2	1
Bit	7	6	5	4	3	2	1	0
Use	B I/O	A I/O	C Noise	B Noise	A Noise	C Tone	B Tone	A Tone

When a bit of 0-5 is set (1), the channel is switched OFF, when reset (0) it is switched ON, so any combination of tone and noise channels is possible.
For bits 6 and 7, when SET the port is an OUTPUT, and when RESET it is an INPUT.

Register 13:

(Envelope controls the volume only)

Contents:

Shape

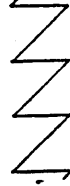


0-3

4-7



8



9



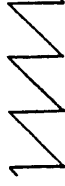
10



11



12



13



14



15



Envelope control is switched on by setting bit 4 of the volume register of the required channel. EG.

Register 8-10 Bit 7 6 5 4 3 2 1 0

not in use | envelope control enable

I wish you a lot of success building it.
When you think it works, enter the following:

```
: REG 221 OUT ;
: SET 223 OUT ;
: REG? 221 OUT 221 IN ;
```

Try the following:

```
0 REG 200 SET
7 REG 254 SET
8 REG 8 SET      It should produce a continuous tone
```

If it fails, enter:

```
0 REG? .      It should print 200
7 REG? .      It should print 254
8 REG? .      It should print 8
```

If not, your PSG is not properly connected. When it is, and if it still fails, the amplifier part or the clock part is not properly connected.

To put all sound off: 7 REG 255 SET

Some Examples:

```
6 REG 8 SET
7 REG 247 SET
8 REG 16 SET
12 REG 20 SET
13 REG 1 SET
(Shot)
```

```
6 REG 28 SET
7 REG 247 SET
8 REG 16 SET
12 REG 20 SET
13 REG 1 SET
(Explode)
```

```
0 REG 200 SET
7 REG 254 SET
8 REG 16 SET
12 REG 20 SET
13 REG 1 SET
(Ping)
```

```
6 REG 16 SET
7 REG 247 SET
8 REG 16 SET
12 REG 10 SET
13 REG 4 SET
(Snap as in Pacman)
```

```
6 REG 20 SET
7 REG 247 SET
8 REG 16 SET
12 REG 50 SET
13 REG 10 SET
(Sea-Waves)
```

```
0 REG 200 SET
2 REG 201 SET
4 REG 100 SET
7 REG 248 SET
8 REG 16 SET
9 REG 16 SET
10 REG 16 SET
12 REG 20 SET
13 REG 8 SET
(Twangy Piano)
```



ESSEX MICRO ELECTRONICS

JUPITER ACE ACCESSORIES

SOUND & I/O BOARD:

AVAILABLE NOW FOR JUST £30.

Three channel sound board with a 16 bit Input/Output port. Sound output is via. an internal loudspeaker or an external jack socket. Complete with 24 page instruction manual which also gives examples of software. We are also including details on how to make a centronics printer port using the soundboard.

FULLY CABED AND TESTED RAMPACKS.

16k Rampack	£25
32k Rampack	£35
16k to 32k upgrade	£13

EDGE CONNECTOR FOR ACE EXPANSION BUS & SOUNDBOARD
I/O PORT: £4

SPECIAL OFFER:

SOUNDBOARD + 32k RAMPACK £59 (inc. P&P).

Send an SAE for details on the above products.

Please add £1 per order to cover P&P
(except where stated).

MAIL ORDER ONLY.

CHEQUES etc. to:

ESSEX MICRO ELECTRONICS,
4, HATCH ROAD,
BRENTWOOD,
ESSEX.
CM15 9PX

INTERRUPTS ON THE ACE

Edward Hattersley

Interrupts are a way of getting the Ace to do something else at regular intervals of 1/50 second, eg. update the clock (which it does automatically), output a sound, etc., all without disrupting the main flow of the program running. Most people will probably find them most useful for programming sound into games, so that sounds may be produced without stopping the game. This is possible using other techniques, but it is a lot easier with interrupts. Another possibility is a clock in the top-left of the screen, again running independently of everything else (It should be noted, however; that the interrupts are disabled when tape input or output is occurring, to preserve the tape timings. Other operations may also disable interrupts, but this seems unlikely.).

So, how to do it? The ACE usually runs in Interrupt Mode 1, which, when an interrupt occurs, will transfer control to location 38 Hex. It is vital that any interrupt routine either exits to or is accessed after this routine. The way to access another address is to use Interrupt Mode 2, which takes the high byte of the next address to be executed from the interrupt register, and the low byte from the data bus. Due to the circuit of the ACE, the data read off the data bus will always be 20 Hex, so we can use any ram with address XX20 to point to our routine. The suggested method is as follows - work out what it is doing, and you will be able to adapt it as necessary to meet your own requirements:

Set RAMTOP to (say) \$7F20 (32544)

Store \$ 7F22 at \$7F20

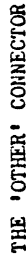
(NB. \$ means the numbers are
base sixteen)

Write code from \$7F22 onwards with either;

RST 38 (FF) as the first instruction, and ending with

RFT (C9)

or Fnd routine with a JP \$38 (C3 mn mn) instruction

$$\begin{array}{c} \uparrow \\ n_0 \\ n_1 + 1 \\ n_2 + 1 \end{array}$$


PIN 4 - 25	Pins 11, 12, 13, 14
PIN 5 - 25	are data out from
PIN 12 - 25	the video ram.
PIN 14 - 25	Pins 1, 2, 3, 4, 5, 6,
PIN 13 - 26	7, 15, 16, 17 are
PIN 11 - 26	address lines into
PIN 14 - 26	the video ram.
PIN 1 - 26	PIN 10 is write-
PIN 15 - 26	enable.
N.C.	Modulator video
EARTH	in could be
	connected to a
	monitor directly.

[illegible]

13

To enable interrupts, call the following M/C routine:

```
3E 7F  LDA 7F
ED 47  LDI,A
ED 5E  IM 2
FD E9  JP (IY)
```

To disable interrupts (this may disable the clock):

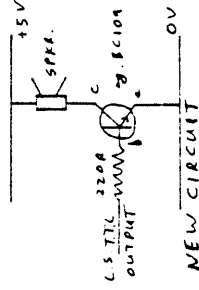
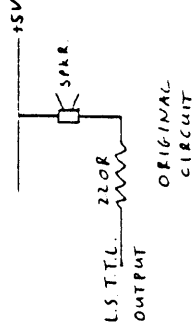
```
142 CALL (decimal)
```

This gives you about 220 bytes to write your interrupt routine. If this isn't enough, you can set RAMTOP to any address that ends in 20, eg. 7020 Hex. Load the I register with the top byte of the address (eg. 70 Hex) using the M/C routine above. Note that the address at X20 can call anywhere in memory - the byte just after the pointer is recommended, but feel free.

SOUND ON THE ACE

Edward Hattersley

Yes, you too can have AUDIBLE sound from your ACE! While you are building the sound-board featured elsewhere in this issue, you can use this circuit to boost the sound level from the ACE's built-in speaker. It still isn't deafening, but it IS louder. The modification requires soldering to the ACF circuit board, but I suspect most of you have had the thing to bits already anyway, so that shouldn't be too much of a problem. First, you need a transistor. Anything will do, so long as it works - the power output is so low that it won't burn out anything. I would suggest a BC109 as being very easily available. Then, just follow the diagrams-



Some notes:

Fitting a full-size Keyboard - There are two ways to fit a full-size keyboard. One is to solder lots of wires to the keyboard matrix, which I would not personally recommend. The other is to build a circuit which simulates the real keyboard, using a circuit that fits on the connector on the back. Such a circuit has been designed by Doug Bollen, and was published in ETI (November issue, Vol. 12, No. 11, page 20). This will also allow the use of joysticks as simulated key pressers, thus allowing the use of joysticks on any game which uses the keyboard, regardless of the keys used. Extra (very) cheap ram - Hands up anyone who would like an extra 16K of ram for under £10? Well, Wireless World published a circuit a short time ago (May 1983, page 64) which uses some very neat circuit tricks to get a very cheap circuit (I was very sceptical, until it worked first time!). Some hints - It is a DIY project, so if you don't feel confident, get someone else to build it for you. The construction technique that I used is a rather unusual one, but it works very well: the chips are placed upside-down on the copper side of a blank PCB, and the ground pins bent down to reach the copper, and soldered on. The pins (which are sticking up in the air) are then wire-wrapped together. Yes, I know, but it works very well, and I haven't blown any chips yet. I bought the components from Rapid Electronics, who are very cheap. DON'T be tempted to alter any resistor values - the circuit uses RC delays to achieve the necessary timing, and any messing around will cause the circuit to fail.

Z80 Machine code programming books - I would recommend Rodney Zak's 'Programming the Z80'. It costs about £14, but it's worth it. The chip in the ACE (NEC D780C-1) is effectively, if not actually, identical to the Z80, so all hardware and software written for the Z80 will work.

JUPITER ACE LINE PRINTER

CMOVE & FILL

G.R. YORKE

Doug Bollen's version of CMOVE (Ace User No3), while being short , does have 2 drawbacks.

The forth-79 CMOVE (addr1 addr2 n -) moves n bytes from addr1 to addr2 & does nothing if n is zero or negative. With Doug's listing if n is zero or negative the program is stopped with an error code.

The second problem occurs if the move involves a small 'shift' so that the Two areas overlap. In this case , if the direction of movement is up, the data is corrupted (Try moving a block one byte up the memory) .

I enclose the following listing which, although longer, overcomes both problems . If given negative or zero nothing is done apart from removing the 3 values from the stack . It also allows small moves by detecting which way the move is to be made and transfers the data starting at the far end working back to the start, which avoids overwriting and corruption.

There is also a listing for my version of FILL which also ignores negative or zero values of n and allows more than 255 bytes to be filled, a problem with some other versions .

CMOVE (addr1, addr2, n -)

DF D5 DF D5 DF D5 EI DI A2 ED 52 F5 19 CI CB 79 CI
20 OC 78 BI 28 IB CB 78 20 I7 ED B0 I8 I3 78 BI 28
OF CB 78 20 OB 09 7B 8I 5F 7A 80 57 2B IB ED B8 FD
E9

FILL (addr, n, byte, -)

DF 7B DF 42 4B DF EB CB 78 20 OB 5F 03 OB 78 BI 28
04 73 23 I8 F7 FD E9

N.B. These are all in Hex. and read from left to right

Jupiter's Forth, like most of its relatives, provides a reasonable selection of program structuring statements. It has three loop control word sets:-

```
DO LOOP
BEGIN UNTIL
BEGIN WHILE REPEAT
```

It does however, lack (again in common with most other versions of Forth) any selection control other than IF ELSE THEN. The more versatile CASE structure is missing. One thing the Ace lacks which other FORTH implementations often contain is easy access to the primitive functions out of which these control structures are made, and which can be used to generate new structures, such as CASE. These were presumably omitted to ease the writing of LIST and EDIT routines.

Here are a set of routines which, by misusing the COMPILER directive, allow one to write your programs using the CASE statement. It does have the drawback that if you list or edit the routines afterwards you'll discover that the CASE structure has disappeared, replaced by nested IF... THEN structures and a number of extra words. This result can be edited and does not suffer if words in front of it are redefined, and has the advantage that CASE structures, if understood, are easier to write than the equivalent nested IF structures, and these routines will make the translation accurately. CASE statements may also be nested within a word without causing problems, and any misuse is reported as an error by the FORTH interpreter.

The routines included comments to explain operation. These comments (enclosed in parentheses) need not be typed in. Anything in square brackets [] must be included, in square brackets, as shown, even though listing afterwards will show these to have disappeared, with the following word, to be replaced by a number.

```
e.g.  [FIND DROP] LITERAL
      will be replaced by the number
      2169
      if you list (or edit) the routines.
```

To use the case routines, follow this example

```
; demo
CASE
1 OF ." one"   ENDOF
2 OF ." two"   ENDOF
3 OF ." three" ENDOF
4 OF ." four"  ENDOF
." the top of stack number was bigger than four" (default)
END/CASE
;
```

The CASE structures takes the number on the top of the stack and compares it with the numbers to the left of each of the OF words in turn. The first one that matches results in the words between that OF and ENDOF being executed, and then the structure being exited. If no match is found the words left at the end (line marked (default) here) are executed. Although simple string print commands are used here any valid FORTH sequence may be used.

Only one OF branch is executed (unless no OF branch is executed, in which case the default, if present, is executed). Also the numbers matched need not be in order, and could be calculated from variables, e.g. the following is also valid:-

```
5 VARIABLE NUM
; demo 2
CASE
7 OF 10 0 DO I . LOOP ENDOF
NUM @ 5 * OF ." twenty five" ENDOF
9 OF NUM @ CASE
5 OF ." unchanged" ENDOF
6 OF ." incremented" ENDOF
DUP ."NUM =" .
ENDCASE
(*)
(*)
(*)
(*)
(*)
```

```
ENDOF
ENDCASE
;
```

```
Try 7 DEMO 2
25 DEMO 2
9 DEMO 2
8 DEMO 2
```

and see if it does what you expect. Note that CASE statements can be nested (note lines marked with (*)), and that a default line is not necessary (not included in main CASE, so 8 DEMO 2 shouldn't do anything at all). Lift DEMO 2 and see how much more complicated it is when you've only got IF...ELSE...THEN for structuring. Its easy to get it wrong if you try to write something like that yourself.

When you've finished, and you no longer need the compiler word, you can use 'redefine' to remove them and save space. The words you've defined don't need them any more.

I trust Jupiter Cqntab don't feel I've taken too many liberties with the facilities they've provided. I'm sure it's not really how they'd intended COMPILER to be used:

(Creation of CASE statements)

```
DECIMAL
0 VARIABLE COUNT
```

```
: 0BRANCH (defines a word to implement a branch over following
words if top of stack is zero. Do not use outside
a definition, or if you don't understand it.)
4739 , HERE 0 ,
;
```

```
: BRANCH (as 0BRANCH, except unconditional branch)
4721 , HERE 0 ,
;
```

```
: BACK (obliterates COMPILER reference in compiled code.
Misuse is disastrous).
HERE 2-
15415 ;
;
```

```

Ø COMPILER CASE
COUNT @
    (set up CASE structure)
    (save count in case this is a nested
CASE statement)
Ø COUNT : BACK
4
RUNS> ;
    (zero count, obliterate COMPILER entry)
    (number on stack for checking usage)
    (meaningless, because of BACK, but
COMPILER requires it.)

Ø COMPILER OF
BACK COUNT @ 1+ COUNT :
4 = IF
    [FIND OVER] LITERAL ,
    [FIND = ] LITERAL ,
    ØBRANCH
    [FIND DROP] LITERAL ,
5
ELSE 77
THEN
RUNS> ;
    (selection word)
    (obliterate compiler reference)
    (usage OK)
    (put OVER in definition)

    (number on stack for checking usage)
    (force error for mismatching)

Ø COMPILER ENDOF
BACK
5 = IF
    BRANCH
    SWAP HERE OVER -
    1- SWAP :
4
ELSE 77
THEN
RUNS> ;
    (terminates selection)

    (set up parameter field for)
    (previous OF ØBRANCH)

Ø COMPILER ENDCASE
BACK
4 = IF
    [FIND DROP] LITERAL ,
    COUNT @ ?DUP
    IF
        Ø DO
        2 [FIND THEN] LITERAL EXECUTE
        LOOP
    THEN
    COUNT :
    ELSE 77
    THEN
    RUNS> ;
    (terminates CASE structure)

    (replace COUNT saved in CASE)

```

? PLOT

M.FINLEY

The Ace plot command lacks the ability to find out whether a particular point has been plotted on or not. Despite the high speed of FORTH, if you attempt to write a FORTH word to find out the "colour" of a pixel you'll find its unacceptability slow for moving graphic games. (I found that my attempt was anyway.)

To correct this lack, I've written the attached machine code routine which takes two numbers from the stack (x,y as in PLOT) and returns a single number; either zero if the pixel is black, or non-zero if white (actually, the number returned gives the bit which determines the state of that particular pixel in the character displayed).

The routine is position independent, so can be entered and used as described in the Users Manual on page 147.

While I'm writing may I make a plea that if you do publish machine code routines, you publish the assembler source as well as the machine code bytes; it's much more readable that way, and enables people to detect typographical errors in the hex codes.

?PLOT

Assembler	Mnemonic	Machine Code (hex)
RST	24	DF
PUSH	DE	D5
LD	C, E	
SRL	C	CB 39
LD	A, 22	3E 16
SUB	C	91
RST	24	DF
PUSH	DE	D5
SRL	E	CB 3B
CALL	0B28H	CD 28 0B
LD	E, (HL)	5E
XOR	A	AF
BIT	7, E	CB 7B
JR	Z, LL1 if using inverse charac-	28 02
LD	A, 127 ter prepare to invert	3E 7F
XOR	E lower bits of character	AB
POP	in A.	D1
POP		C1
BIT	0, C	CB 41
JR	Z, LL2A	28 0A
LD	L, 1	2E 01
BIT	0E	CB 43
JR	NZ, LL2	20 02
LD	L, 2	2E 02
LD	LL3	18 08
JR	L, 4	2E 04
LD	0, E	CB 43
BIT	NZ, LL3	20 02
JR	L, 128	2E 80
LD		

this code uses the bottom bits of C & E to determine which bit should be tested in the character.

	Assembler	Mnemonic	Machine Code(hex)
LL3	AND	L	A5
	LD	E,A	5F
	LD	D,0	16 00
	RST	16	D7
	JP	and puts it on the data stack.	FD E9

Notes that this word will return zero if the pixel is inset (i.e. black), and the bit in the character if it is set. It would be easy to modify it to return 1 if set, but it was left this way as it gives more information. This extra information may be useful, and is still treated as 'TRUE' by the forth interpreter, so nothing is lost.

BYTE-WATCHING IN MALLORCA

I am interested that you have at least one serving member of the Royal Navy in the Club. I am a retired RN officer, and was of the marine and air engineering specialisations, and was also a pilot, and if any RN members are serving in ships which visit Mallorca, perhaps they could drop me a line in time for us to be able to meet. Naturally the same applies to other Club members if they are going to visit the island.

without memory expansion, Byte-watching is vital, and while experimenting I find this work very useful:

```
: L? DUP 1- C@ SWAP 5 - @ + . ;
```

Used in the entry "FIND "NAMEOFWORD" L? " it prints out the length of your word, name and all. Of course it won't work for the last word in your dictionary as the length field had not been set up. In this case I use the entry " : X ; FIND "NAMEOFWORD" L? FORGET X" to rectify the lack. L? is 31 bytes long.

Either there is something wrong with my Ace, or there are a couple of typing errors at the top of p20 of ACE USER No.2. I get the value 233 for the byte at 1D7A, which doesn't seem to tie up with ASCII 32 being a blank. Should it not be 1D78, which is 0? Also ASCII 63 - 94 appear to be 6 bytes each, not 8. Finally ASCII 127 would appear to be the only character stored as its full 8 bytes.

P.A.L. Watson, Ca'n Reus, Pollensa, Mallorca, Spain.



MicroProcessor Engineering Ltd

21 Hanley Road Shirley
Southampton SO1 5AP
Tel: 0703 775482

JET-DISC add-on disc drive system

JET-DISC --- universal disc upgrade extends the JUPITER ACE

Add the power of discs to your Jupiter Ace. The JET-DISC universal floppy disc controller can be configured to run on nearly all home computers!

Remember, peripherals often cost more than the computer itself. With the JET-DISC, if you upgrade to a more powerful computer, or need discs on a second computer, you will not have to sell the JET-DISC, just obtain a new configuration ROM and connecting cable, and connect JET-DISC to your new computer!

JET-DISC is not just universal, it is in itself powerful and upgradeable. JET-DISC consists of three system components.

The disc controller system uses one the most recent floppy disc controller chips which is capable of handling four disc drives. These drives can be of any size 3.5", 5" or 8" - single/double density - single/double sided - and can be intermixed. The most important and difficult part of any disc controller is the read data separator system. The JET-DISC's DIGITAL PHASE-LOCK LOOP needs no setting up, and will never drift - fast reliable disc reads every time. The board can be set up to be driven by any micro. JET-DISC's on board ROM and RAM sockets take the driver software and disc buffers for the computer to which JET-DISC is attached.

The interconnect cable and ROM configure JET-DISC for the computer in use.

The system box contains the power supply unit, one or two 3" disc drives, and the JET-DISC disc controller board. The power supplies in many home computers are often running too hot as it is. The JET-DISC disc pack contains its own power supply and thus prevents any overloading of the computer.

JET-DISC's internal disc drives are 3" drives featuring hard cassette covers. As the disc is completely sealed until it has been inserted into the drive, it is safe for use in both home and industrial environments.

JET-DISC is fast, rugged, and cheap. JET-DISC's controller board and interface cable can be bought separately.

JET-DISC --- technical specification

Disc controller - TMS 9909 - supports 4 drives of any size, software selectable single or double density, software selectable single or double sided, software selectable 3", 5" or 8" size, Any mixture of drive sizes may be used.

Computer Interface - The computer interface allows the disc controller to be configured to the computer hardware. The 9909 controller can be I/O mapped for 8080/780 type processors or memory mapped for 6502/6809 type processors. The interface can accept either processor type's control signals.

Memory facilities - two sockets are provided for the configuration PROM and the optional disc buffer RAM. Sockets take 2716 EPROMs or 6116 RAMs. If the facilities are not required they may be removed from the memory map. This additional memory may be at any 4k memory block. If you have a computer with limited memory facilities JET-DISC will not use any of the computer's own memory - everything you need is on the JET-DISC.

Memory & I/O override - JET-DISC can be set up to override the internal PROM or RAM of your computer if your computer allows this.

Prices - JET-DISC is available in several configurations.

With Jupiter Ace software & interface, power supply, case, one 3" 180k capacity drive, - complete with FORTH-DOS disc handling software ... £260+VAT
Additional 180k drive ... £115+VAT

Disc controller board complete and tested with interface cable and FORTH-DOS software as above ... £95+VAT



MicroProcessor Engineering Ltd

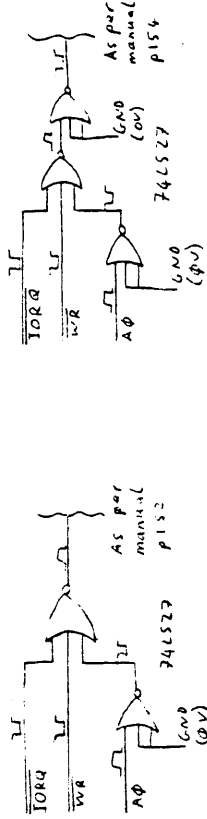
21 Hanley Road Shirley
Southampton SO1 5AP
Tel: 0703 775482

CHARACTERS AND YET MORE 1/0

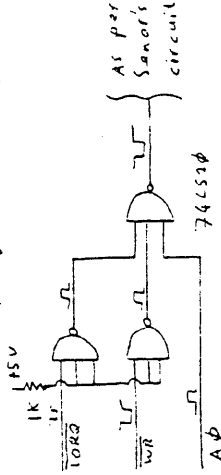
I would like to take the opportunity to pass on a few hints, tips, ideas etc.

1) The April and May issues of "Your Computer" have carried articles on character-forming and machine code scrolling/filling routines respectively. According to Roger Liddiard (the former article) ASCII characters 32-62 and 95-126 occupy seven bytes each in the ROM, 63-94 occupy 6 bytes and character 127 (normally "Delete" in ASCII but used for the copyright symbol on the ACE) actually occupies 8 bytes. Interested parties are referred to these articles. Also TJ's workshop in Personal Computer World occasionally contains tit-bits of information on the ACE. A recent one contained "LISTV" which only lists user defined words in the dictionary.

2) As regards the I/O port circuits on pages 152,154 of the manual, the Senor is correct about the decoding of the port address (ACE USER No.2, page 9). An alternative solution which uses the IC's in the manual, however, is as follows:



The alteration is simply to use one of the spare NOR gates to invert the A0 line and take the output th where A1 is in the circuit in the manual (instead of A1!) For those using the Senor's circuit, I would suggest a small modification to avoid loading the edge-connector outputs which may not be buffered (They're not. Ed) :



The alteration here is to tie the spare inputs of the NAND gates used as inverters to +5V via a 1k Ohm resistor. This means that the outputs at the edge connector only see one LS load each (instead of 3 in the Senor's circuit). The circuit as shown can also be used for the port on page 154 in the manual by taking the input from RD instead of WR and connecting the output directly to the 74LS368.

NB. The circuits as shown only check one address line (A0) and therefore cannot be used with other circuits using the odd-numbered ports, as they will all respond to I/O requests to any odd-numbered ports.

Finally, a question of my own - why are there four copies of the dictionary, stacks and system variables? Is the address not being fully decoded (ie. bits A10, A11 being ignored)? If this is the case, then 3K of possible expansion area is unable to be used without internal modifications. Even with the efficiency of FORTH it seems to be a waste.

I hope some of this is interesting/useful to you,

A.J.Paterson

Yes, the duplication is due to incomplete decoding of the address lines, as you say. This is done to simplify system design (ie. make it cheaper). A note to those of you who have 48K of ram, the house lights controller and whatever else connected to the back (Jodrell Bank?), it would be a good idea to buffer all the outputs to prevent excessive loading of the Z80A lines, which could lead to logic-level problems, or even damaging the chip.

Some information of my own on the confusion about the number of I/O ports: the Z80 has 256 'official' I/O ports, but it is possible in certain circumstances to have 65536 individually addressable ports. To clarify; Whenever the Z80 does an I/O operation, it puts the value in the C register on the lower eight bits of the address bus, and the B register on the upper eight bits. This works fine for FORTH I/O, because the B register isn't being used for anything else, but in machine code it can cause problems... There is a command (with some other related commands) called INIR (Input and Increment with repeat), which transfers B bytes of data from I/O port C to memory starting at HL (B,C,HL are register contents). Obviously this is going to cause some confusion to any I/O port that reads the top eight address lines, as the B register is incremented to keep track of how many bytes have been transferred. Instant Chaos! So, BE WARNED!!!

EH

Here, for what it is worth, is my own version of DUMP-
you are welcome to use it if you want:

```
: dump
cr dup 5 mod ?dup
if
  5 * 5 + spaces
then
dup 100 + swap
do
  i 5 mod 0 =
  if
    cr i 0 # # # # # # # type t
  then
    space i c@ dup dup 13 =
    if
      drop space
    else
      emit
    then
      0 # # # # # # # type
    loop
  cr
;
;
```

(addr-), displaying contents of 100 bytes after (and
including) addr., alpha and numeric form,

Dave Crook.

Welcome to THE CLUB

It's a shame about Jupiter Cantab because I've just
bought an ACE for £50!
Still, I'm hungry for information, eg. What is the 2nd.
edge connector for, or even connected to? Can I hack the
machine to give a composite video output?
I hoped the User's Group may have some technical notes
on the machine,

Iestyn Walters.

PS. I've got to admit it, it's far more fun than my
Newbrain! (The Newbrain looks like having its 8K
BASIC replaced by FORTH Ace-style).

See elsewhere for information on the 2nd. edge connector,
including composite video. For all the technical notes
available, see the last two newsletters (including a
circuit diagram).

KEYBOARD WARNING

To anybody fitting a full size keyboard to the ACE, a word of warning. I fitted a non-encoded keyboard to the ACE which has an 8 x 5 matrix arrangement and found that I needed to shorten the lead wires from their keyboard to mine and the ACE keyboard does not like much capacitance across the matrix. Using small 15 core screened flexible cable I found that 1 metre would cause malfunction or non-function but that it would tolerate up to half-metre. My arrangement is simply to connect my key matrix in parallel with theirs so I can use either keyboard at the same time.

Don Horn, Surrey.

JOYSTICKS, SYNTHESISERS, ETC.

Nearly all Remsoft's and Jupiter's games are useless with a joystick. Tape 9 is O.K., and 'Aliens' with its multikeypress is acceptable but slow to respond. You should take this into account when selecting future programs. Cambridge Computing's actual joystick (not the interface) is also useless as it is of the 'floppy-stick-with-no-self-centring' variety. Good joysticks are difficult to find, but a revelation to use.

This Winter I am hoping to design and possibly market a computer music synthesiser which can be driven by any Z80 micro. Between us, Garry and I have developed a good solution to the 'human interface', and the next stage is the multi-use output interface. It does look as if the trend is designing hardware and writing software for CPU's instead of computers, as this gives a potentially bigger market. Perhaps you ought to start the first National Z80 club! (or 6502/6809). Enough of this rambling, All the best, Doug Bollen.

PACER

Can you write me on the PACER-32K-memory-unit, which I own, is filled with the 4116-chips, which work better with 12V ??? (See Ace User).

Again, thank you very much for all and I am glad to be a member of the Jupiter Ace Club.

K.Toet, Gravenhage.

Answer: Yes!!

J.N.

SOUND OUTPUT

Your doing a great job down there in Brighton.

I would be very grateful if you could give me some more useful addresses for ROM routines and a machine routine for producing random numbers. It would be nice if you could market the tapes of some of the smaller software companies.

I hope the hardware pleas to Jupiter Cantab works because at the moment the Ace has the fewest number of hardware add-ons of the computers under £200.

Unlike many people, I have had absolutely no problems with loading and saving programs (I use my stack system). I am also the proud owner of Callisto and Ganymede, assembler and dissassembler. Although I managed quite easily to edit the loading routine, I would be very pleased if someone could explain how the program self starts.

So many people are proudly boasting full sound with their programs, why can't we (with a bit of modification) use Spectrum Voice Synthesis program on page 109 of the June issue of Your Computer. Could someone explain why this program won't work.

Please could someone work out how to get sound through the T.V. speaker.

I am about to buy a joystick and interface from Cambridge Computing and so I will write in and tell you how it goes. Keep up the good work!

R. Maycell, Kent.

SMALLER WORDS!

I am writing for information about several things; First, could you send me a list of all the club members living in the Exeter area. I am sure there must be some but I haven't met any! I'm beginning to get lonely down here (the only ACE I've seen is mine!).

In issue 2 of ACE User Doug Bollen said that the William Stuart Chatterbox is usable with the ACE. Can I have more information on this please?

Martyn Sudworth in his letter in issue 3 mentioned the use of more and smaller words in FORTH dictionary files. The importance of this cannot be overstressed not only from the memory point of view (to do with editing) but also Forth is meant to be composed of subroutines which are easily transportable to other dictionary files.

An example of this is the Big Letters listing in issue 2. When giving memory sizes of Remsoft tapes could you always give the exact amount of memory used by the programs because this is more useful if one wants to have more than one program in memory at once. Could you also give the prices of books mentioned in Biblioforth. Do you know where I can get Reversi/Othello for the Ace? Where also can I get a ROM listing and analysis similar to that for the Sinclair machines?

I have bought the E.M.E. Soundboard and will be able to do a review of this once I have bought a KAM expansion and when I receive the rest of the manual (I ordered in mid-August so they sent it with only most of the manual to keep me happy)-the sections on applications of the sound and I/O port are yet to be delivered.

M. Webb, 10 Pamela Road, Exeter, Devon. EX1 2UF

TAPE LOAD & MERGE

Tape No. 11 and 14 did load well at the beginning of each session, but with the power connected to the micro for an hour or more resulted in bad loading. Now I have replaced the internal heat sink with an external one, and all loading troubles are gone. Thus, in my case, a combination of high noise level on tapes and high temperature in the micro caused loading problems.

And now another problem, not yet solved: It would be an advantage to have complete listings of the programs. I am trying to combine program 11 and 14 in order to get the hires graphs plotted on paper. However, program 14 has user defined characters, and program 11 is not able to print them. Parts of both programs are unlistable (machine coded), which causes extra work in order to get the programs to co-operate. A hint to David Atkins, defining inverse trig. functions: Modify word SIN, page 92 in the manual. Inverse trigs can also be expressed as series which you can find in a collection of math. formulas.

Lars Arell, Sweden.

ULLNUM & BROTHER TYPEWRITERS

I am the happy owner of a Jupiter Ace and therefore very interested in the User club. My Jupiter has a 16K RAM-pack and is also connected to an Ultimium computer interface.

I am thinking of using the memory in the Ultimium as a bank with the memory paged in 32K packs which I can switch in and out by software. The 16K is "on" all the time. There are some small problems about the RAM in the Ultimium, but I think they will be solved within a month. Now it is a port card and a printer that is on my 'wishing list'. The printer has to be good and not too high priced, so I have decided to buy a Brother typewriter which should arrive shortly here to me in Norway; it is priced around £200 which is very cheap here.

I am now typing on the smaller Brother of that one I have decided to buy. However, there is less written about the Jupiter today, and I am suffering from very little contact with other people and their experiences and programs.

So please send me some information about the Jupiter Ace Users Club.

Arve Asheim, Norway.

Answer: Yes, the Brother EP22 looks very good. Do tell us how standard the RS232 is!

J.N.

THE COMPLETE FORTH

Alan Winfield

This is a most welcome addition to the limited literature available for FORTH users and will be of interest to both new and more experienced users .

It starts by explaining FORTH in a simple and clear manner that may make clear any problems left after working through the ACE manual, and it includes exercises to check that it is fully understood .

It has the added advantage that, since it is based on FORTH-79, a better understanding of FORTH in general can be gained . Once the idea of screens, of definitions is understood and the action of words not in the ACE dictionary can be seen it is a relatively easy task to adapt any FORTH program to run on the ACE and so become less "machine dependant" .

The second half of the book looks into some more complex techniques of using FORTH and ends with two programs to illustrate some of them . There is a calendar listing that will set out the calendar for any month of any year or tell you the day any date falls on . This is both useful and, with help from the book, easy to implement on the ACE . (N.B. The definition of jan1st is incorrect in the blocks list but correct earlier in the chapter) The second is a squash program that requires more adjustment to suit the ACE display and is probably better as an introduction to some useful techniques rather than for the game itself .

At £6.95 the book is excellent value and should find a place on every ACE users bookshelf .

G.R. YORKE

By Ralph Bancroft

PCN readers have helped design what could be the successor to the Stricken Jupiter Acs (Issue 35).

A Brighton company, Microkey, demonstrated a pre-production version of its 4500 colour Forth computer to the Forth Interest Group in London last week.

The 4500 is unique in that the final configuration was designed with the help of potential end users — all of them **PCN** readers. Two thousand of you responded to an advert asking you to complete a questionnaire about what facilities you would like to see on the machine.

Paul Wynter, managing director of Microkey, expects to launch the 4500 in January. It will offer the choice between a 6502 or a 6809 as the main processor (both will run at 2 MHz), 128K of bank-switched RAM and a graphics resolution of 640 by 200 in colour or 1,280 by 200 in monochrome.

A disk system will be available at the same time using Sony micro-floppies.

The hardware for the 4500 was designed by David Huxley of Custom Computers. It is based on a single board computer he designed for Microkey's parent company

which manufactures front-end processors for typesetting machines.

The ROM-based Forth for the 4500 has been written by Sandy Robson of the systems engineering department of Brighton Polytechnic. The department has built up experience of using Forth on different machines for a variety of applications. Mr Robson has based the 4500's Forth on the Forth-79 standard with extensions to encompass the facilities available under Fig-Forth.

The ROM will include interpreter, high-level compiler, editor, double precision arithmetic and utilities including system monitor.

Interfaces on the 4500 are a cassette interface, disk interface, three bi-directional parallel ports and expansion bus. The parallel ports can be used for keyboards (by splitting the 128K RAM into two separate blocks of 64K, the 4500 can allow two users to use the machine in a multi-tasking mode).

Rather than configure the parallel ports for specific applications, eg a Centronics printer interface, Microkey intends that users will adapt them for their own purposes by means of 'pods' that will plug into the ports and provide the required configuration circuitry.

Further details:

Hardware: Microkey, 98a St James St,
Brighton (Paul Wynter, 0273-672950)
Software: Remsoft, 18 George St,
Brighton (John Noyce, 0273-602354)