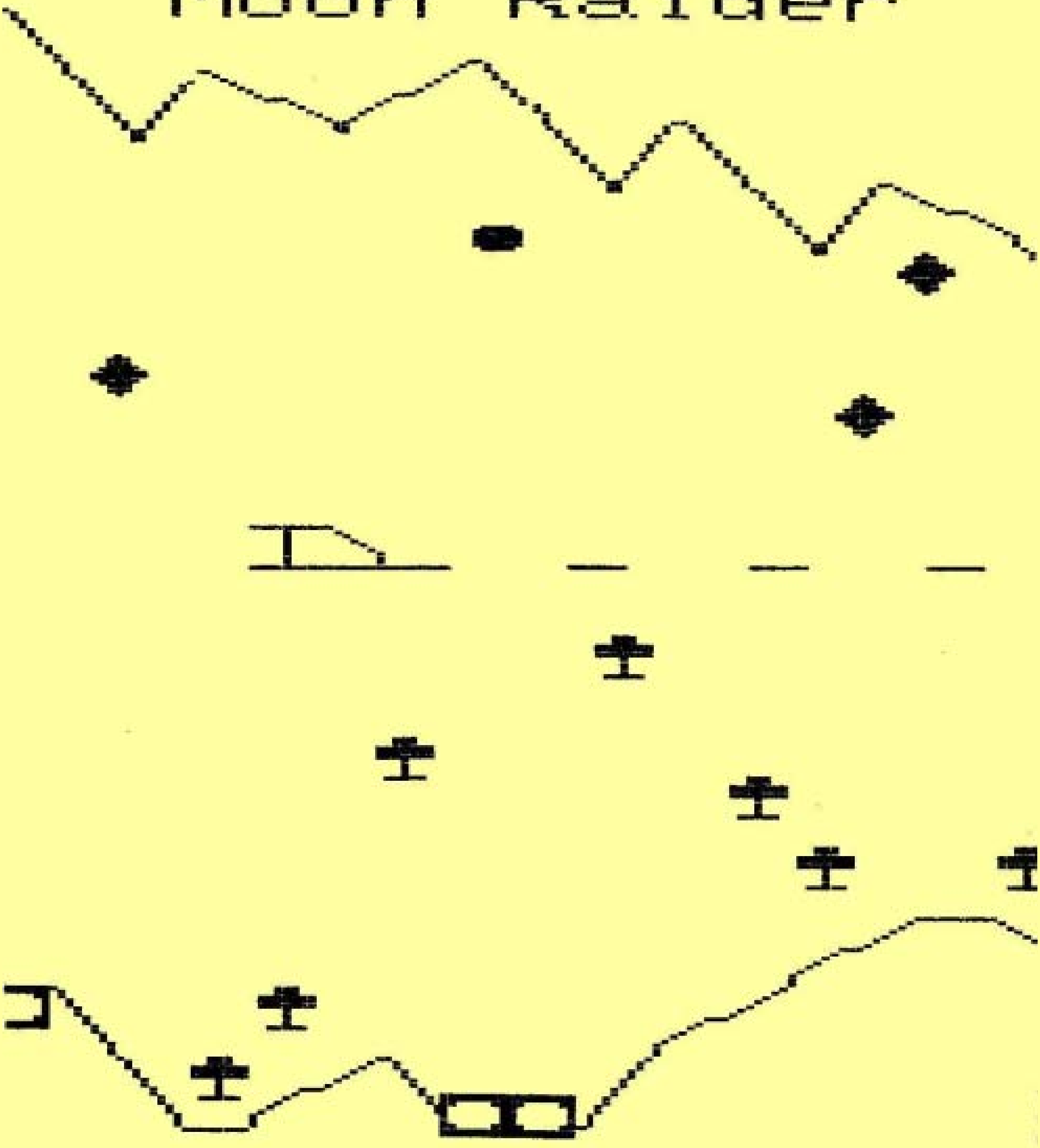
μP

MICROPOWER

VOLUME 2, NUMBER 1

A MAGAZINE FOR NASCOM USERS

Moon Raider



February, 1982

95 p

PROGRAM POWER

NASCOM 1 & 2

Nasprint 80

Nasprint 80 is a 2K progam which greatly extends and simplifies the operation of Nas-Pen.New functions supplied by Nasprint 80 include:

Pagination
Output a title in each page
Centre title
Text formatting with embedded control codes, e.g. Change line length; change line spacing; change margins; centre line between margins; new page; output control codes to printer.

The program contains a parallel printer routine for a Centronics type interface, specifically designed for the Epron MX-80,but the program can be used with any printer, parallel or serial, as the output is routed through an address in RAM.

The program also facilitates the operation of a printer with Zeap, Nas-Dis, De-bug, Nas-Sys & ROM Basic; the software/firmware being used is selected from a menu and Nasprint 80 then changes the necessary addresses to produce hard copy output.

The program is supplied in 2x2708's or in one 2716,together with instructions for fitting 2716's in the RAM A card. £14.95

New Fase (16K/MC/G)

New version of the space invaders type with each new fleet of invaders having a different shape & kind of action.Missiles fired at you come straight down or diagonally left to right & vice versa.

Destroy one 'fase' & move onto the next.The fuel level is shown graphically and you can refuel if you obliterate four fleets.Your score is shown at the end of a game and the top ten scorers are ranked.Once again the difficulty level has been set very high. £7.95

AY-3-8910 Programmable Sound Chip

YES – This IS the amazingly powerful "Clang, Bang, Zap, Tweet" sound & music generator, with three channels which can be independently programmed for sound output and amplitude.In addition it has an 'envelope controlled' noise generator, ideal for creating explosions and firing sounds. £6.45

Sound Chip Data Manual (60 pages)

This contains a full description of the architecture & operation of the chip, detailed advice on the interfacing to various microprocessors, and comprehensive explanations on the generation of music and sound effects. £2.25 (no vat)

Sound Chip Interface Board

The board has been designed to interface between the Parallel Input/Output Port (PIO) of the Nascom and the sound chip. It is supplied ready-built and just plugs straight onto your PIO connector. Nascom 1 connectors available on request. Sound generation is illustrated in machine code & Basic (chip not included) £13.50

Sound Chip Demo Program (MC)

A brief summary of the main registers is given, together with a description of their functions. Thereafter, two separate modes may be selected. Direct mode allows values to be entered into the chip registers via the keyboard, making experimentation simple, thus leading to a rapid appreciation of the chip's potential. The second mode turns the keyboard into a 7 octave 'piano', displaying the notes being played as well as the values in the registers. £5.95

*** NASCOM 1 – Cottis Blandford cassette interface for N2 format, reliability & fast load £14.90
- 8K RAM required unless otherwise stated
- Please state if Nascom TAPE Basic required.

ALL PROGRAMS SUPPLIED ON CASSETTE IN CUTS/KANSAS CITY FORMAT

Please add 55p/order P & P + VAT @ 15%.
Large (15½p) Sae for FULL CATALOGUE.

PROGRAM POWER
5, Wensley Road
Leeds LS7 2LX.

CONTENTS

EDITORIAL

This is the first issue of Volume 2 of micropower and the start of our first full year of publication. There will be six issues this year at approximately 2 month intervals, so the next magazine will appear at the end of April.

When we started the magazine in August last year (what a long time ago that seems) we were unsure that there would be enough interest in a Nascom magazine to provide the necessary support, although we felt that was a tremendous fund of enthusiasm in the Nascom clubs and computer groups up and down the country.

It appears that the interest does exist - the magazine has sold well (we have had to have reprints of the early issues) and, even more important, articles have been sent in for publication. This does not mean that we have enough articles for the magazine - we are always pleased to receive more, so sit down NOW and write up your pet projects.

Remember that the composition of the magazine reflects the interests of the ‘active’ readers - that is, the readers who also contribute. If you feel that the magazine should contain articles on a certain topic you could write in and ask someone to write them, but a better way is to write a short article yourself. You have a head start, because all your readers will be Nascom enthusiasts; all you have to do is to communicate your particular interest.

NASCOM CONTROLS WASHING MACHINE

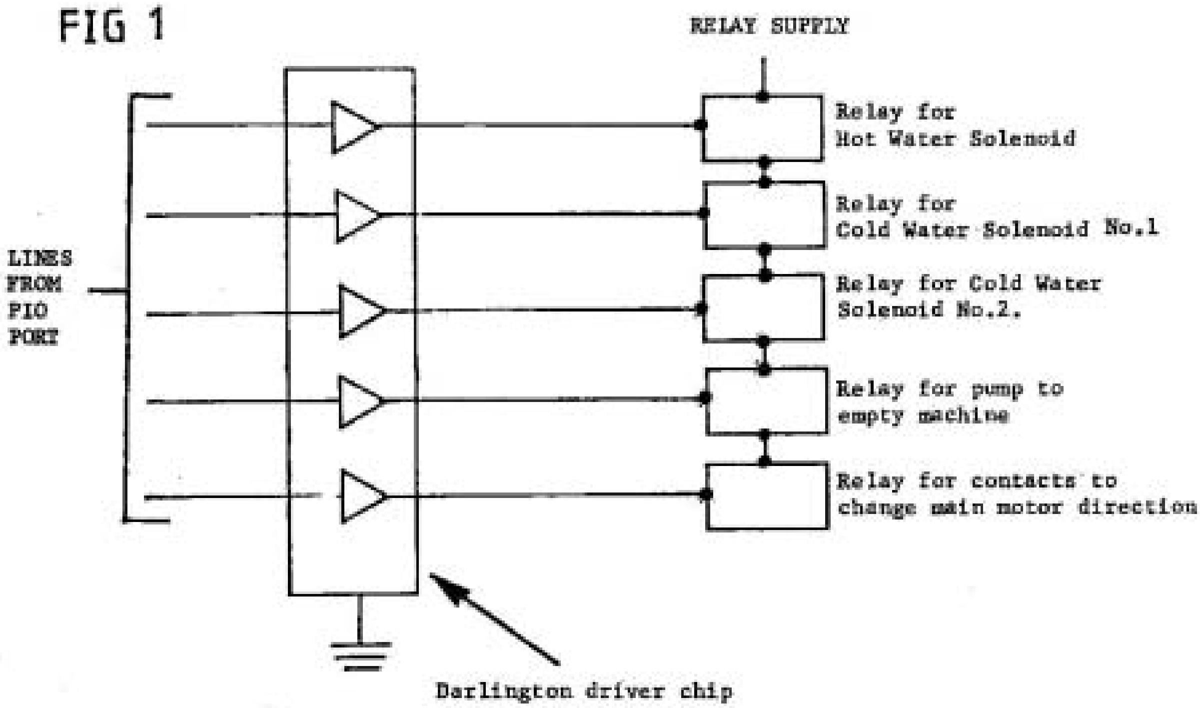
by J. C. Lord

Some months ago I was given a Hoover Keymatic front loading washing machine which needed a new controller. The cost of such a controller being at that time about £60, I decided to have a go at controlling the machine from my Nascom. The project has proved successful, and I think that readers may be interested to see how the job was done using ten P.I.0. lines.

I removed from the washing machine the main motorised controller unit, the motor drive board, and all of the wiring. This left the two level switches (empty and full), the three solenoids (two for cold water, and one for hot), the pump and the main motors. I then built a board to interface between the Nascom and the various items on the washing machine.

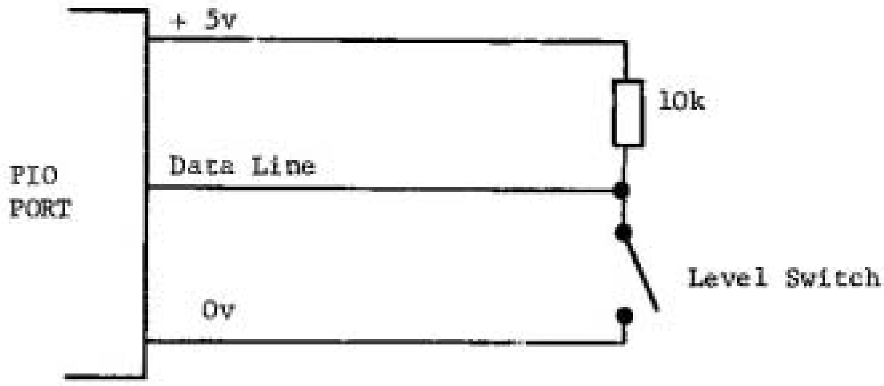
Five of the P.I.O. lines are used as on/off outputs. The output system consists of a Darlington driver chip operating five transistorised relays. The Darlington driver acts as a buffer to provide 7 mA for the relays; the P.I.0. should not source or sink more than 2 mA. The relays switch 240 V. A.C. for the various machine components, as shown in figure 1. The 240 V. to microprocessor isolation in this circuit is provided by the relays. An alternative method would be to use optically isolated solid state relays.

FIG 1



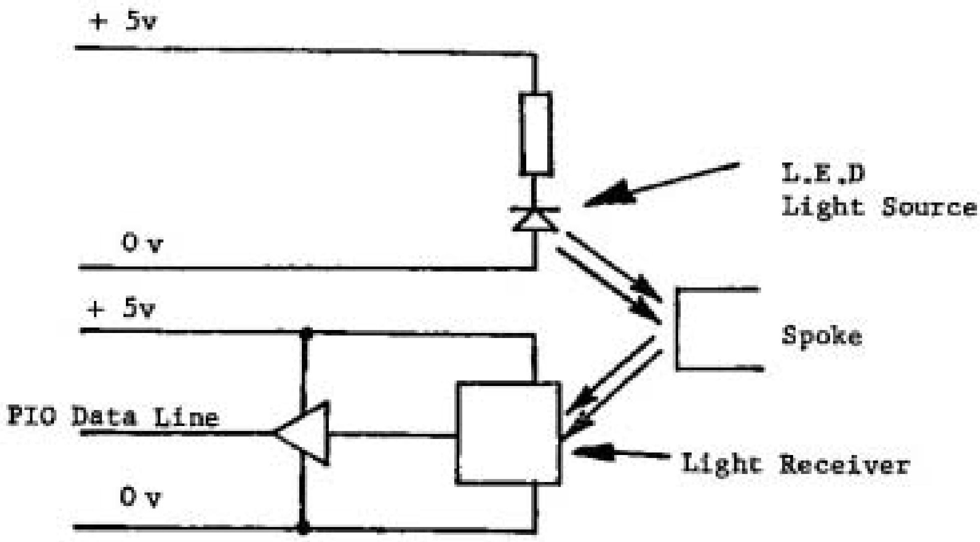
Three lines are used as Inputs. Two input signals come from the level switches which indicate when the drum is full and empty. As shown in figure 2, these switches are connected to +5 V. through 10 Kohm pull up resistors.

FIG 2



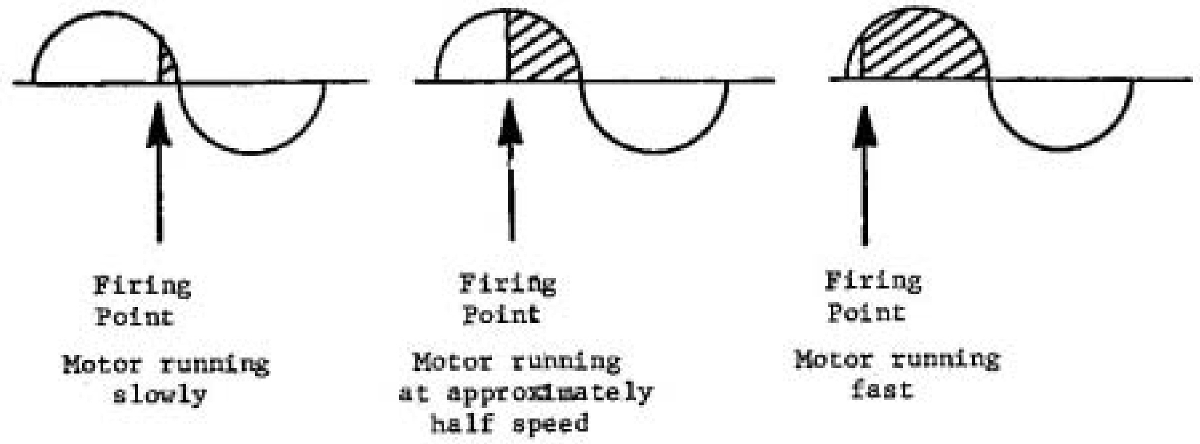
The third input is used by the processor to determine the speed of the drum for feedback purposes. A reflective optical switch points at the metal spokes of the pulley which turns the drum. When a spoke goes past the switch light is reflected and a ‘1’ is generated on the P.I.0. input line, which is usually at ‘0’. The circuit requires a transistor amplifier to step up the output from the receiver to TTL level for the P.I.0.

FIG 3



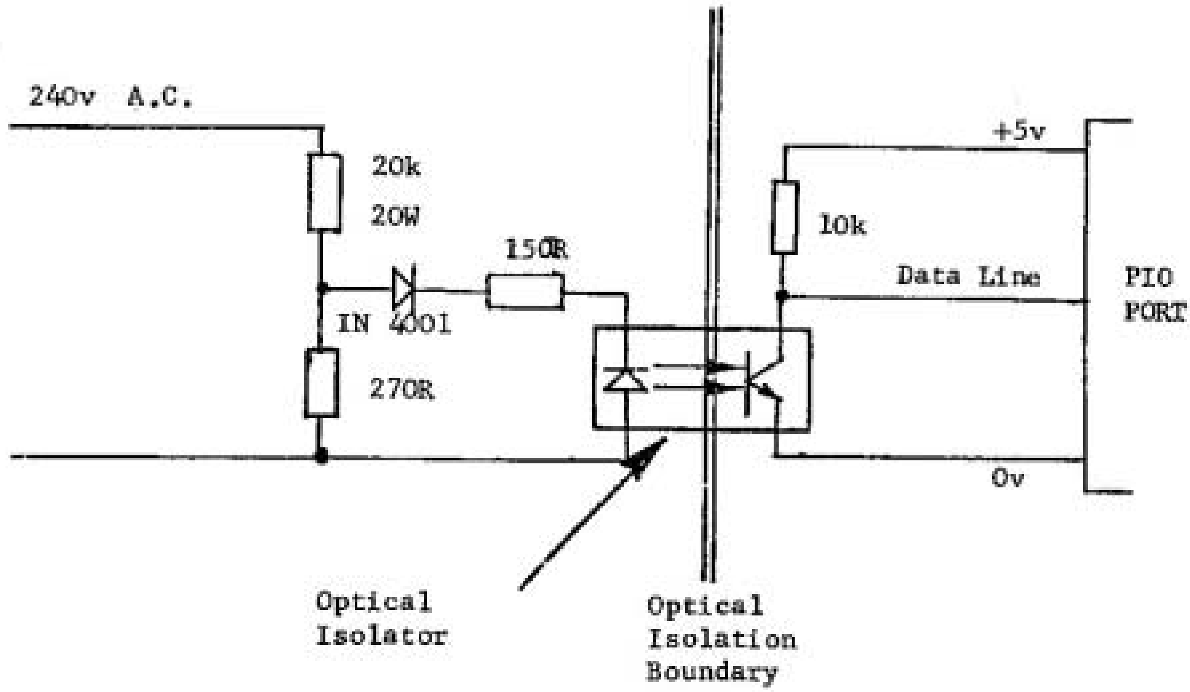
By timing the interval between successive spokes, the speed of the drum can be found, and hence the required speed feedback is obtained. An alternative to this optical sensor would be a 5 V. proximity switch.

So far so good, now we come to the problem of controling the speed of the main motor. The circuit uses a thyristor - a device which can be made to pass current into the motor for part of the positive half of the A.C. wave. The earlier in the half wave it is switched on, the more current it will pass and the faster the motor will go (see figure 4).



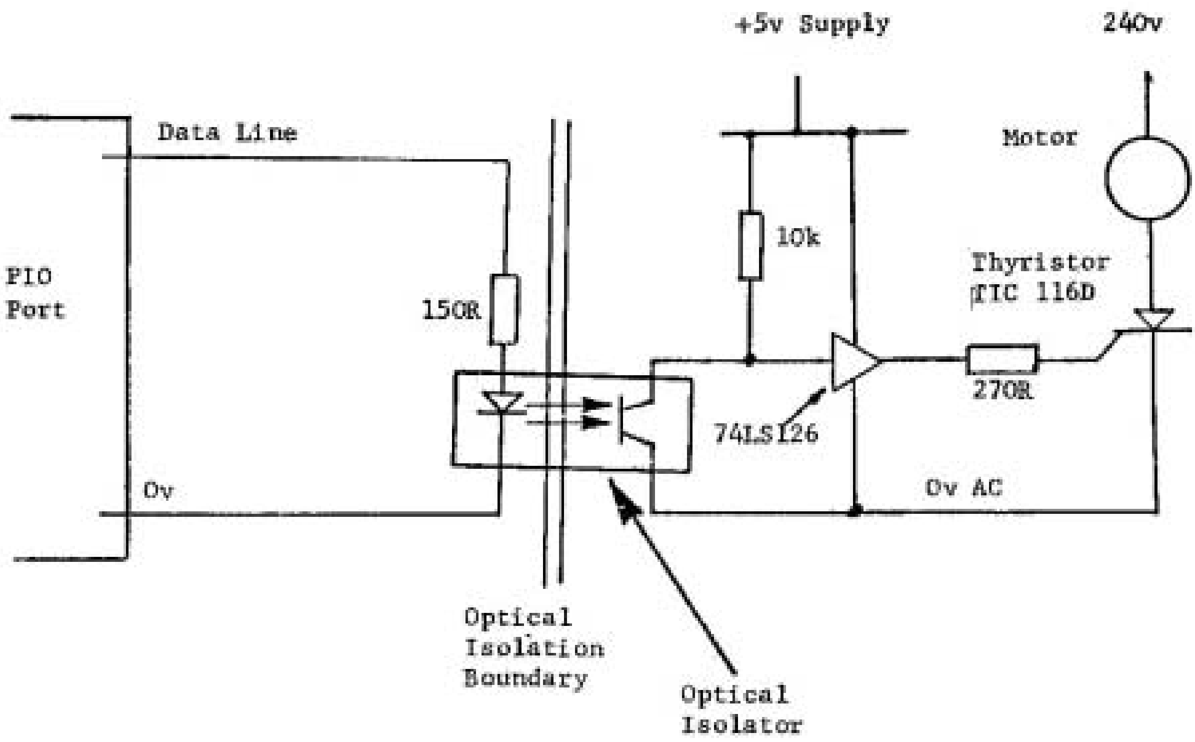
Thus in order to control the speed a pulse has to be sent to the thyristor to switch it on at the required point in the positive half cycle. The circuit of figure 5 generates a reference pulse for the processor shortly after each positive-going zero crossing point.

FIG 5



t15The P.I.0. line which receives the signal is an interrupt line, so that each time the zero crossing point signal is received the processor enters an interrupt service routine. In this routine the thyristor is fired after a delay calculated from the programmed speed requirement and the feedback speed. Figure 6 shows the firing circuit. Adequate suppression has to be included to prevent interference with other domestic equipment.

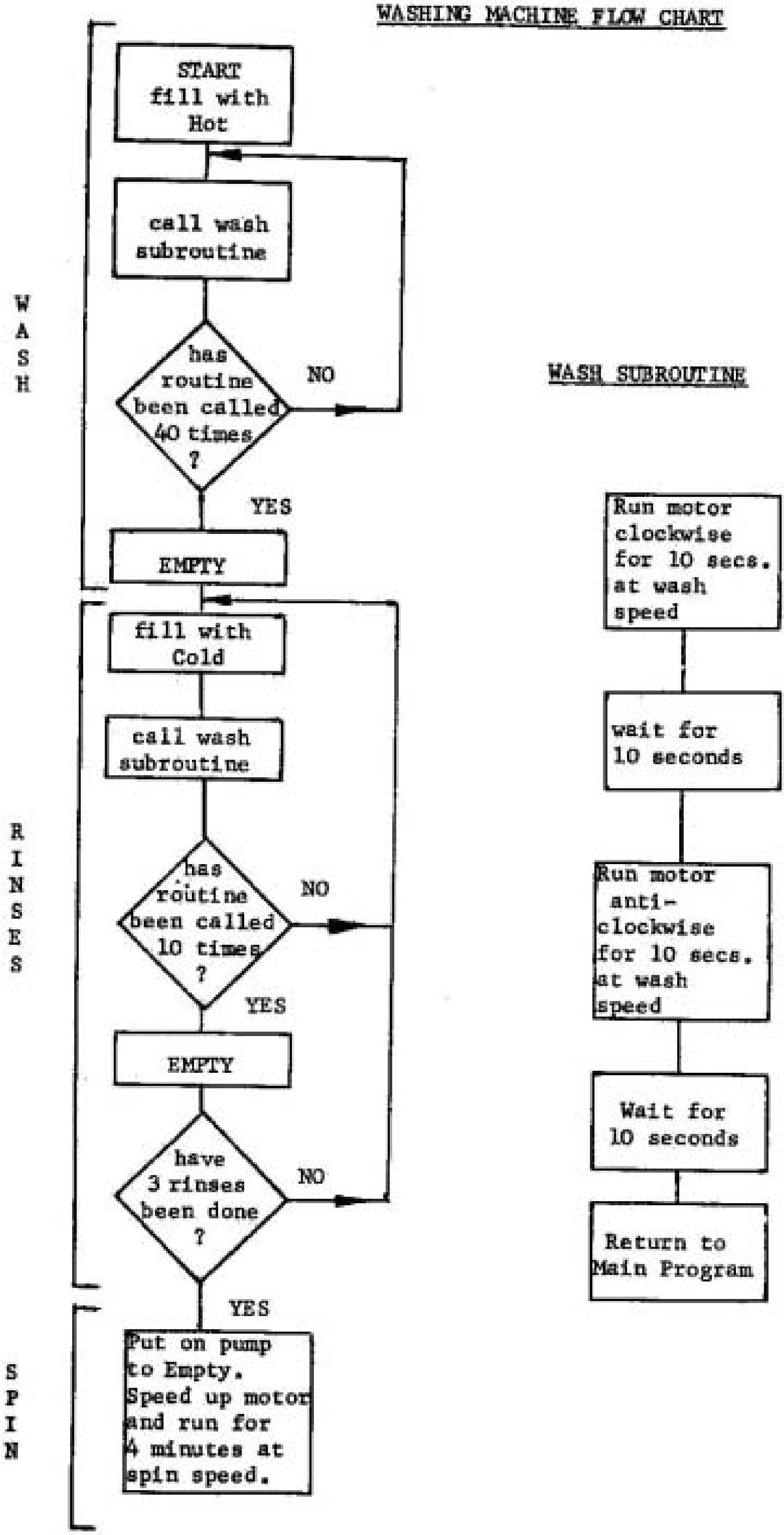
FIG 6



The 5 V. power supply for the circuit to the right of the isolation boundary must be derived from the 240 V. supply. If the 5 V. Nascom supply was used, there would be no isolation.

The program for controlling the operation of the system, which consists of sequencing for filling, emptying, pumping out and the motor control subroutine, was developed using the Zeap assembler. A flow chart for the system on which the program was based, is shown on the next page.

* + · + * + · + * + · + * + *


WASHING MACHINE FLOW CHART
START
fill with
Hot
call wash
subroutine
has
routine
been called
40 times
?
NO
YES
EMPTY
fill with
Cold
call wash
subroutine
has
routine
been called
10 times
?
NO
YES
EMPTY
have
3 rinses
been done
?
NO
YES
Put on pump
to Empty.
Speed up motor
and run for
4 minutes at
spin speed.
W
A
S
H
R
I
N
S
E
S
S
P
I
N
WASH SUBROUTINE
Run motor
clockwise
for 10 secs.
at wash
speed
wait for
10 seconds
Run motor
anti-
clockwise
for 10 secs.
at wash
speed
Wait for
10 seconds
Return to
Main Program

nascom

- Housed in strong, stylish case with high quality QWERTY keyboard.
- 0.7Mbyte floppy disc system available in matching case.
- Full 8K RAM, expandable to 200K with page mode and RAM boards.
- Factory-built options plus additional range of Nascom-approved hardware and software.

Think of Nascom 3 as an advanced personal computer, built to professional standards and offering the total systems versatility needed by enthusiasts whose imaginations are already ahead of the toy computer field.

Think of Nascom 3 as the powerful heart of a truly versatile educational or business computer system, with added peripherals and an extensive range of firmware and software options.

Or think of Nascom 3 as a custom-structured industrial control unit, well capable of cutting production costs in many key areas.

Nascom 3, reliable, expandable, affordable – and backed by one of Britain's best known engineering groups. Think about it.

OTHER NASCOM PRODUCTS

- Nascom 1 from £125 + VAT
- Nascom 2 from £225 + VAT
- Memory Extension Unit from £80 + VAT
- Disc systems from £375 + VAT
- Input/Output board from £37 + VAT

NEW

- Advanced video controller from £155 + VAT
- Enhanced BASIC from £40 + VAT
- Pascal compiler from £45 + VAT
- Compiled BASIC from £150 + VAT

SPECIAL OFFER IMP PRINTERS £199 + VAT WHILE STOCKS LAST

NASCOM DEALERS

UK

Bits & PCs
Wetherby [illegible]

Business & Leisure Microcomputers
Kenilworth [illegible]

[illegible]
Edinburgh [illegible]

Crystal Electronics
Torquay [illegible]

Electrovalue Ltd
Egham [illegible]
Manchester [illegible]

Eley Electronics
Leicester [illegible]

Henry's Radio
London [illegible]

[illegible]

Interface Components Ltd
Amersham [illegible]

[illegible]

Lucas Computer Centre
Leeds

[illegible]

[illegible]

[illegible] Computer Centre
Crewe [illegible]

[illegible]
London [illegible]

[illegible]

[illegible]

[illegible]
London [illegible]

[illegible]

[illegible]
Brighton [illegible]

Target Electronics
Bristol [illegible]

Trading Post
Hastings [illegible]

[illegible] Computers
[illegible]

OVERSEAS

[illegible]
Copenhagen [illegible] (Denmark)

[illegible]
Paris [illegible] (France)

[illegible] (Germany)

[illegible] Computer Consultants
[illegible] (Netherlands)

[illegible] AB
[illegible] (Sweden)

[illegible] Instruments
Helsinki [illegible] (Finland)

Lucas Logic

BEYOND THE 64K BARRIER

by Chris Blackmore

Anyone who has owned a Nascom for any great length of time will tell you that, unlike systems that arrive in ready-made plastic cases, Nascoms tend to expand almost indefinitely. The abilty to expand was designed into the system from the very beginning, and it shows. When you start out, you think that a 32K RAM board will never be too small. There is left over space in the memory map, and a great many of the expansion boards that are available are memory mapped, which means that they appear to the CPU to be memory, and they take up memory space. Then one day you upgrade your system to 64K of RAM, or perhaps even more than that, because the Nascom will allow you to have enormous amounts of RAM (unlike the aforementioned plastic boxes...). Now your memory mapped sound generator board, programmable character generator, and home made digital clock card become nuisances, as they overlap some of your nice new memory.

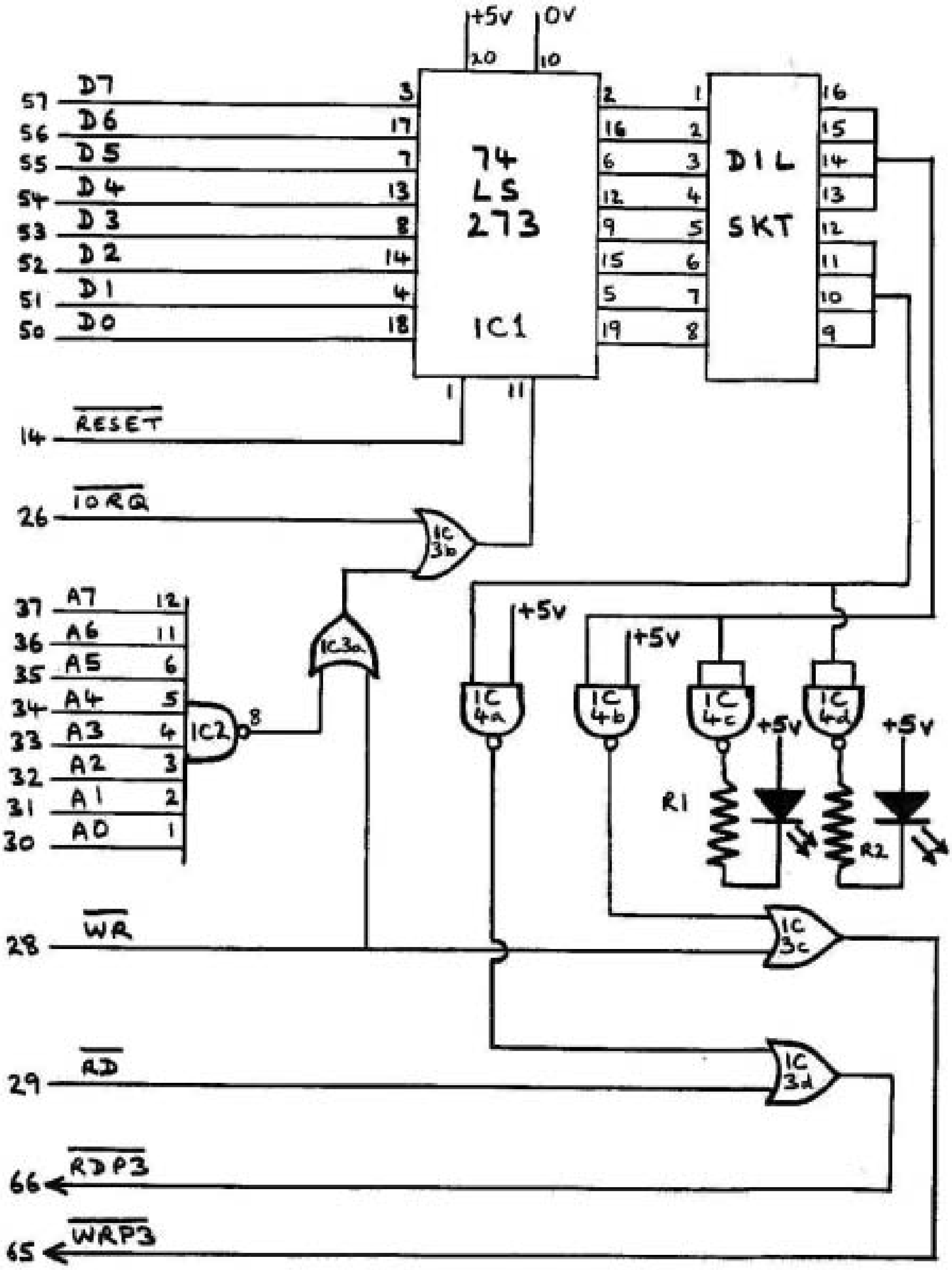
So you have to find a way of preventing the waste of memory, and the Nascom provides it in the form of memory paging. Your main RAM board will be on page 0, and all your memory mapped ‘extras’ (unless you have more than 64K of them) will be on page 1, 2 or 3.

So this means that you will have to fit a paging circuit to each of these boards, doesn’t it? No, it doesn’t - one will do! There are some spare lines on the bus, called NDEF1 and NDEF2 in the Gemini 80 specification, and shown as ‘reserved’ in the Nasbus specification, which can be used in more or less any way you wish. I suggest that they should be used to carry the read and write signals to all boards that are on memory page 3.

Only a simple modification is needed on each of these boards, instead of the quite complex task of adding a paging circuit to each of them. The single paging circuit can be built on a prototyping board, of which there tends to be at least one in any system that has reached this stage of development.

Modify each board that is to appear on page 3 as shown in diagram 2, or fit the board with equivalent switching if you want the board to be usable on systems without the page 3 signals, or with software that has not allowed for the use of paged addressing.

The circuit to provide the page 3 read and write signals is shown in diagram 1. It will not take up much space on the prototyping board, as it consists of very few components. It could even be fitted to the end of the mother board if you are that short of space. A header should be made up with links from pins 1 to 16 and 5 to 12 for page 3 operation; different header connections will allow for different page selections it is even possible to have the read signal on one page and the



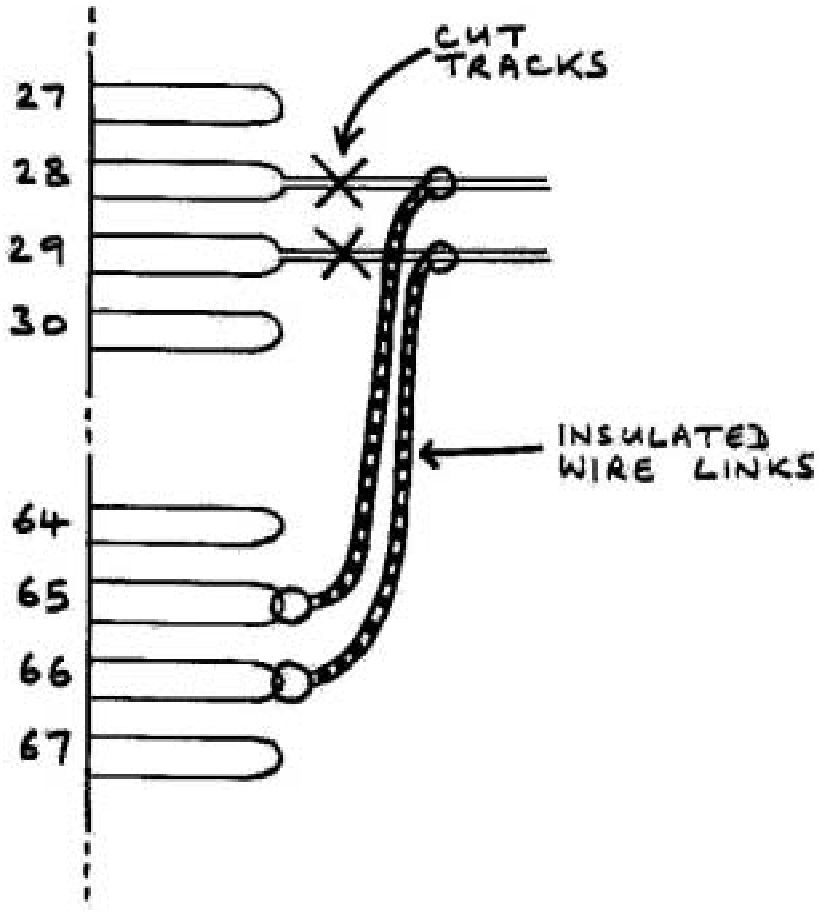
DIARAM 1: PAGING CRCUIT

IC 1	74 LS 273	IC 4	74 LS 00
IC 2	74 LS 30	R1	150 Ohms
IC 3	74 LS 32	R2	150 Ohms

write signal on another, but even I don't know why anyone would want to do this! Two LEDs have been added for the benefit of those who, like me, feel that computers should have flashing lights on them - these are handy for checking that the circuit is actually operating, when your program to use it seems not to be working....

REFERENCES:
The full definition of the Gemini 80 bus can be found in INMC-80, No. 4, pages 24-30
The original Nasbus is defined in Nascom Document PF/007 Issue No. 1.

DIAGRAM 2: BOARD MODIFICATIONS



NASCOM USERS

Take a look at the NASCOM APPROVED HS-IN STORAGE SYSTEM. Where else can you get features like these . . .

- A full on screen instant display of the catalogue
- Auto vertification of each file as it is written.
- CRC error checking.
- Link selectable 2Mhz or 4Mhz option.
- Fast data transfer rate of 6000 bps.
- Powered from NASBUS.
- 8" sq NASBUS compatible PCB.
- Far more reliable than any floppy disk system.
- 112K on-line storage with 2 drive system.

The HS-IN has a Command Set which makes it a floppy-disk "look-alike". It can load an 8K program in under 11 seconds and can store up to 56K (28 files) on each side of tape. Why spend £700 on a floppy disk system when the less expensive HS-IN system has a command set like this . . .

BRIDGE THE GAP BETWEEN EXPENSIVE FLOPPY DISK SYSTEMS AND UNRELIABLE CASSETTES.

B — Write a Basic file
C — Instant display of catalogue.
D — Delete file.
J — Jump to Basic.
N — Jump to NAS-SYS.
Q — Warm start to NASPEN text editor.
R — Read a file.
T — Transfer file to another drive.
W — Write a file.
X — Exit and rewind cassettes.
Z — Warm start to Basic.

SEE us at the SCOTTISH COMPUTER SHOW Albany Hotel, Glasgow 16th-18th MARCH

This Mini-Cassette Storage System is technologically far ahead of anything like it on the market and is extremely reliable into the bargain. AND THE COST?

Single Drive System built and tested	£199
Double Drive System built and tested	£279

Carriage £3.50.

OFFER NO 1
ALL RAM B boards supplied until April 30th come with an **EXTRA 32K FREE** on board.

OFFER NO 2
NASCOM 2 built, 48K RAM B board built, 3A PSU
ONLY £360 + VAT. **SAVE £37.50** SEE OFFER 1

OFFER NO 3
NASCOM 2 built, 48K RAM B board built, 3A PSU, HS-IN SINGLE DRIVE SYSTEM
ONLY £530 + VAT. **SAVE £66.50** SEE OFFER 1

ALL OFFERS END APRIL 30TH 1982

OFFER NO 4
NASCOM 2 built, 48K RAM B board built, 3A PSU, HS-IN single drive system, EPSON MX80FT-1, NASPEN & ALL CABLES **ONLY £900** + VAT. **SAVE £126.50**
SEE OFFER 1

OFFER NO 5
NASCOM 3, 48K RAM B built, Gemini Intelligent Video Card (IVC)
ONLY £540 + VAT. **SAVE £85** SEE OFFER 1

OFFER NO 6
SHARP MZ80K WITH SUPER GRAPHICS +5 GAMES, EPSON MX80FT-1 WITH PAPER.
ONLY £825 + VAT. **SAVE £229**

MICRO-SPARES JOIN MICROVALUE GROUP

MICRO-SPARES have now become the **MICROVALUE GROUP** member supplying Scotland and now add super new products like the Gemini, Sharp & Epson to the **MICRO-SPARES** range.

QUALITY MEMORIES

AT PRICES THAT CANNOT BE BEATEN IN THE U.K.
MICRO-SPARES can supply these memories in quantities from 1 to 10,000+. Parts delivery is fast — orders received by 4.30pm are shipped same day.

All memories are guaranteed for 1 year from date of purchase. Memories supplied are good quality but should you have a faulty part a replacement will be sent as soon as the part is received — without question.

Thousands of memories have already been supplied to Manufacturers Computer Traders, Government Bodies and Individuals all over the U.K. and the continent. If you are buying in large quantities please telephone for price. Official orders are welcome!

		1—49	50—249
2114L	(200ns & 300ns) low power Suitable for Acorn Atoms	98p	93p
2114N	(200ns & 300ns)	98p	93p
4118	(250ns)	63p	63p
	(200ns)	66p	63p
	(150ns)	78p	73p
2708	(450ns)	1.40p	1.34p
2716		2.06p	1.92p

Buy an EPSON PRINTER — Get a COMPUTER FREE!

YES — YOU WILL RECEIVE A FREE SINCLAIR ZX81 WITH EVERY ONE OF THE MODELS BELOW. EVEN THOUGH THE PRICE IS GOOD — BUT HURRY — THIS OFFER LASTS AS LONG AS THE ZX81's ARE IN STOCK

EPSON MX80T	£369 + VAT
EPSON MX80FT1	£399 + VAT
EPSON MX80FT2 (new type)	£465 + VAT
EPSON MX100	£575 + VAT

PAYMENT AND DELIVERY
Payment is by Cheque, Postal Order, ACCESS, VISA etc. PLEASE add postage and VAT. Postage on component orders under £30 is 50p. All in stock items sent same day. All none Kit items have a 1 year guarantee. Official orders welcome. Discount on large orders by arrangement.

SUPPLIERS TO TRADE
LOCAL GOVERNMENT
EDUCATION
INDIVIDUALS
INDUSTRY

EDINBURGH

Micro-Spares

19 Roseburn Terrace, Edinburgh EH12 5NG
Tel: 031-337 5611

BARCLAYCARD VISA

SCOTLAND

COMPUTERS
PERIPHERALS
COMPONENTS
& NATIONWIDE
MAINTENANCE

XTAL BASIC EXTRA

by David Elliott

Here are six more commands tor Crystal Basic 2.2. OLD recovers a program which has been 'exterminated' by a NEW or a cold start. STRING$(nn,cc) creates a string of nn characters with ASCII code cc. LOWER$(..) and UPPER$(..) convert a given string to lower and upper case respectively, while REVERSE$(..) changes the case of the characters in a string. Finally, EVAL(…) returns the value of a given string. The assembler listing of the commands is followed by the code to be entered using the loader program described in the last issue.

```
2D00              0090          ORG £2D00
                  0100   ;
                  0110   ; @@@@@@@@@@@@
                  0120   ; @@    O  L  D    @@
                  0130   ; @@@@@@@@@@@@
                  0140   ;
                  0150   ; OLD RECOVERS A PROGRAM PREVIOUSLY
                  0160   ; ERASED BY A 'NEW' COMMAND
                  0170   ;
2D00   FD2A8312   0180   OLD     LD IY, (1283H)   ; GET TEXT START
2D04   FDE5       0190           PUSH IY          ; SET IX TO START
2D06   DDE1       0200           POP IX
2D08   AF         0210           XOR A
2D09   0600       0220           LD B, 0          ; DOING FIRST LINE
2D0B   110400     0230           LD DE, 4         ; SKIP LINE NUMBER
2D0E   FD19       0240           ADD IY, DE       ; AND LINK ADDRESS
2D10   FDBE00     0250   OLD1    CP (IY)          ; END OF LINE?
2D13   2804       0260           JR Z, OLD3       ; IF SO, JUMP
2D15   FD23       0270   OLD2    INC IY           ; NO, TRY NEXT BYTE
2D17   18F7       0280            JR OLD1
                  0290   ;
2D19   78         0300   OLD3    LD A,B           ; END OF 1ST   LINE?
2D1A   B7         0310           OR A
2D1B   200D       0320           JR NZ, OLD4
2D1D   0601       0330           LD B,1           ; IF SO, RESET FLAG
2D1F   FD23       0340           INC IY           ; RESTORE LINK
2D21   FDE5       0350           PUSH IY          ; TO SECOND LINE
2D23   D1         0360           POP DE
2D24   DD7300     0370           LD (IX),E
2D27   DD7201     0380           LD (IX+1),D
                  0390   ;
2D2A   AF         0400   OLD4    XOR A
2D2B   FDBE01     0410           CP (IY+1)        ; END OF PROGRAM?
2D2E   20E5       0420           JR NZ,OLD2       ; NO, TRY AGAIN
2D30   FDBE02     0430           CP (IY+2)        ; END OF PROGRAM?
2D33   20E0       0440           JR NZ, OLD2      ; NO, TRY AGAIN
                  0450   ;
                  0460   ; END OF PROGRAM FOUND
                  0470   ;
2D35   110300     0480           LD DE, 3         ; RESTORE POINTERS
2D38   FD19       0490           ADD IY, DE
```

```
2D3A  FD22B70C   0500              LD (£0CB7), IY     ; END OF TEXT
2D3E  FD22BB0C   0510              LD (£0CBB), IY     ; END OF ARRAYS
2D44  C9         0520              RET
                 0530  ;
                 0540  ; @@@@@@@@@@@@@@@@@@@@@
                 0550  ; @@        STRING$(NN,CC)        @@
                 0560  ; @@@@@@@@@@@@@@@@@@@@@
                 0570  ;
                 0580  ; CREATES A STRING OF NN CHARACTERS
                 0590  ; WITH AN ASCII CODE CC
                 0600  ;
2D43  E1         0610  STRING      POP HL
2D44  23         0620              INC HL
2D45  CD5022     0630              CALL IN255         ; GET NUMBER 0-255
2D48  F5         0640              PUSH AF
2D49  CD4C15     0650              CALL TSTCOM        ; GET COMMA
2D4C  CD5022     0660              CALL IN255         ; GET CHAR. CODE
2D4F  F5         0670              PUSH AF            ; SASVE CHARACTER
2D50  CD5115     0680              CALL TSTCHR        ; TEST FOR
2D53  29         0690              DEFB “)            ; CLOSING BRACKET
2D54  F1         0700              POP AF             ; RECOVER CHAR.
2D55  47         0710              LD B,A             ; PUT IN B
2D56  F1         0720              POP AF             ; RECOVER NUMBER
2D57  E5         0730              PUSH HL            ; SAVE TEXT POINTER
2D58  4F         0740              LD C, A            ; PUT NUMBER IN C
2D59  79         0750              LD A, C
2D5A  C5         0760              PUSH BC            ; SAVE BC
2D5B  CDAB1F     0770              CALL ASNSTR        ; CREATE NEW STRING
2D5E  C1         0780              POP BC             ; RECOVER BC
2D5F  EB         0790              EX DE, HL          ; SET HL TO START
2D60  70         0800  ST1         LD (HL), B         ; COPY CHARACTER
2D61  23         0810              INC HL             ; INCREMENT POINTER
2D62  0D         0820              DEC C
2D63  20FB       0830              JR NZ, ST1         ; LOOP
2D65  C3D91F     0840              JP STREND          ; RETURN TO BASIC
                 0850  ;
                 0860  ; @@@@@@@@@@@@@@@@@@@@@@@@@@@
                 0870  ; @@          CONVERT TO LOWER CASE          @@
                 0880  ; @@@@@@@@@@@@@@@@@@@@@@@@@@@
                 0890  ;
2D68  3E4C       0900  LOWER       LD A, “L           ; SET FLAG TO
2D6A  32FE2D     0910              LD (CONV), A       ; LOWER CASE
2D6D  180C       0920              JR CONVRT
                 0930  ;
                 0940  ; @@@@@@@@@@@@@@@@@@@@@@@@@@@
                 0950  ; @@          CONVERT TO UPPER CASE          @@
                 0960  ; @@@@@@@@@@@@@@@@@@@@@@@@@@@
                 0970  ;
2D6F  3E55       0980  UPPER       LD A, “U           ; SET FLAG TO
2D71  32FE2D     0990              LD (CONV), A       ; UPPER CASE
2D74  1805       1000              JR CONVRT
                 1010  ;
                 1020  ; @@@@@@@@@@@@@@@@@@@@@
                 1030  ; @@         REVERSE STRING        @@
                 1040  ; @@@@@@@@@@@@@@@@@@@@@
                 1050  ;
2D76  3E52       1060  REVRSE      LD A, “R           ; SET FLAG TO
```

```
2D78    32FE2D    1070              LF (CONV), A      ; REVERSE
                  1075    ;
                  1080    ; CONVERT STRING
                  1085      ;
2D7B    E1        1090    CNVERT    POP HL
2D7C    23        1100              INV HL
2D7D    CD8B1B    1110              CALL EXPR         ; GET STRING
2D80    CD5115    1120              CALL TSTCHR       ; TEST FOR
2D83    29        1130              DEFB “)           ; COSING BRACKET
2D84    E5        1140              PUSH HL
2D85    CD6A21    1150              CALL ASCO         ; GET STRING ADD.
2D88    2B        1160              DEC HL            ; AND LENGTH
2D89    2B        1170              DEC HL
2D8A    2B        1180              DEC HL
2D8B    46        1190              LD B, (HL)        ; B=STRING LENGTH
2D8C    EB        1200              EX DE, HL         ; HLSTRING ADDRESS
                  1210    ;
                  1220    ; GOT STRING ADDRESS
2D8D    E5        1230              PUSH HL           ; SAVE IT
2D8E    E5        1240              PUSH HL           ; SAVE IT
2D8F    78        1250              LD A, B           ; SET A TO LENGTH
2D90    CDA81F    1260              CALL ASNSTR       ; CREATE NEW STRING
2D93    C1        1270              POP BC
2D94    E1        1280              POP HL
2D95    3AFE2D    1290    CNV1      LD A, (CONV)      ; GET FLAG
2D98    FE52      1300              CP “R             ; REVERSE?
2D9A    2824      1310              JR Z, REVSTR      ; IF SO, JUMP
2D9C    FE4C      1330              JR NZ, CVNU       ; IF NOT, UPPEROOD
2DA0    7E        1340              LD A, (HL)
2DA1    FE41      1350              CP “A             ; BEFORE “A”?
2DA3    3813      1360              JR C, CNV2        ; IF SO, LEAVE
2DA5    FE5B      1370              CP “Z+1           ; AFTER “Z”?
2DA7    300F      1380              JR NC, CNV2       ; IF SO, LEAVE
2DA9    C620      1390              ADD A, 20H        ; CONVERT
2DAB    180B      1400              JR CNV2
2DAD    7E        1410    CNVU      LD A, (HL)
2DAE    FE61      1420              CP “a             ; BEFORE “a”?
2DB0    3806      1430              JR C, CNV2        ; IF SO, LEAVE
2DB2    FE7B      1440              CP “z+1           ;   AFTER “z”? 2DB4
3002              1450              JR NC, CNV2       ; IF SO, LEAVE
2DB6    D620      1460              SUB 20H           ; CONVERT
2DB8    12        1470    CNV2      LD (DE), A        ; SAVE CHARACTER
2DB9    13        1480              INC DE
2DBA    23        1490              INC HL
2DBB    10D8      1500              DJNZ CNV1
2DBD    C3D91F    1540              JP STREND         ; BACK TO BASIC
2DC0    48        1550    REVSTR    LD C, B           ; SET BC TO LENGTH
2DC1    0600      1560              LD B, 0
2DC3    09        1570              ADD HL, BC        ; ADD TO START
2DC4    2B        1580              DEC HL            ; TO GET END
2DC5    41        1590              LD B,C            ; SET B TO LENGTH
2DC6    7E        1600    REV1      LD A, (HL)        ; COPY CHARACTER
2DC7    12        1610              LD (DE), A
2DC8    13        1620              INC DE            ; ALTER POINTERS
2DC9    2B        1630              DEC HL
2DCA    10FA      1640              DJNZ REV1         ; LOOP
2DCC    C3D91F    1650              JP STREND         ; BACK TO BASIC
```

```
                1670  ; @@@@@@@@@@@@@@@@@@@@@@@@@@@@
                1680  ; @@          EVALUATE A STRING           @@
                1690  ; @@@@@@@@@@@@@@@@@@@@@@@@@@@@
                1700  ;
2DCF  E1        1710  EVAL     POP HL
2DD0  23        1720           INC HL
2DD1  CD8B1B    1730           CALL EXPR        ; GET STRING
2DD4  E5        1740           PUSH HL
2DD5  CD6A21    1750           CALL ASCO        ; GET ADDRESS IN DE
2DD8  2B        1760           DEC HL           ; FIND LENGTH
2DD9  2B        1770           DEC HL
2DDA  2B        1780           DEC HL
2DDB  7E        1790           LD A, (HL)       ; GET LENGTH IN A
2DDC  FE5A      1800           CP 90            ; IS IT TOO LONG?
2DDE  3805      1810           JR C, EVAL2      ; IF NO, JUMP
2DE0  1E0F      1820           LD E, 15         ; IF YES, WRITE
2DE2  C31913    1830           JP ERROR         ; ERROR MESSAGE
2DE5  4F        1840  EVAL2    LD C, A
2DE6  0600      1850           LD B, 0
2DE8  EB        1860           EX DE, HL
2DE9  11D50C    1870           LD DE, BUFFER
2DEC  EDB0      1880           LDIR
2DEE  23        1890           INC HL
2DEF  3600      1900           LD (HL), 0
2DF1  21D50C    1910           LD HL, BUFFER    ; COMPRESS OVER
2DF4  CD4914    1920           CALL CMPRSS      ; ITSELF
2DF7  23        1930           INC HL
2DF8  CD771B    1940           CALL EXNMCK      ; EVALUATE EXPR.
2DFB  C3AA2B    1950           JP FNEND         ; BACK TO BASIC
2DFE  00        1960  CONV       DEFS 1         ; CONVERSION FLAG
                1965  ;
                1970  ; ROUTINES IN CRYSTAL BASIC
                1980  ;
2DFE  154C      1990  TSTCOM   EQU 154CH        ; TEST FOR COMMA
2DFE  2250      2000  IN255      EQU 2250H      ; GET NUMBER 0 – 255
2DFE  1FAB      2010  ASNSTR   EQU 1FABH        ; CREATE NEW STRING
2DFE  1FD9      2020  STREND   EQU 1FD9H        ; RETURN WITH RESULT
2DFE  1551      2030  TSTCHR   EQU 1551H        ; TEST FOR NEXT BYTE
2DFE  215B      2040  LEN1     EQU 215BH        ; GET STRING LENGTH
2DFE  1B8B      2050  EXPR     EQU 1B8BH        ; EVALUATE EXPRESSION
2DFE  216A      2060  ASCO     EQU 216AH        ; GET ADDRESS & LENGTH
2DFE  1449      2070  CMPRSS   EQU 1449H        ; COMPRESS TO BUFFER
2DFE  2BAA      2080  FNEND    EQU 2BAAH        ; RETURN FROM FUNCTION
2DFE  1B77      2090  EXNMCK   EQU 1B77H        ; EVALUATE NUMERIC EXP.
2DFE  0CD5      2100  BUFFER   EQU 0CD5H        ; BASIC INPUT BUFFER
2DFE  1319      2110  ERROR    EQU 1319H        ; PRINT ERROR MESSAGE
```

These routines should can be entered into the crystal basic interpreter in the normal way, or they can be added by means of the loader program given in the last issue. The code to be entered when using this program is given below.

NAME:OLD

```
0000   FD  2A  83  12  FD  E5  DD  E1  AF  06  00  11  04  00
```

000E	FD	19	FD	BE	00	28	04	FD	23	18	F7	78	B7	20
001C	0D	06	01	FD	23	FD	E5	D1	DD	73	00	D	72	01
002A	AF	FD	BE	01	20	E5	FD	BE	02	20	E0	11	03	00
0038	FD	19	FD	22	B7	0C	FD	22	BB	0C	C9	.		

NAME:STRING$(

0000	E1	23	CD	50	22	F5	CD	4C	15	CD	50	22	F5	CD
000E	51	15	29	F1	47	F1	E5	4F	79	C5	CD	AB	1F	C1
001C	EB	70	23	0D	20	FB	C22D9		1F	.				

NAME:LOWER$(

0000	3E	4C	32	+0096	18	0C

NAME:UPPER$

0000	3E	55	32	+008F	18	05

NAME:REVERSS$(

0000	3E	52	32	+0088		E1	23	CD	8B	1B	CD	51	15	29
000E	E5	CD	6A	21	2B	2B	2B	46	EB	E5	C5	78	CD	AB
001C	1F	C1	E1	3A	+0088		FE	52	28	24	FE	4C	20	0D
002A	7E	FE	41	38	13	FE	5B	30	0F	C6	20	18	0B	7E
003B	FE	61	38	06	FE	7B	30	02	D6	20	12	13	23	10
0046	D8	C3	D9	1F	48	06	00	09	2B	41	7E	12	13	2B
0054	10	FA	C3	D9	1F	.								

NAME: EVAL(

0000	E1	23	CD	8B	1B	E5	CD	67	21	2B	2B	2B	7E	FE
000E	5A	38	05	1E	0F	C3	19	13	4F	06	00	EB	11	D5
001C	0C	ED	B0	23	36	00	21	D5	0C	CD	49	14	23	CD
000A	77	1B	C3	AA	2B	4C	00	00	00	.				

\- - - - - - - - - - - - - -

In the last issue a section of code was omitted from the command loader listing on page 13. The following text should be added between line 850 and line 1020

```
4EA5   FD7500     0850           LD (IY), L
4EA8   FD7401     0860           LD (IY+1),H
4EAB   FD23       0870           INC IY
4EAD   FD23       0880           INC IY
4EAF   18D9       0890           JR IN1
4EB1   CDCD4E     0900   IN2     CALL NUM8
4EB4   FD7100     0910           LD (IY), C
4EB7   FD23       0920           INC IY
4EB9   18CF       0930           JR IN1
4EBB   E5         0940   NUM6    PUSH HL
4EBC   DF64       0950           SCAL NUM
4EBE   E1         0960           POP HL
4EBF   381C       0970           JR C, ERROR
4EC1   ED4B210C   0980           LD BC, (NUMV)
4EC5   3A200C     0990           LD A, (NUMV)
4EC8   FE04       1000           CP 4
4ECA   20 11      1010           JR NZ, ERROR
```

* * * * * * * * * * * * * *

EPROM PROGRAMMER/CHECKER/READER

Part 2

By C. Bowden

The previous article described an EPROM programmer that I recently designed and built. The present article gives the circuit diagrams of the programmer and a suitable power supply, together with a Veroboard layout for the programmer and the source code of the software necessary for its operation.

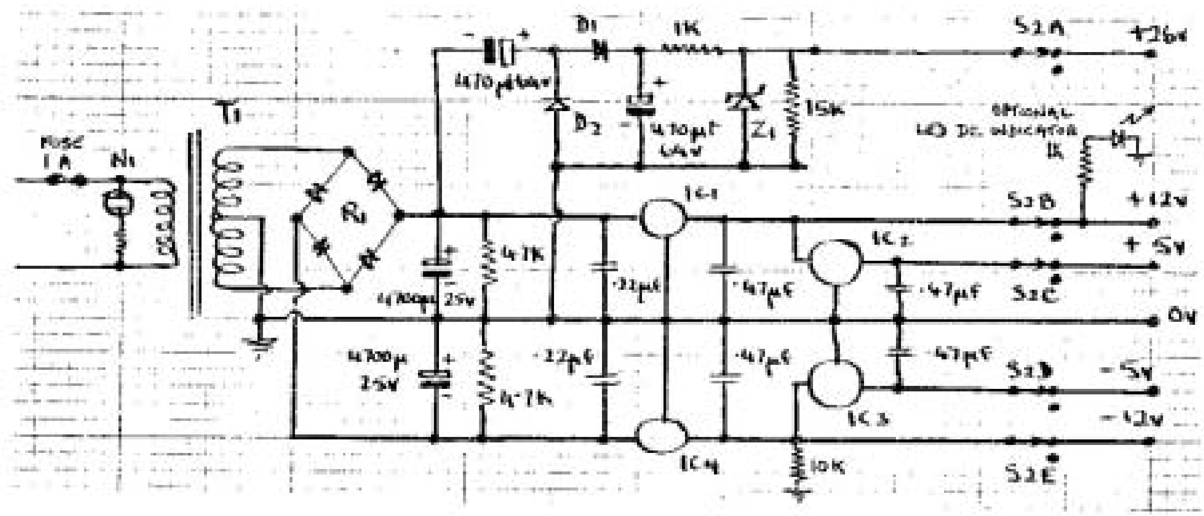


Fig. 0   PROGRAMMER POWER SUPPLY

```
;***************************************
;*                                     *
;*   EPROM PROGRAMMER FOR 2708/2716    *
;*                                     *
;***************************************
;
;    FOR NASCOM 2 WITH 4 MHZ CLOCK.
;
;          (NO WAIT STATES)
;
;
;             WRITTEN FOR MACRO 80 ASSEMBLER.
;
;   WILL NEED TO BE MODIFIED FOR ZEAP ASSEMBLER.
;
;        C.BOWDEN                    NOV.  1981
;
;***************************************
;
;        EQUATES:-
```

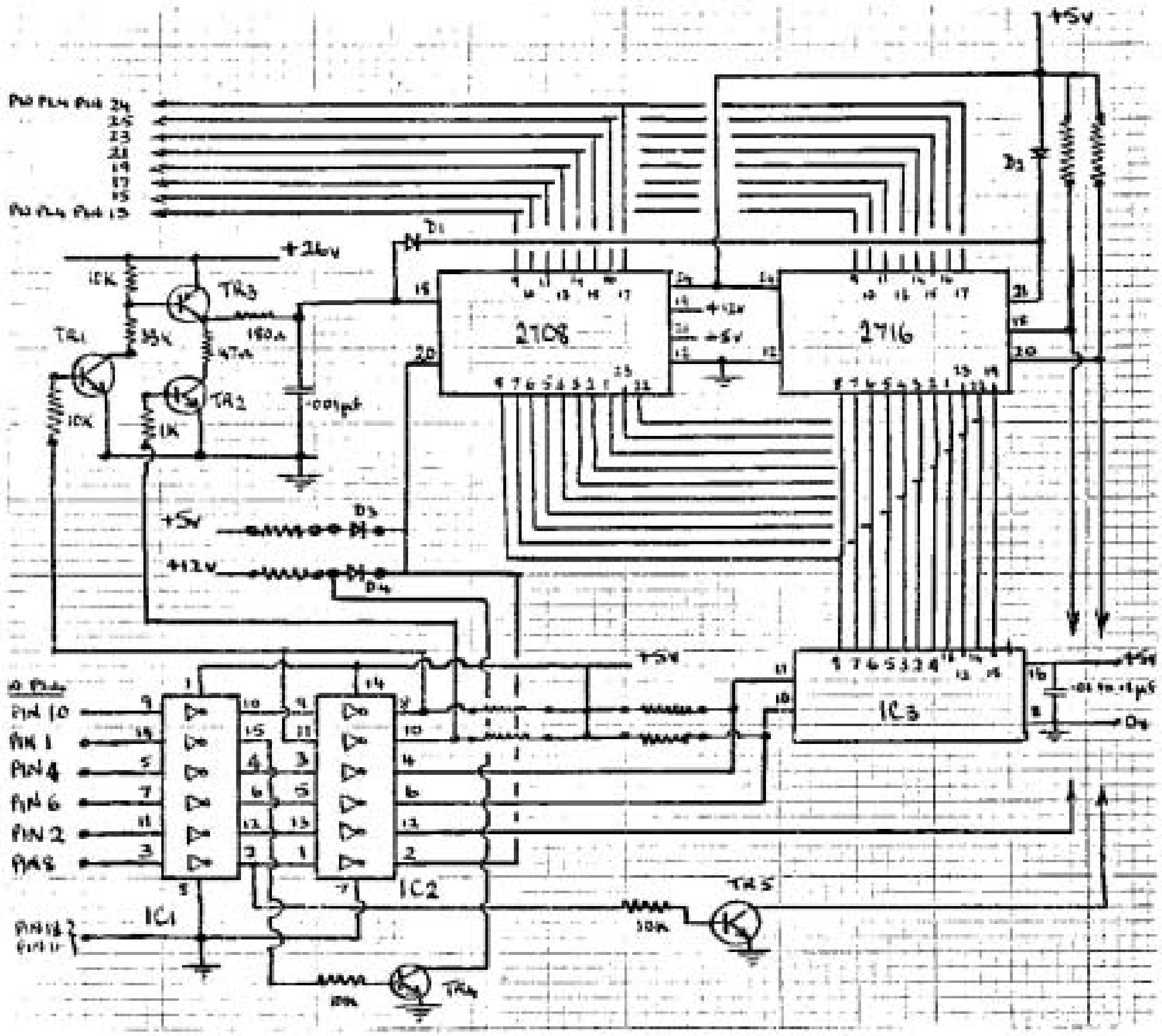


Fig. 1   2708 PROGRAMMER CIRCUIT

NOTES: 1) PORT A OF PIO USED FOR DATA, PORT B FOR CONTROL
2) DECOUPLING CAPACITORS 0.01 – 0.1 uf SHOULD BE USED ON EACH IC (AS ON IC 3)
3) THE PIN NUMBERS SHOWN REFER TO THE NASCOM 2 26-WAY PIO CONNECTOR
4) OBSERVE CMOS HANDLING PRECAUTIONS WITH ICs 1 AND 3, AND THE EPROMS
5) IF POSSIBLE USE ZERO INSERTION FORCE SOCKETS FOR THE EPROMS
6) UNMARKED PULL UP RESISTORS CAN BE IN THE RANGE 4.7K TO 10K

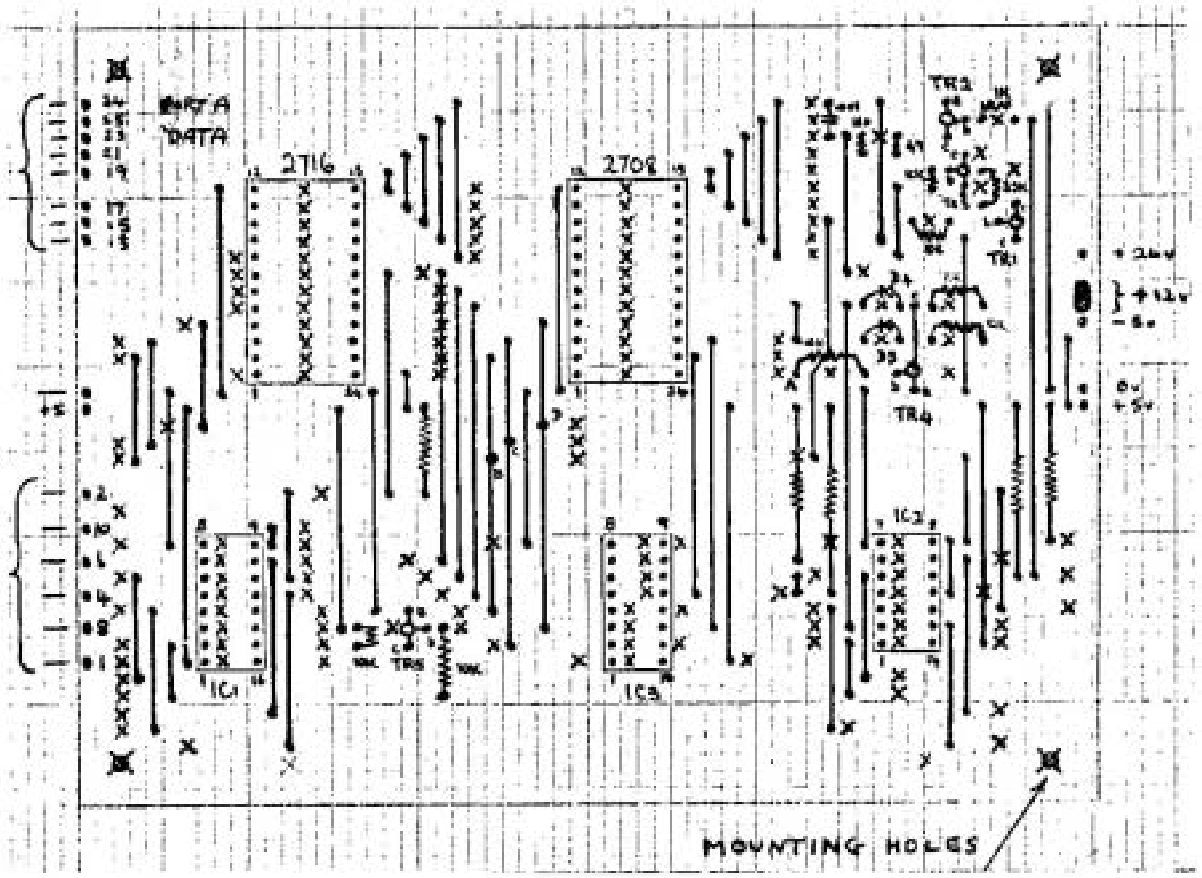


Fig. 2 A SUITABLE VERO BOARD LAYOUT

NOTES: 1) THE LAYOUT IS SHOWN FROM THE COPPER SIDE. ALL COMPONENTS EXCEPT THE LINKS AND 2 DIODES NOTED BELOW ARE MOUNTED ON THE OTHER SIDE OF THE BOARD

2) IF ZERO INSERTION FORCE SOCKETS ARE USED MOUNT ALL LINKS NEAR THEM FIRST AS THEY WILL COVER THE HOLES

3) x SIGNIFIES A TRACK CUT THROUGH, • IS A SOLDERED CONNECTION, LINES SHOWN AS •————• ARE TINNED COPPER LINKS, THREE OF WHICH ARE TAPPED PART WAY ALONG THEIR LENGTH (B, C AND D)

4) DECOUPLING CAPACITORS ARE NOT SHOWN, BUT 0.01 TO 0.1 µF ARE RECOMMENDED (1 PER IC)

5) THE FOLLOWING INSULATED WIRE LINKS, MOUNTED ON THE COPPER SIDE OF THE BOARD, ARE NOT SHOWN:

IC1 PIN 15 TO 10KOhm A	IC3 PIN 9 TO 2708 PIN 8
TR5 COLL. TO 2716 PIN 20	IC3 PIN 13 TO 2708 PIN 1
IC3 PIN 15 TO 2716 PIN 19	

CONNECT THE 2708 PINS 1,4,5,6,7,8,9, 10, 11, 22 & 23 TO THE CORRESPONDING PINS ON THE 2716 SOCKET

6) A 1N4148 DIODE IS CONNECTED FROM +5V TO PIN 21, 2716 (CATHODE TO PIN21) AND ANOTHER 1N4148 FROM PIN 21, 2716 TO PIN 18, 2708 (CATHODE TO PIN 21)

```
;
LF          EQU 0AH
CR          EQU 0DH
KBD         EQU 62H
B2HEX       EQU 68H
CRLF        EQU 6AH
TDEL        EQU 5DH              ; ONE SECOND DELAY
SRLX        EQU 6FH
SCAL        EQU 0DFH
ADATA       EQU 14H
ACTRL       EQU 16H
BDATA       EQU 15H
BCTRL       EQU 17H
HSHAKE      EQU 00H
;
            ASEG
            .Z80
;
            ORG 100H
            .PHASE 0A000H
;
; ----------------------------------------------------------------
START:      LD HL, TEXT1         ; TITLE TO TOP LINE
            LD DE, 0BCBH
            LD BC, 10H
            LDIR
            CALL STPIO1          ; PORT A TO I/P, B TO O/P
            CALL RESET1          ; RESET COUNTER, CHIP OFF
            JR RESTR1
;
RESTRT:     CALL CLRCRT          ; RETURN HERE AFTER A ROUTINE
REMOVE      LD HL, TEXT3         ; MESSAGE ON EPROM REMOVAL
            LD HL, 0A0BH
            LD BC, 41H
            LDIR
            LD HL, TXT17A        ; “SAME EPROM - ?”
            LD DE, 0B0BH
            LD BC, 17
            LDIR
ANSWER:     DEFB SCAL, KBD       ; SCAL KEYBOARD FOR REPLY
ANS1:       JR NC, ANSWER
            CP “Y”
            JR Z, PROMPT
            CP “N”
            JR NZ, ANSWER        ; TRY AGAIN IF NOT Y OR N
            JP RESTR2
;
RESTR1:     CALL CLRCRT          ; CAUTION MESSAGES FOR 1ST RUN
            LD HL, TEXT20
            LD DE, 090BH
            LD BC, 34
            LDIR
            LD HL, TEXT21
            LD HL, 098BH
            LD BC, 38
            LDIR
            DEFB SCAL, TDEL      ; WAIT APPROXIMATELY 2 SECONDS
            DEFB SCAL, TDEL
```

```
        ;
        RESTR2:     CALL CLRCRT          ; CLEAR TYPE NUMBER
        CLRTYP:     LD B, 16             ; TYPE ADDRESS ON SCREEN
                    LD HL, 0BE8H         ; SPACE CHARACTER
                    LD A, " "
        CLRT1:      LD (HL), A
                    INC HL
                    DJNZ CLRT1
        KEYC:       LD HL, TEXT3         ; EPROM HANDLING MESSAGE
                    LD DE, 0A0BH
                    LD BC, 41
                    LDIR
                    LD HL, TEXT4         ; PROMPT FOR KEY "C"
                    LD DE, 0B0BH
                    LD BC, 19
                    LDIR
        SCAN:       DEFB SCAL, KBD       ; GET INPUT
                    JR C, SCAN1
                    JR SCAN
        SCAN1:      CP "C"
                    JR NZ, SCAN          ; LOOP UNTIL "C" IS PRESSED
        ALLOK:      CALL CLRCRT
                    LD HL, TEXT1A        ; "KEY A---FOR 2708"
                    LD DE, 090BH
                    LD BC, 33
                    LDIR
                    LD HL, TEXT1B        ; "KEY B--------"
                    LD DE, 0991H
                    LD BC, 27
                    LDIR
        KEY:        DEFB SCAL, KBD       ; GET INPUT
                    JR C, KEY1
                    JR KEY
        KEY1:       PUSH AF              ; SAVE KEY
                    CALL CLRCRT          ; CLEAR SCREEN
                    LD HL, TEXT11        ; "KEY ?-----Y/N"
                    LD DE, 090BH
                    LD BC, 27
                    LDIR
                    POP AF
                    LD HL, 090FH         ; SCREEN ADDRESS FOR KEY
                    LD (HL),A            ; PRINT IT
                    EX AF, AF'           ; SAVE IT AGAIN
        KEY2:       DEFB SCAL, KBD       ; GET KEY AGAIN
                    JR NC, KEY2
                    CP "Y"
                    JR Z, TYPE
                    CP "N"
                    JR Z, ALLOK
                    JR KEY2              ; TRY AGAIN IF NOT Y OR N
        ;
        TYPE:       EX AF, AF'           ; GET ORIGINAL ENTRY BACK
                    CP "A"
                    JRZ, TYP1K           ; 1K EPROM
                    CP "B"
                    JR Z, TYP2K          ; 2K EPROM
                    CALL CLRCRT
```

```
TYPERR:     LD HL, TEXT12          ; TYPE ---- A OR B"
            LD DE, 090BH
            LD BC, 29
            LDIR
            DEFB SCAL, TDEL        ; WAIT APPROXIMATELY 2 SECONDS
            DEFB SCAL, TDEL
            JP ALLOK
TYP1K:      LD A, 4                ; FLAG FOR 2708
            LD (ROMFLG), A)
            LD HL, TXT13A          ; TYPE TO TOPLINE
            LD DE, 0BE8H
            LD BC, 16
            LDIR
            JR PROMPT 150:         ; FLAG FOR 2K EPROM
            LD (ROMFLG), A
            LD HL, TXT13B          ; TYPE TO TOPLINE
            LD DE, 0BE8H
            LD BC, 16
            LDIR
;
PROMPT:     CALL CLRCRT
            LD HL, TEXT2           ; PROMPT FOR KEY P----.
            LD DE, 094BH
            LD BC, 25
            LDIR
            LD HL, TEXT2A          ; KEY C----
            LD DE, 09D2H
            LD BC, 1EH
            LDIR
            LD HL, TEXT2B          ; KEY T----
            LD DE, 0AD2H
            LD BC, 1FH
            LDIR
            LD HL, TEXT2C          ; KEY E----
            LD DE, 0AD2H
            LD BC, 20H
            LDIR
            LD HL, TEXT2D          ; KEY D----
            LD DE, 0B52H
            LD BC, 1AH
            LDIR
;
OPTION:     XOR A                  ; WHICH ROUTINE
            DEFB SCAL, KBD         ; SEE IF KEY PRESSED
            JR C, WHICH            ; IF SO, JUMP TO WHICH
            JR OPTION              ; ELSE KEEP LOOKING FOR KEY
WHICH:      EX AF, AF'             ; SAVE KEY
            CALL CLRCRT
            LD HL, TEXT11          ; KEY ? ---Y/N?
            LD DE, 090BH
            LD BC, 27
            LDIR
            LD HL, 090FH           ; SCREEN ADDRESS
            EX AF, AF'             ; RECOVER A
            LD (HL), A             ; PRINT IT
            EX AF, AF'             ; RESAVE IT
WH1:        DEFB SCAL, KBD
            JR NC, WH1
```

```
                CP "Y"
                JR Z, WH2
                CP "N"
                JR NZ, WH1          ; TRTY AGAIN IF NOT Y OR
                JR PROMPT           ; REPROMPT IF "N"
WH2:            CALL CLRCRT         ; CLEAR SCREEN
                EX AF, AF'
                CP "P"              ; WAS IT A "P"?
                JR Z, PROGRM        ; IF SO, GO TO PROGRAM
                CP "C"              ; WAS IT A "C"?
                JR Z, CMPARE        ; IF SO, GO TO COMPARE
                CP "T"              ; WAS IT A "T"?
                JR Z, TRNFER        ; IF SO, GO TO TRANSFER
                CP "E"              ; WAS IT AN "E"?
                JR Z, ERASED        ; IF SO, TEST FOR ERASURE
                CP "D"              ; WAS IT A "D"?
                JR Z, OUTPUT        ; IF SO, GO TO DUMP ROUTINE
                JR OPTION           ; INVALID ENTRY, TRY AGAIN
; *******************************
; *   ROUTINE TO PROGRAM EPROM      *
; *******************************
PROGRM:         LD HL, TEXT17       ; "COPIED INTO EPROM"
                LD DE, 0A4BH
                LD BC, 11H
                LDIR
                CALL RAMADR         ; GET START ADDRESS
SETPIO:         CALL STPIO3         ; PORT A LINES TO O/P
                LD A, (ROMFLG)      ; 1K OR 2K EPROM?
                CP 8
                JR Z, PR2716        ; JUMP IF 2716
                LD B, 68H           ; NUMBER OF CYCLES TO COUNT
PROGR1:         PUSH BC             ; SAVE CYCLES COUNT
                CALL RESET          ; RESET COUNTER, SET 12V AND WE
                LD DE, 0            ; CLEAR COUNTER
                LD HL, (STOR1)      ; GET START ADDRESS
PROGR2:         LD A, (ROMFLG)      ; FLAG EQUALS 4 FOR 2708
                CP D                ; IS D 4?
                JR Z, PROGR3        ; IF SO, JUMP TO PROGR3
                LD A, (HL)          ; ELSE GET BYTE AND
                OUT (ADATA), A      ; SEND IT TO THE EPROM
                LD B, 10H           ; SHORT DELAY TO ALLOW
HOLD:           DJNZ HOLD           ; DATA TO STABILISE
                LD A, 23H           ; OUTPUT CONTROLS TO PUT
                OUT (BDATA), A      ; 12V AND 26V ON EPROM
                LD B, OEH
DELAY1:         NOP                 ; KEEP 26V ON FOR 1 mS
                DJNZ DELAY1
                LD A, 22H           ; NOW SWITCH 26v OFF
                OUT (BDATA), A      ; BUT KEEP 12V ON
                LD B, 10H           ; SHORT DELAY
HOLD1:          DJNZ HOLD1
                LD A, 26H           ; HOLD 12V AND INCREMENT
:               OUT (BDATA), A      ; ADDRESS COUNTER
                LD B, 10H           ; SHORT DELAY
HOLD2:          DJNZ HOLD2
                LD A, 22H           ; 12V ONLY
                OUT (BDATA), A
```

```
                INC HL                  ; POINT TO NEXT BYTE
                INC DE                  ; INCREMENT BYTE COUNT
                JR PROGR2               ; BACK, SEE IF 1K DONE
PROGR3:         POP BC                  ; 1K DONE, GET CYCLES
                LD A, B                 ; PRINT CYCLES LEFT
                DEC A                   ; ADJUST COUNT
                DEFB SCAL, B2HEX        ; PUT ON SCREEN
                DJNZ PROG1              ; IF NOT 104, GO TO PROGR1
PROGR4:         CALL STPIO2             ; PORT A TO I/P
                CALL RESET1             ; 12V OFF, 5V ENABLED, CHIPS OFF
                DEFB SCAL, TDEL         ; TWO SECONDS DELAY
                DEFB SCAL, TDEL
                CALL MESS19
                JP RESTRT               ; BACK TO START
;
;**********************************
;*      ROUTINE FOR 2516/2716 EPROMS         *
;**********************************
;
PR2716:         LD DE, 0                ; BYTE COUNTER
                JR PRINTD
PR27A:          LD A, (ROMFLG)          ; FLAG EQUALS 8 FOR 2K ROM
                CP D                    ; 800H BYTES DONE?
                JR Z, PROGR4            ; IF SO, EXIT VIA PROGR4
                LD A, 3                 ;TURN ON 26V AND OE
                OUT (BDATA), A
                LD B, 10H               ; SHORT DELAY
WAIT:           DJNZ WAIT
                LD A, (HL)              ; DATA
                OUT (ADATA), A
                LD B, 10H
WAIT1:          DJNZ WAIT1              ; SHORT DELAY
                LD A, 13H
                OUT (ADATA), A          ; TURN ON 26V, OE, PGM
                LD BC, 1D00H            ; COUNT FOR 50 mSEC.
PR27C:          DEC BC
                LD A, B
                OR C
                JR NZ, PR27C
                LD A, 3                 ; PGM PULSE OFF
                OUT (BDATA) ,A
                LD B, 10H WAIT2
WAIT2:          DJNZ WAIT2              ; SHORT DELAY
                LD A, 1
                OUT (BDATA), 1          ; TURN OE OFF
                LD B, 10H
WAIT3:          DJNZ WAIT3              ; SHORT DELAY
                LD A, 5
                OUT (BDATA), A          ; 26V, INCREMENT COUNTER
                LD B, 10H
WAIT4:          DJNZ WAIT4              ; SHORT DELAY
                LD A, 1
                OUT (BDATA), A          ; INCREMENT PULSE OFF
                INC HL
                INC DE
                LD A, E
                CP 0
```

```
                JR Z, PRINTD
                JP PR27A
PRINTD:         LD A, D
                DEFB SCAL, B2HEX     ; O/P COUNT TO SCREEN
                DEFB SCAL, CRLF
                JP PR27A
;
; **************************************
; *  ROUTINE TO COMPARE EPROM/1K BLOCK  *
; **************************************
CMPARE:         LD HL, TEXT8         ; “COMPARED TO EPROM
                LD DE, 0A4BH
                LD BC, 11H
                LDIR
                LD A, 0
                LD (ERRFLG), A       ; SET ERROR FLAG
                CALL RAMADR          ; GET START ADDRESS
                LD DE, 0             ; CLEAR BYTE COUNTER
VERFY2:         LD A, (ROMFLG)
                CP D                 ; ALL DONE?
                JR Z, VERFY4         ; IF SO, JUMP TO VERFY4
                CALL ENABLE          ; TRUN ON CHIP
                IN A, (ADATA)        ; GET BYTE FROM EPROM
                PUSH AF              ; SAVE IT
                CALL COUNT           ; INCREMENT ADDRESS, CHIP OFF
                POP AF;              ; GET BYTE BACK
                CP (HL)              ; ARE THEY THE SAME?
                JR NZ, ERROR         ; IF ERROR, PRINT IT
VERFY3:         INC HL               ; NEXT BYTE TO COMPARE
                INC DE               ; INCREMENT BYTE COUNTER
                JR VERFY2
ERROR:          LD A, 0FFH           ; ERROR FLAG
                LD (ERRFLG), A
                LD A, H              ; PRINT ERROR ADDRESS
                DEFB SCAL, B2HEX
                LD A, L
                DEFB SCAL, B2HEX
;               DEFB SCAL. TDEL      ; OPTIONAL DELAY
; IF DELAY REQUIRED, REMOVE FIRST COLON ON ABOVE LINE
                DEFB SCAL, CRLF      ; SCROLL DISPLAY
                JR VERFY3            ; NEXT BYTE
```

The remainder of the listing will be published in the next issue. If you don’t wish to wait that long, or you can’t be bothered to type it all in – after all the source code is 18K long – I will provide copies of the source and object codes. Just send me a cassette, or a disc with approximately 26K free, and £1 to cover copying and postage. Source can be supplied in Zeap compatible form or Macro 80 form. Please state which you require. Also please state the disc format required; I can supply single or double density to suit either CP/M or Polydos, for Nascom/Gemini G805/G809

C. Bowden, ‘Tregwyn’, Stithians, Truro, Cornwall

ALTERNATIVE GRAPHICS EPROMS

Two specially designed sets of pixels for Nasom-type minicomputers. 8x16 format programmed EPROMS giving an enhanced display when substituted for the Nascom 2 ROM.

EPROM No. 1 — REVERSE (Black on white ground) NUMERALS & ALPHABET (Upper Case)
HIGH RESOLUTION Horizontal & Vertical lines
BAR GRAPH FACILITIES Solid, chequered or lines to plot high resolution horizontal or vertical bargraphs (BASIC software listings supplied)

EPROM No. 2 — GAMES SHAPES:

- MILITARY - Tanks, planes, helicopters, minisubmarines
- SPACE - Aliens, mothership, laser-base, rockets and missiles
- DUNGEONS - Anubis, Devil, dwarf, men, key/keyhole, treasure pile
- VARIOUS - Galleys, race-cars

Both EPROMS are ideal for use with a graphics ROM adaptor. Price: £13.50 per EPROM (inclusive) S.A.E. for details.

UNICORN AND MORTAR, 11, FREETH ROAD, BROWNHILLS, WALSALL.

LETTERS

Dear Sir,

Re the "Nascom 1 Keyboard Upgrade", an interesting article currently under construction, readers may be interested to know that Licon Keys (the later angled variety) are available from Target Electronics, 16, Cherry Lane, Bristol, BS1 3NG. Prices in December 1981 were; Keyswitch 65p each., Keytop 17p each.

My only criticism of the magazine "so far" is that the density of the type varies considerably, some is difficult to read.

On expanding my Nascom 1 to include a 64K RAM card, EPROM expansion card with 8K Basic running under Nasbug T4 I encountered a hardware problem. Executing FOR loops with large values, i.e., greater than 1000, produced "crashes"; Basic programs occasionally jump to machine code, and vice versa. After considerable difficulty I tried connecting a thick lead (20 A current capability) directly from the Z80 earth pin to 0 V on the Buffer board. To date this has cured all programming problems.

P. E. Acton, Leicester

Dear Sir,

In the program "Rings of Hanoi" on p. 31 of the December issue of Micropower, I think the followinq corrections are needed:-

```
220 A (I, 0) = I:A(I ,1) = 0: A (I, 2)=0
290 SCREEN 1,1: PRINT : SCREEN 6,15
400  GOSUB  920: SCREEN  2,2:PRINT: SCREEN 2,2: PRINT   "MOVE
FROM";
410 INPUT IN$
420 GOSUB 1300:TF=IN:IF TF < 0 THEN 400
440 SCREEN 1,1:PRINT:SCREEN1,1
470 GOSUB 920:SCREEN 2,2:PRINT: SCREEN 2,2: PRINT "TO PILE";
```

F. Johnson, Stockport

Dear Sir,

The content of Micropower is quite good and the technical level is about right, I think the hardware/software mix is probably O.K., although I would personally like to see more programs in each issue.

I have managed to make the "Snowdinger" mod work at 2 Mhz and 4 Mhz, although I have had to delay the WAIT signal by two gate propagations in order to get the system to operate at 2 Mhz. The improvement in screen quality is excellent.

The dual monitor (2716 for 2708s) was nearly correct, though it is not permissible to wire ‘or’ the two outputs of a 74LS139 - it is necessary to combine the signals correctly by using, for example, a 7402.

D. R. Piercy, Wareham

Dear Editor,

The contents of the magazine seem quite well balanced - there must be something in each issue of interest to every Nascom user. I would like to see some explanation of essential parts of programmes, i.e. brief notes on those parts of software on which the operation depends. For example, a note on how the high-res graph plotting software works would have been interesting. I would be particularly interested in articles on fault finding techniques for or with the Nascom. If anyone knows of a cheap and reliable MODEM design then the details would be most welcome.

The Lincoln computer club now meets on the 1st and 3rd Wednesday of each month at the Hare and Hounds, 330, High Street, Lincoln, starting at 19.30. Further details from:

John Clifford, Lincoln (0522) 21607

MICRO=MARKET

Small non-commercial advertisments, £2 per ad.

FOR SALE Two ICL Termiprinter units Type 7572
Both fully over hauled and in full working order. 118 char./line. Prints at 60 c.p.s. Both fitted with Horizontal and Vertical Tab facilities and Form Feed mechanism. RS-232 Serial interface will fit directly to Nascom 1 or 2. Baud rate 110, 300, 1200 and others. Tractor and friction feed. Service documentation available. Software supplied for Nas-Sys 1 or 3. Fully gauranteed for 3 months.
Offers over £200 for each
IBM Selectric Golfball printer with Acculab Professional interface unit in full working condition. Parallel interface fits directly to Nascom PIO. Selection of Golfball typeheads available. Tractor and friction feed. Software Subroutine supplied for Nas-Sys 1 or 3. Best offer over £300 for the pair.

For further information on any of the above, phone Wolverhampton (0902) 893037. 10 a.m. to 10 p.m. Prefer buyer to colect and see full demonstration, but carriage could be arranged.

FOR SALE Solenoid Operated Cassette Transports.
These machines may be put into wind, rewind, stop and play with signals from the PIO, buffered with reed ralays or open-collector gates. Record tabs, cassette present and tape end can similarly be fed back to the PIO.

It takes a little imagination and a bit of programming to turn it into an automatic program and data store, putting back the fateful day when you have to buy discs. All this for only £25 + carriage. More details from Pat Crabb, Holmfirth (0484 489) 5263

FOR SALE	RAM A card with 8K	£45 o.n.o.
	RAM B card with 16K	£70 o.n.o.
	Both in full working order	C. Bowden

Tel. 0209 860 480 (Evenings) 0209 712 780 (Daytime)

THE NAS-SYS MONITORS

By J. Haigh

THE EXTERNAL COMMAND X xx

When this command is executed, the low byte of the argument after the X is stored at £0C28, and the pointers to the input and output command tables at £0C75 and £0C73 are reset. Input now scans a routine called XKBD (SCAL £74) and then scans the ‘normal’ Nascom keyboard routine, which in the case of Nas-Sys 1 is SCAL £61 while Nas-Sys 3 uses the Repeat Keyboard routine SCAL £7D. Output sends data to an external output routine XOUT (SCAL £6E), then to the user output (SCAL £75) and finally to the CRT (SCAL £65).

The external keyboard routine scans the serial input port. If no input is received it returns from this routine and scans the Nascom keyboard in the normal way – any options set under the X command have no effect on the Nascom keyboard, so to the user the operation of the system is unchanged.

If an input is received from the serial input port, bit 7 of the input byte is first set to zero. This is done because many systems use the most significant bit for parity checking to eliminate errors; the number of ones in the lower seven bits is counted and the value of bit 7 is then adjusted so that the number of ones in the byte is either always even (even parity) or always odd (guess what that’s called). The Nascom may have to communicate with systems which use either (or no) system of parity checking, and it does this the easy way by merely stripping off the parity bit and ignoring it,

The value of the byte entered as argument to the X command is now used to control various options available, If bit 5 of the byte was zero, the input routine 'echos' each byte received from the external keyboard, that is, it transmits it back to the sender through the serial port; if bit 5 is set, echo is suppressed. Nas-Sys 3 also tests bit 1 of the option byte; if this bit is zero, any echo of an input byte by the user program is suppressed. This is done by using bit 7 of the option byte as a "suppression flag". The flag is tested on output to determine whether to transmit a byte or not, and is then reset. Nas-Sys 1 does not use bit 1 of the option byte

The external output routine tests the parity of the byte to be transmitted; if it is odd it inverts bit 7. Bit 0 of the option byte is now tested; if bit 0 is set, this tells the Nascom to transmit in 'odd parity' format, so it re-inverts bit 7. The byte is now sent out through the serial port. Although you can ignore the parity of the received data, you will usually be communicating with a device which tests parity and must therefore conform to its requirements.

For the same reason, a line feed (code £0A) is output after a carriage return (£0D), as most systems need both signals; however, the line feed can be suppressed by setting bit 4 of the option byte. The Nas-Sys 1 external output routine ignores nulls, but Nas-Sys 3 has been modified so that nulls are output correctly.

The many options available make this a very powerful command. Because the user output routine is automatically brought into operation when the X command is invoked, you can have a parallel printer on line in addition to the Nascom keyboard and an ASCII terminal (for example, a teletype keyboard and printer). Of course, you must set up the user output routine by storing the address at £0C78 in the usual way.

YJUMP Y

In Nas-Sys 1, Y produces an error message - the address in the subroutine table is £030A. Nas-Sys 3 uses Y to jump to £B000; this will normally be used to access software in an EPROM at this address, such as the Basic "Programmer's Aid" or an extension to the monitor. I have the initialisation routine for my printer here, so that Y configures the PIO ports, clears the print buffer and resets the printer options.

ZJUMP Z

This command is normally used to 'warm start' the Microsoft Basic at £FFFD. 0f course, if you haven't got the Basic in ROM you can use the command to access other software by changing the address stored for the Z command. I use it to access Zeap, using Z for a warm start and ZC for a cold start. Because C is a valid hexadecimal number, you can use it as an argument to a command. The software at the start of the command tests the value in the L register; if it is £0C, it does a jump to the cold start address, otherwise it does a warm start. This has two advantages. Firstly, it gives you two commands for the price of one - when you start modifying your software you soon run out of command letters. Secondly, it reduces the chance of performing a cold start when you meant a warm start; it still happens, but not as often. There always seems to be room in the software you are accessing for the extra code needed to test the value of ARG1.

However, there is a snag. I tried the same ystem for Basic, using J for a warm and JC for a cold start. I then found that I couldn't RUN programs that appeared to CLOAD correctly. On LISTing garbage was displayed. The reason was that the argument C was used as an offset in the Nas-Sys 3. READ routine used by CLOAD - the program was being read in to an address twelve bytes higher than the correct address.

The solution is simple - after testing the argument the routine should reset ARG1 to zero. Alternatively, you can POKE the value to zero from Basic.

That brings us to the end of the Nas-Sys commands, but there are many more addresses in the subroutine call table - 34 in Nas-Sys 1 and 37 in Nas-Sys 3. These SCALs cannot be accessed directly from the keyboard like the command letters. Many of them correspond to lower case letters, but if you try to enter them as commands you will just get an error message, because the routine which accepts commands test the input character to see if it lies in the range A - Z. Of course they are there to be used in programs.

SCAL MRET DF 5B

This is the normal way to return to the monitor from a program. The routine resets the monitor and user stacks, prints the monitor message (-- NAS-SYS 3 --, or whatever you have substituted) on the screen, restores the byte replaced by any breakpoint that has been set, and then waits for an input. The screen is not cleared, and the monitor message will appear at whatever point the cursor was left by the program - for tidyness you should shift the cursor to the left of the screen by a carriage return if it has been moved. If you want to clear the screen on return to the monitor you can use RST 0 (£C7); this will re-initialise the workspace and clear the screen before jumping to MRET.

SCAL SCALJ DF 5C

This enables you to access any Nas-Sys subroutine by storing the subroutine 'number' at ARGC (£0C0A). The routine saves the HL, AF and DE registers, picks up the routine number from ARGC, and then jumps to section of code in the subroutine call restart where the call address is calculated from the subroutine number. Of course, any requirements of the normal call must be met if a subroutine is accessed 'indirectly' by SCALJ; for example, to call the READ command via SCALJ you would have to store £52 ("R") at £0C0A, but you would also have to place £52 at £0C2B, or the routine would only 'verify' the tape.

SCAL TDEL DF 5D

This routine calls the 'delay' £FF restart, RDEL (RST £38, £FF) 512 times. As each RDEL takes rate 2.7 msec. with a clock rate of 4 Mhz, the total delay in TDEL is 1.38 seconds. Obviously, at 2 Mhz the above times are doubled. Registers A and B are both set to zero on return from this routine.

JCAL FFLP DF 5E

This routine sets and then resets output lines in port 0. On entering the routine the accumulator must have the bits corresponding to the lines to be changed set to one. This data is exclusive ORed with the byte at £0C00, which maps the current state of port 0, so that only the selected bits are changed, and output to port 0. The original state of port 0 is then recovered from £0C00 and output to port 0.

SCAL MFLP DF 5F

This routine changes the state of the tape LED, which is controlled by bit 4 of port 0. The LED gives an indication that loading or saving of tape data is proceeding, but if a small relay is connected to the output which drives the LED the signal can be more usefully employed to switch the cassette motor on and off through the 'remote' socket. If you don't use the signal to control a tape motor, and alternative use is to produce "music' by connecting the LED output to a small 80 Ohm speaker. Notes can be produced by flipping bit 4 at different rates. However, there are two spare output lines on port 0, bits 2 and 5, and you can use these to produce sounds by simply adding a simple buffer and loudspeaker. These bits can be flipped by setting the required bit in the accumulator and doing a direct call to £0053 (CD 53 00). Now you are not supposed to use direct calls to Nas-Sys - the whole idea of the subroutine call system is that if changes are made to the monitor software using the monitor does not need to be changed because it does not use absolute addresses. However, providing you realise the full implications of what you are doing, there is no reason why you shouldn't use a direct call this once. The alternative is to write a short piece of machine code to change the necessary bits of port 0.

SCAL ARGS DF 60

This routine loads the contents of the ARG!, ARG2 and ARG3 (£0C0C, £0C0E, £0C10) into HL, DE and BC. As noted previously, when a program is entered by the Execute command, the contents of HL, DE and BC are picked up from the Register save area in the monitor workspace, and so you cannot pass arguments entered under E directly to a program, you must recover the values atored at ARG1 – ARG3, and this can most easily be done by means of SCAL ARGS,

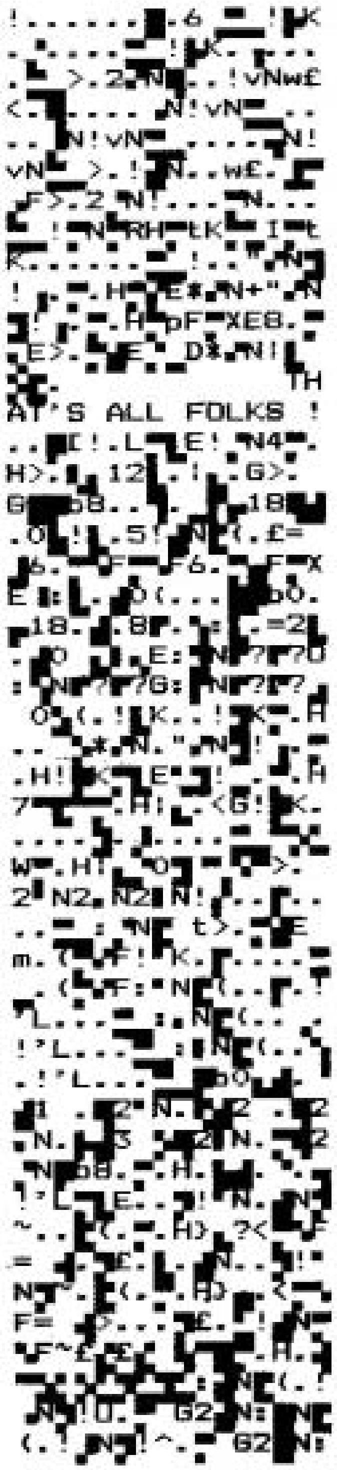
In the next article, I shall continue with the subroutine calls, starting with keyboard calls, DF 61 and DF 62

FRUIT MACHINE

By S. C. Allen

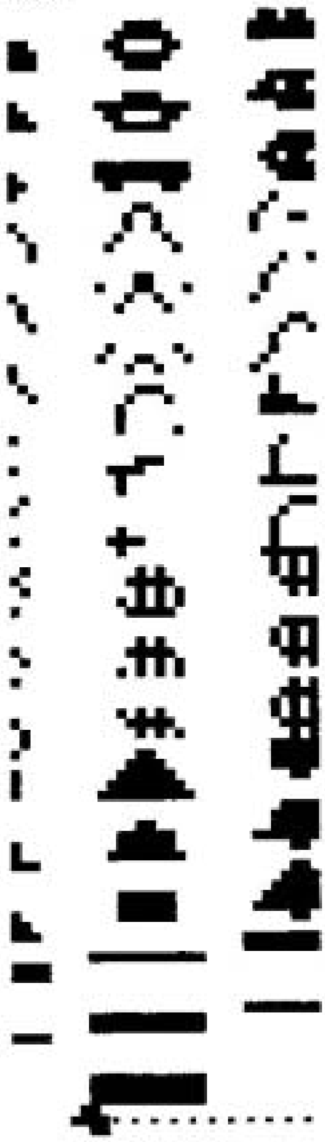
This is a machine code 'One Armed Bandit' simulation, which runs under Nas-Sys 1 or Nas-Sys 3 and uses the standard pixel set to draw the symbols on the reels. To start the game enter E4400; after you have read the brief instructions, press 'Return' to play. The game ends when you have lost all your money – unfortunately it doesn't pay out when you win.

```
4400  21 00 08 11 01 08 01 FF 03 36 A0 ED B0 21 D7 4B
4410  11 CC 0A 01 08 00 ED B0 21 D7 4B 11 F0 0A 01 08
4420  00 ED B0 3E 02 32 86 4E AF 06 10 21 76 4E 77 23
4430  3C 10 FB 01 12 00 11 88 4E 21 76 4E ED B0 01 12
4440  00 11 9A 4E 21 76 4E ED B0 01 12 00 11 AC 4E 21
4450  76 4E ED B0 3E 01 21 BE 4E 06 06 77 23 10 FC CD
4460  98 46 3E 01 32 CC 4E 21 00 08 11 CD 4E 01 00 04
4470  ED B0 21 CD 4E CD 52 48 CD 74 4B CD E5 49 CD 74
4480  4B 11 00 08 01 00 04 ED B0 21 0A 00 22 CA 4E FD
4490  21 F0 0B CD 1D 48 CD CE 45 2A CA 4E 2B 22 CA 4E
44A0  FD 21 F0 0B CD 1D 48 CD 70 46 CD 58 45 38 0B CD
44B0  A4 45 3E 06 CD BE 45 CC E4 44 2A CA 4E 7C B5 20
44C0  D5 EF 0C 20 20 20 20 20 20 20 20 20 20 20 54 48
44D0  41 54 27 53 20 41 4C 4C 20 46 4F 4C 4B 53 20 21
44E0  0D 00 DF 5B 21 1B 4C CD B5 45 21 CC 4E 34 CD 05
44F0  48 3E 07 A5 C6 31 32 E7 0B 7C E6 0F 47 3E 10 80
4500  47 FF DF 62 38 04 10 F9 18 E4 CF D6 31 38 FB FE
4510  03 30 F7 21 E7 0B 35 21 BE 4E B7 28 04 23 3D 20
4520  FC 36 01 CD D1 46 CD D1 46 36 02 CD BC 46 CD 58
4530  45 D8 3A E7 0B FE 30 28 1C 06 00 FF DF 62 30 08
4540  D6 31 38 04 FE 03 38 CB 10 F1 3A E7 0B 3D 32 E7
4550  0B FE 30 20 E6 C3 A4 45 3A BB 4E CB 3F CB 3F 4F
4560  3A A9 4E CB 3F CB 3F 47 3A 97 4E CB 3F CB 3F B8
4570  20 30 B9 28 05 21 B5 4B 18 03 21 BB 4B CD 14 48
4580  16 00 5F D5 2A CA 4E 19 22 CA 4E FD 21 F0 0B CD
4590  1D 48 21 DF 4B CD B5 45 E1 FD 21 E4 0B CD 1D 48
45A0  37 C9 B7 C9 CD 05 48 7C E6 03 3C 47 21 DF 4B 11
45B0  0C 00 19 10 FD 11 DC 0B 01 0C 00 ED B0 C9 D5 E5
45C0  57 CD 05 48 7C E6 7F 92 30 FD 82 E1 D1 C9 3E 01
45D0  32 C1 4E 32 C2 4E 32 C3 4E 21 96 0B 11 97 0B 01
45E0  16 00 ED B0 3A CC 4E B7 20 74 3E 06 CD BE 45 20
45F0  6D 06 28 CD CA 46 21 C1 4B 11 97 0B 01 16 00 ED
4600  B0 06 28 CD CA 46 3A C1 4E B7 28 0B 11 97 0B 21
4610  27 4C 01 04 00 ED B0 3A C2 4E B7 28 0B 11 A0 0B
4620  21 27 4C 01 04 00 ED B0 3A C3 4E B7 28 0B 11 A9
4630  0B 21 27 4C 01 04 00 ED B0 DF 62 30 B4 FE 0D C8
4640  FE 31 20 06 AF 32 C1 4E 18 A7 FE 32 20 06 AF 32
4650  C2 4E 18 9D FE 33 20 99 AF 32 C3 4E 18 93 AF 32
4660  CC 4E DF 62 38 05 CD 05 48 18 F7 FE 0D C8 18 F2
4670  21 27 4C CD B5 45 06 03 DD 21 C1 4E 11 BE 4E DD
4680  7E 00 12 B7 28 06 CD 05 48 7D E6 3F 3C CD D1 46
4690  3D 20 FA 13 DD 23 10 E7 11 BE 4E 06 03 DD 21 C1
46A0  4E DD 7E 00 B7 28 10 CD 05 48 7D F6 0F 3C CD D1
46B0  46 3D 20 FA 3E 02 12 13 DD 23 10 E5 21 BE 4E CD
46C0  D1 46 7E 23 B6 23 B6 20 F3 C9 FF CD 05 48 10 FA
46D0  C9 F5 C5 D5 E5 DD E5 FD E5 3A BE 4E B7 28 0D 21
46E0  88 4E DD 21 55 08 CD 20 47 32 BE 4E 3A BF 4E B7
46F0  28 0D 21 9A 4E DD 21 5E 08 CD 20 47 32 BF 4E 3A
```



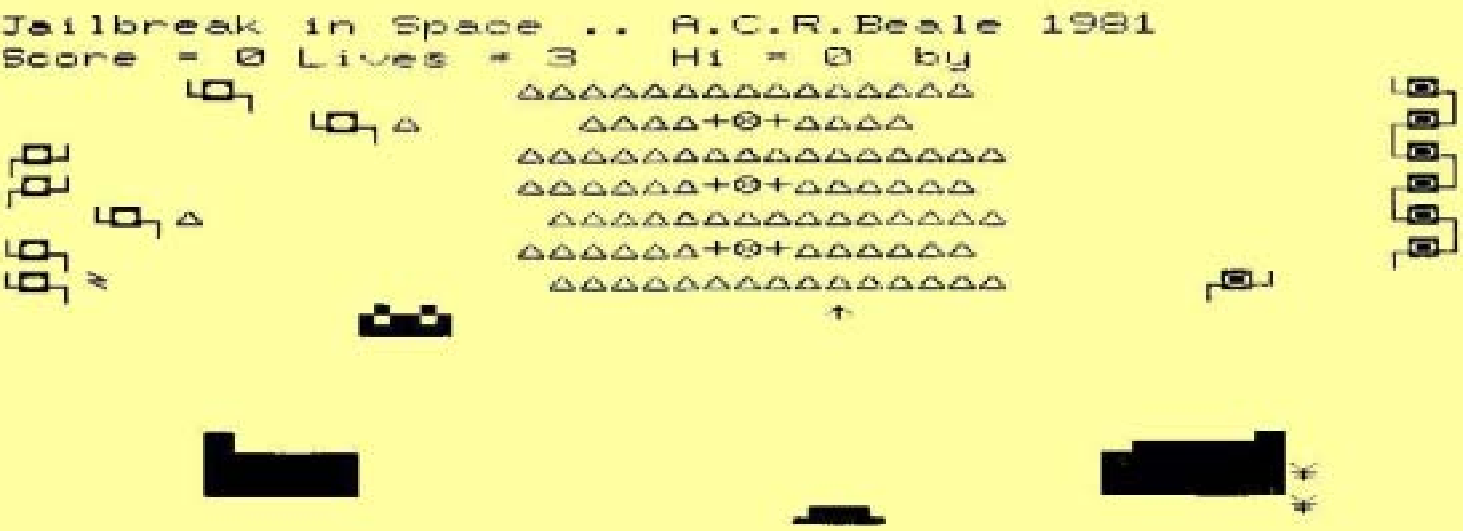
```
4700  C0 4E B7 28 0D 21 AC 4E  DD 21 67 08 CD 20 47 32
4710  C0 4E 06 06 CD CA 46 FD  E1 DD E1 E1 D1 C1 F1 C9
4720  E5 11 76 4E 01 12 00 ED  B0 32 87 4E FD 21 7A 4E
4730  0E 0C FD 7E 00 E6 1C CB  3F 47 CB 3F 80 47 3A 86
4740  4E 80 87 87 47 FD 7E 00  E6 03 80 11 00 00 5F CB
4750  23 CB 12 CB 23 CB 12 CB  23 CB 12 21 33 4C 19 06
4760  08 7E DD 77 00 23 DD 23  10 F7 11 38 00 DD 19 FD
4770  23 0D 28 09 FD 7E 00 E6  03 28 B7 18 E2 3A 87 4E
4780  FE 02 20 1A 3A 86 4E FE  02 20 13 3A 7A 4E E6 03
4790  20 0C 3E 04 32 87 4E 3E  08 32 75 4E 18 5A 3A 87
47A0  4E FE 04 20 17 3A 75 4E  3D 32 75 4E 20 05 32 87
47B0  4E 18 45 FE 01 20 41 32  86 4E 18 3C 3A 86 4E 3C
47C0  32 86 4E FE 03 20 31 AF  32 86 4E 21 84 4E 11 85
47D0  4E 01 0F 00 ED B8 3E FF  32 76 4E 3A 79 4E FE FF
47E0  20 16 CD 05 48 7C E6 0F  21 A5 4B CD 14 48 21 76
47F0  4E 06 04 77 3C 23 10 FB  01 12 00 D1 21 76 4E ED
4800  B0 3A 87 4E C9 2A 73 4E  F5 C5 45 0E 29 09 22 73
4810  4E C1 F1 C9 3C 3D 28 03  23 18 FA 7E C9 DD 21 9F
4820  4B 3E 2F DD 4E 00 DD 46  01 C6 01 ED 42 F2 29 48
4830  09 FD 77 00 FD 23 DD 23  DD 23 0D 20 E4 FD 36 00
4840  30 06 03 FD 7E FD FE 30  C0 FD 36 FD 20 FD 23 10
4850  F2 C9 EF 0C 20 2A 2A 20  46 52 55 49 54 20 4D 41     .  ** FRUIT MA
4860  43 48 49 4E 45 20 2A 2A  20 28 43 6F 70 79 72 69   CHINE ** (Copyri
4870  67 68 74 20 53 2E 43 2E  41 6C 6C 65 6E 20 31 39   ght S.C.Allen 19
4880  38 31 29 0D 0D 20 20 20  59 6F 75 20 68 61 76 65   81)..   You have
4890  20 61 6E 20 69 6E 69 74  69 61 6C 20 73 74 61 6B    an initial stak
48A0  65 20 6F 66 20 31 30 30  70 2E 0D 20 20 20 20 20   e of 100p..
48B0  20 54 68 65 20 63 6F 73  74 20 6F 66 20 65 61 63    The cost of eac
48C0  68 20 67 6F 20 69 73 20  20 31 30 70 2E 0D 0D 20   h go is  10p...
48D0  20 20 20 20 20 20 20 20  20 20 20 20 20 20 20 57                  W
48D0  49 4E 4E 49 4E 47 53 0D  20 20 20 53 79 6D 62 6F   INNINGS.   Symbo
48F0  6C 20 20 20 20 20 20 20  31 73 74 20 26 20 32 6E   l       1st & 2n
4900  64 20 20 20 20 20 41 6C  6C 20 33 0D 20 20 20 20   d     All 3.
4910  42 41 52 20 20 20 20 20  20 20 20 20 20 20 20 32   BAR            2
4920  30 70 20 20 20 20 20 20  20 20 20 20 33 30 70 0D   0p          30p.
4930  20 20 20 20 42 45 4C 4C  20 20 20 20 20 20 20 20       BELL
4940  20 20 20 32 30 70 20 20  20 20 20 20 20 20 20 20      20p
4950  33 30 70 0D 20 20 20 44  4F 4C 4C 41 52 20 20 20   30p.   DOLLAR
4960  20 20 20 20 20 20 20 33  30 70 20 20 20 20 20 20          30p
4970  20 20 20 20 38 30 70 0D  20 20 20 50 4F 55 4E 44       80p.   POUND
4980  20 20 20 20 20 20 20 20  20 20 20 33 30 70 20 20              30p
4990  20 20 20 20 20 20 20 20  38 30 70 0D 20 20 43 48           80p.  CH
49A0  45 56 52 4F 4E 20 20 20  20 20 20 20 20 20 20 34   EVRON          4
49B0  30 70 20 20 20 20 20 20  20 20 20 31 35 30 70 0D   0p         150p.
49C0  20 20 20 20 46 41 43 45  20 20 20 20 20 20 20 20       FACE
49D0  20 20 31 30 30 70 20 20  20 20 20 20 20 20 20 35     100p         5
49E0  30 30 70 00 C9 EF 0C 20  20 20 20 20 20 20 20 20   00p.  .
49F0  20 20 20 20 20 20 20 2A  20 20 48 4F 4C 44 20 20          *  HOLD
4A00  2A 0D 0D 20 20 20 20 57  68 65 6E 20 48 4F 4C 44   *..    When HOLD
4A10  20 66 6C 61 73 68 65 73  20 72 65 65 6C 73 20 63    flashes reels c
4A20  61 6E 20 62 65 20 68 65  6C 64 20 62 79 0D 20 20   an be held by.
4A30  20 20 70 72 65 73 73 69  6E 67 20 6B 65 79 73 20     pressing keys
4A40  31 2C 32 20 61 6E 64 20  33 2E 57 68 65 6E 20 68   1,2 and 3.When h
4A50  65 6C 64 20 48 4F 4C 44  0D 20 20 20 20 20 20 20   eld HOLD.
4A60  20 20 20 20 20 20 20 20  63 65 61 73 65 73 20 74           ceases t
4A70  6F 20 66 6C 61 73 68 2E  0D 0D 20 20 20 20 20 20   o flash...
4A80  20 20 20 20 20 20 20 20  20 20 2A 20 20 4E 55 44             *  NUD
4A90  47 45 20 20 2A 0D 0D 57  68 65 6E 20 20 4E 55 44   GE  *..When  NUD
4AA0  47 45 20 20 69 73 20 61  6E 6E 6F 75 6E 63 65 64   GE  is announced
4AB0  20 2C 20 70 72 65 73 73  20 61 6E 79 20 20 6B 65    , press any  ke
```

4AC0	79	20	20	74	6F	0D	73	74	6F	70	20	20	74	68	65	20	y to.stop the
4AD0	20	6E	75	64	67	65	20	20	63	6F	75	6E	74	20	2E	20	nudge count .
4AE0	55	73	65	20	6B	65	79	73	20	31	20	2C	20	32	20	61	Use keys 1 , 2 a
4AF0	6E	64	20	33	0D	74	6F	20	6E	75	64	67	65	20	72	65	nd 3.to nudge re
4B00	65	6C	73	20	64	6F	77	6E	20	2E	20	4F	6E	63	65	20	els down . Once
4B10	73	74	61	72	74	65	64	20	6E	75	64	67	65	73	20	77	started nudges w
4B20	69	6C	6C	0D	61	75	74	6F	6D	61	74	69	63	61	6C	6C	ill.automaticall
4B30	79	20	63	6F	75	6E	74	20	64	6F	77	6E	20	74	6F	20	y count down to
4B40	30	2E	20	54	6F	20	6E	75	64	67	65	20	61	20	72	65	0. To nudge a re
4B50	65	6C	0D	6D	6F	72	65	20	74	68	61	6E	20	6F	6E	63	el.more than onc
4B60	65	20	70	72	65	73	73	20	6B	65	79	20	61	67	61	69	e press key agai
4B70	6E	2E	00	C9	EF	0D	20	20	20	20	20	20	20	20	20	20	n..
4B80	50	72	65	73	73	20	27	43	27	20	74	6F	20	63	6F	6E	Press 'C' to con
4B90	74	69	6E	75	65	2E	2E	2E	00	CF	FE	43	20	FB	C9	64	tinue....
4BA0	00	0A	00	01	00	00	04	08	08	0C	0C	0C	10	10	10	10	....
4BB0	04	14	14	14	14	0A	04	03	03	02	02	32	0F	08	08	04	...........2....
4BC0	04	48	4F	4C	44	20	20	20	20	20	48	4F	4C	44	20	20	.HOLD HOLD
4BD0	20	20	20	48	4F	4C	44	57	49	4E	20	4C	49	4E	45	59	HOLDWIN LINEY
4BE0	4F	55	20	57	49	4E	20	20	20	20	30	20	20	59	4F	55	OU WIN 0 YOU
4BF0	20	4C	4F	53	45	20	20	20	20	55	4E	4C	55	43	4B	59	LOSE UNLUCKY
4C00	20	20	20	20	20	20	4E	45	41	52	4C	59	20	20	20	57	NEARLY W
4C10	48	41	54	20	41	20	50	49	54	59	20	20	20	20	4E	55	HAT A PITY NU
4C20	44	47	45	20	20	20	20	20	20	20	20	20	20	20	20	20	DGE
4C30	20	20	20	C0	C0	E0	F6	F6	C4	C0	C0	C0	F0	FF	FE	F7	
4C40	FF	C6	C0	C0	C0	D9	F6	F6	CB	C0	C0	C0	C0	C0	C0	C0	
4C50	C0	C0	C0	C0	C0	C0	E4	E4	C0	C0	C0	C0	E0	FE	FD	EF	
4C60	F7	C4	C0	C0	C8	FB	ED	ED	DF	C1	C0	C0	C0	C0	C9	C9	
4C70	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	F4	FB	DF	
4C80	E6	C0	C0	C0	D8	FF	DB	DB	FF	C3	C0	C0	C0	C8	DB	DB	
4C90	C1	C0	C0	C0	C0	C0	D4	E2	C0	C0	C0	C0	E0	CA	E0	C4	
4CA0	D1	C4	C0	C0	C0	D4	C1	C8	E2	C0	C0	C0	C8	C0	C0	C0	
4CB0	C0	C1	C0	C0	C0	C0	E0	C4	C0	C0	C0	C0	C0	D4	C1	C8	
4CC0	E2	C0	C0	C0	C8	E0	CA	D1	C4	C1	C0	C0	D0	C1	C0	C0	
4CD0	C8	C2	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	E0	CA	D1	
4CE0	C4	C0	C0	C0	D0	C1	D4	E2	C8	C2	C0	C0	E0	CA	C0	C0	
4CF0	D1	C4	C0	C0	C0	C0	D4	D2	C4	C0	C0	C0	C0	FC	E4	C0	
4D00	C0	C0	C0	C0	C0	F8	C0	C0	E0	C0	C0	C0	C0	C9	C9	C9	
4D10	C1	C0	C0	C0	C0	C0	E0	E4	C0	C0	C0	C0	C0	F0	C1	C0	
4D20	C1	C0	C0	C0	C0	F9	C9	C0	C0	C0	C0	C0	C0	DA	D2	D2	
4D30	CA	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	E0	CA	C9	
4D40	C2	C0	C0	C0	C0	FA	D2	C0	C0	C0	C0	C0	C0	FC	E4	E4	
4D50	D4	C0	C0	C0	C0	C0	F4	F4	C4	C0	C0	C0	C0	D8	FC	FC	
4D60	CC	C0	C0	C0	C0	D0	FC	FC	DC	C0	C0	C0	C0	C0	C8	C8	
4D70	C0	C0	C0	C0	C0	C0	E0	E0	C0	C0	C0	C0	C0	F0	F9	F9	
4D80	D1	C0	C0	C0	C0	E0	F9	F9	F1	C0	C0	C0	C0	C0	D9	D9	
4D90	C1	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	E0	FA	FA	
4DA0	E2	C0	C0	C0	C0	C8	FA	FA	E2	C0	C0	C0	C0	C8	FA	FA	
4DB0	CA	C0	C0	C0	C0	C0	F4	E6	C0	C0	C0	C0	C0	F8	FF	FF	
4DC0	C7	C0	C0	C0	E0	FE	FF	FF	F7	C4	C0	C0	C0	C0	C8	C1	
4DD0	C0	C0	C0	C0	C0	C0	E0	C4	C0	C0	C0	C0	C0	F0	FF	FF	
4DE0	C6	C0	C0	C0	C0	FC	FF	FF	E7	C0	C0	C0	C8	C9	D9	CB	
4DF0	C9	C1	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	E0	FE	F7	
4E00	C4	C0	C0	C0	C0	F8	FF	FF	C7	C0	C0	C0	D0	DB	FB	DF	
4E10	DB	C2	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	F6	F6	F6	F6	
4E20	F6	F6	C0	C0	C9	C9	C9	C9	C9	C9	C0	C0	C0	C0	C0	C0	
4E30	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	E4	E4	E4	E4	
4E40	E4	E4	C0	C0	DB	DB	DB	DB	DB	DB	C0	C0	C0	C0	C0	C0	
4E50	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	C0	
4E60	C0	C0	C0	C0	FF	FF	FF	FF	FF	FF	C0	C0	C0	C0	C0	C0	
4E70	C0	C0	C0	DA	BD	00	0C	0D	0E	0F	04	05	06	07	14	15	




GRAPHIC GOLF
Strokes 0 Total 7 Hole 2 418 yds. Par 5
Club Wind direction 1 Speed 5


Score=00140 INVASION EARTH Bases=1


Jailbreak in Space .. A.C.R.Beale 1981
Score = 0 Lives = 3 Hi = 0 by