inmc news issue 2

INTERNATIONAL NASCOM MICROCOMPUTER CLUB

It is with great pleasure, and relief, that I write this preface to the second newsletter. I have always wanted the INMC to be run by hobbyists and certainly not by Nascom. My staff will, of course, continue to do the donkey work in answering your queries and collating and printing the newsletter.

As you will see as you get further into this newsletter, I have persuaded various London-based hobbyists to become an INMC Committee. We have been very fortunate in persuading David Hunt to chair this committee. They layout their ideas later on. However, I would immediately say that they have a totally free hand in the running of INMC, they are entitled to publish in the INMC News any relevant Nascom information, be it complimentary or otherwise, that they feel is worthwhile with the exception of any detrimental comments against a Nascom competitor.

I hope that you will really start to support them fully now and that the INMC can become the force that it should be within hobbyist and other computing and that its value to the Nascom user in particular will be significant.

I wish you all every success and hope that the Club library will now show an enormous upturn for the better.

K Boland

Chairman's Letter.

A few weeks ago Kerr Borland of Nascom approached us and asked if we would be interested in forming a committee to run the INMC; so after a most undemocratic election (each proposing the other and voting despite the protests of the nominee), we reluctantly agreed to become the committee until such time as a more democratic method could be adopted. In return for Kerr's generosity in suggesting we become the committee, we landed him with the job as President.

So, having taken the job let us introduce ourselves:

Kerr Borland	Nascom Sales Director	President
Dave Hunt	A Nascom Distributor	Chairman
Richard Beal	Systems analyst/consultant	Software-co-ordinator
Howard Birkett	Film Editor	Hardware co-ordinator
Paul Greenhalgh	Nascom Engineering	General dogsbody

So our first job was to define the aims of the INMC, which we set out as follows:

1. That the INMC should be self-supporting; which would allow the INMC to be reasonably independent of the manufacturer.

2. To distribute hardware and software information about Nascom as cheaply as possible, consistent with making the INMC self-supporting.

3. To do this by means of a software library and newsletter.

4. That the library and mailing address for the INMC should remain as Nascom Microcomputors at Berkhamsted, and that Nascom would publish and distribute the newsletter.

We would like the newsletter to contain news and information that members might consider of interest to other members, as well as software and hardware notes. We would also like to set up a 'Problems Page' to answer specific questions that might be of interest to other members. So send in your articles, problems, moans, praise etc, Nascom will forward them to the appropriate members of the committee for editing and inclusion in the newsletter.

Remember, the more feedback we get from members, the more lively (and more frequent) the newsletter will become. So to sum up, this is your newsletter, USE IT!!!

/ .. ..

All INMC correspondence should be addressed to:

The Editor
INMC Newsletter
c/o Nascom Microcomputors Ltd
121 High Street
Berkhamsted
Herts HP4 2DJ

Yours sincerely

Dave Hunt

NASCOM I - Various Technical Notes

1). Floating Inputs to PORT 0. Keyboard user in puts.

Although the software ignores spurious characters which may appear on PORT O, the keyboard routine still carries out a search to determine whether the input was valid. If, as is likely, the two user inputs on SKT 1 are left unconnected, this could have a detrimental effect on the running of any program with interactive keyboard routines.

For example, as B-BASIC V1.1 scans the keyboard for change at the end of each statement, any spurious input to PORT O, will cause the keyboard routine to 'waste time' searching for a character which does not appear in the keyboard lookup table. This has the undesirable effect of making a 'FOR - NEXT' timing loop vary with each spurious character detected, causing imprecise timings.

This flaw may be easily rectified by connecting the two user inputs to +5 volts, forcing them permanently 'high'. Under these circumstances, no spurious inputs occur.

2). UART clocks.

The effective speed of Load and Dump (and Read and Write) may be doubled by connecting the UART clock link to pin 12 of IC2 (it is normally connected to pin 11 of IC2, via the UART clock link). This modification has been found to work on the majority of Nascoms; further, on some Nascoms it has been found that the speed may be doubled yet again by connecting the link to pin 13 of IC2. It should be noted that these modifications are not 'guaranteed' to work.

Adjustment of the 176OHz (10 chars./sec.) UART clock, without test equipment. Firstly, it should be noted that this clock need not be adjusted until such time as a printer or other

/ .. ..

serial peripheral is added. Adjustment is affected by VR1. Clockwise rotation reduces the clock speed. With a printer attached via the RS232 or 20mA outputs, a short test program may be written that will output continual text in the form '1234567890123.....' etc. Adjustment is made by observing the printed output;

If the clock is too fast, random garbage will appear, thus:

123c5u789,P234+z7 .... etc.

If the clock is too slow, characters will be missed, thus:

1235679013457891 .... etc.

Note that VR1 is a multuturn (20 turns) preset, and that the end stops are detected by an increase in rotational torque at the ends of the track. No harm can be done by over 'turning' the preset.

Correct adjustment is the mid point between garbage and missing characters, this is a latitude of 4 to 5 turns of the pot.

The clock speed may be changed to 4800Hz (30 chars./sec.) by changing C12 to a 8n2 10% polyester capacitor. Setting of the 4800Hz clock is as above.

3). "Snow Plough" and NMI "Break".

The snow plough is used in conjunction with IC11 to increase the VDUSEL blanking time to eliminate 'snow' on the screen during memory access to the video RAM. See Fig. 1. The simplest method of construction is to take one of the spare 16 pin dil plugs (supplied) and cut off two pins, making it a 14 pin plug. Then cut a piece 0.1" pitch vero board about 1" wide by about 1.5" long, with the tracks in the longer direction. Mount a 14 pin socket at one end (breaking tracks as appropriate) and build the LS123 circuit at the other, connecting the output of the 123 to pin 5 of the 14 pin socket. Then solder the 14 pin plug pin for pin to the underside of the 14 pin socket (except pin 5). Link pin 5 of the plug to the input of the 123 circuit, and low!! a little plug in module which carries IC11 and the 123, with a plug that fits directly into IC11 socket on the board. Neat, tidy and effective. Don't forget to connect power to the 123, in parallel with pins 7 and 14 of IC11.

Plug in the module, and a TV display should appear as usual. Tab from 0 to FFFF and adjust the preset pot such that the 'snow' just disappears.

/ .. ..

The NMI 'Break' can only be used with NASBUG T4 and B-BUG. This should be made on a small piece of veroboard and mounted somewhere appropriate. To connect it, a wire should be run from pin 8 of IC42 (under the board) to the input of the circuit. The CPU should be lifted from the board and pin 17 carefully bent out horizontal, the CPU may then be replaced. The output of the circuit is connected to pin 17 of the CPU, using a 'Soldercon' pin. DO NOT SOLDER TO THE CPU.

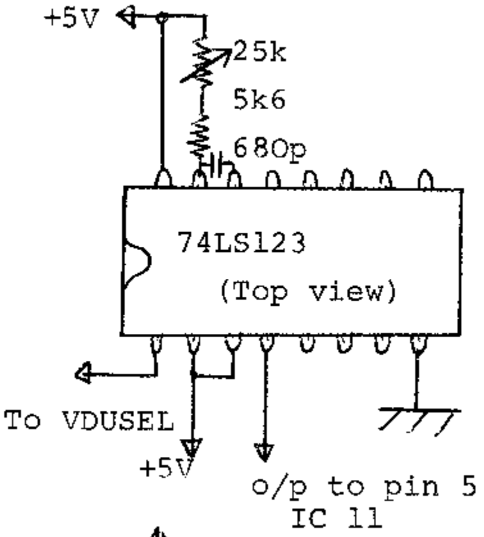

Fig 1. "Snow Plough"

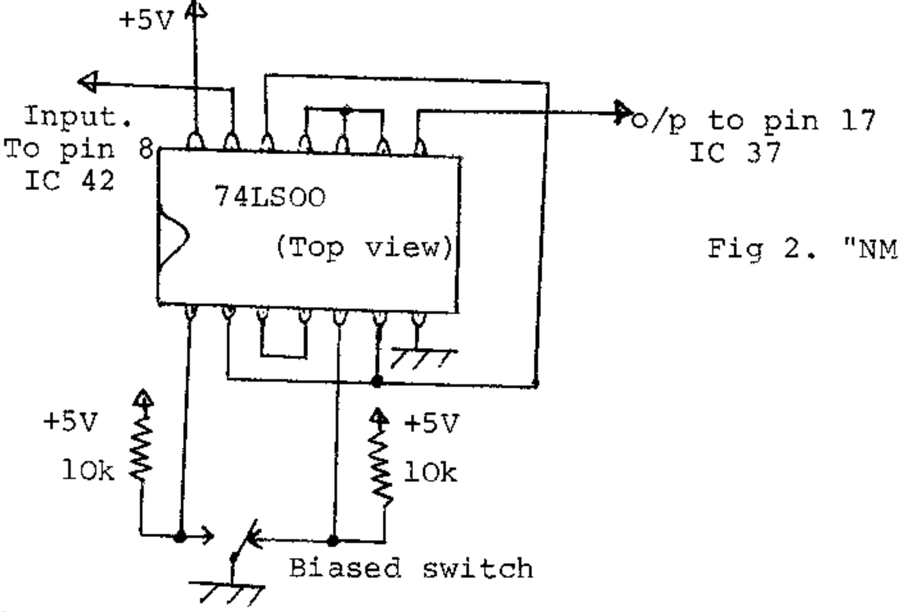

Fig 2. "NMI Break"

SOFTWARE HINTS

1. Suppose you want to compare HL with DE, without changing the contents of either register.
Try this:

```
B7              OR    A
ED 52           SBC   HL, DE
19              ADD   HL, DE
```

/ .. ..

If HL = DE, the Z flag is set, otherwise it is reset.
If HL is greater than or equal to DE, the carry flag is reset.
If HL is less than DE, the carry flag is set.

And it only takes four bytes!

2. Not everyone has realised that the Nascom monitor program uses the Z80 restart instructions to provide some useful features. Print String is an easy way of putting out messages.

```
EF                               RST      28H
48  45  4C  4C  4F               DEFM     /HELLO/
00                               DEFB     0
```

These seven bytes will make the message 'HELLO' be displayed. Don't forget to put the value 00 at the end of the message, or the screen will fill up with the contents of the rest of your program!

3. Have you wondered about the meaning of the characters which hex values 00 to IF give you on the screen? Each one is, in fact, a picture which represents the equivalent ASCII code. For example, 🔔 is a bell!

4. The breakpoint command uses a restart to stop the program and display the registers. If you want, you can put the same code, E7 in hex, in several places in your program. You may find it a good idea to fill any empty space with this code, because if you jump to it by mistake, the program will stop, and the register display may give you some clues.

5. In case all this has been too easy, here is a puzzle for you.

```
AF                 XOR  A              Set A to 0
06  00             LD B,0              Set B to 0

3C     LAB1        INC A              Increment A
27                 DAA                Decimal adjust
10 FC              DJNZ LAB1          Repeat, 256 times.
E7                 RST BRKPT          Display registers.
```

Now A has been incremented 256 times, and the DAA instruction makes this work in decimal, so A should be 56 at the end. Why isn't it, and how would you correct the program? (No, the Z80 doesn't have a fault in it!)

6. The original Nascom Software notes suggested jumping to an

/ .. ..

address in the monitor to end a program. This will cause problems. It is always safe to jump to address O, which restarts the monitor program correctly.

If you don't want to clear the screen

Reset the stack to OC33H
then jump to PARSE

Notes on PIO Operation.

The Nascom I has on board two totally uncommitted 8 bit parallel I/O ports complete with handshake lines, in the shape of an MK3881 Z80 - PIO. The PIO is, in itself, a fairly complicated processor, which needs programming before it will operate in any of its 4 modes:

OUTPUT	MODE O	
INPUT	MODE 1	(automatically set on PIO Reset)
BIDIRECTIONAL	MODE 2	
CONTROL	MODE 3	

It is not the purpose of these notes to describe in detail these operational modes, but to help clear up a few common problems encountered in controlling the PIO.

One very important fact to note is that the PIO is not reset by the RESET button on the keyboard. This resets the CPU only, NOT the PIO. It may be reset in two ways. The simplest is to switch the power off and on again; a bit drastic but the PIO does have automatic power on reset. The second method (shown in fig.3) is to apply an Ml without either $\overline{RD}$ or $\overline{IORQ}$. It should, however, be pointed out that, since the CPU can be reset, it is always possible to regain control of the PIO in software, by simply reprogramming it.

Now to 'interrupts'. Don't forget that the PIO is designed to operate in the Z-80 Interrupt Mode 2, so before doing anything put the CPU into this mode by executing 'IM 2' (HEX code ED 5E). Remember that a CPU reset puts the Z-80 back to Interrupt Mode O clears the I register, and dissables CPU interrupts (having no effect on the PIO).

In Interrupt mode 2, the CPU finds the address of the interrupt routine, by loading the Program Counter (P.C.) with the contents of the memory address. This is formed by the I register (high byte), and the interrupt vector sent from the interrupting port (low byte).

/ .. ..

For example, let us suppose that an interrupt routine for Port A starts at OE12H, and that the interrupt address table will be stored at OF8OH. In order that the routine should be found correctly, the I register should contain OFH, the value 8OH should be sent to the control register of Port A, and finally, memory locations OF8OH and OF81H should contain 12H and OEH respectively (low byte first). At an interrupt, CPU interrupts are automatically disabled and must be re-enabled, if required, by the programmer.

Always end an interrupt service routine with the RETI instruction, as this is the only way to indicate to the PIO port, that the service routine is finished. This feature can cause some dismay to the unwary. Take the following example: everything is set up correctly, and the PORT interrupts correctly. However, unfortunately the interrupt routine crashes. No problem to our intrepid experimenter, he presses reset, debugs the interrupt routine and tries again, remembering to reset IM 2, I register and interrupt enable. Dismay! Nothing happens. No interrupt.

The problem is that the PIO still thinks that is is being serviced for its initial interrupt, and is internally inhibited from causing another. A useful routine to get out of this sort of problem is as follows:-

```
21 OO OO          LD        HL,  OOOOH
E5                PUSH      HL
ED 5E             RETI
```

This will tell the PORT that its service routine is finished and then restart the monitor by executing from OOOOH. It can be used at any time, if there is any doubt as to the status of a PIO.

Once the mode and interrupt control have been set, the Port interrupt may be enabled or disabled by sending 83H or O3H to the control register. This feature could form the basis of a generalized interrupt control program for a given system. However, it should be noted, that the correct way to disable a port interrupt, is to first of all disable CPU interrupts before the Port interrupt. This is because an interrupt by that Port, during the execution of the instruction to disable its interrupt, would cause a system crash.

Finally, when a Port has been disabled, an interrupt may be pending, so that when the Port is again enabled it will at once interrupt the CPU. This Pending interrupt may be cleared, if required, by sending an interrupt control word with bit 4 set. This is effective in all modes.

/ .. ..

Please let us know of any interesting applications for your PIO, or better still write an article for YOUR newsletter.

Two programs by Dave Hunt will be available from the Software library for those interested in checking out the ports.

These are PIO Latch Test
& PIO Vectored Interrupt Test

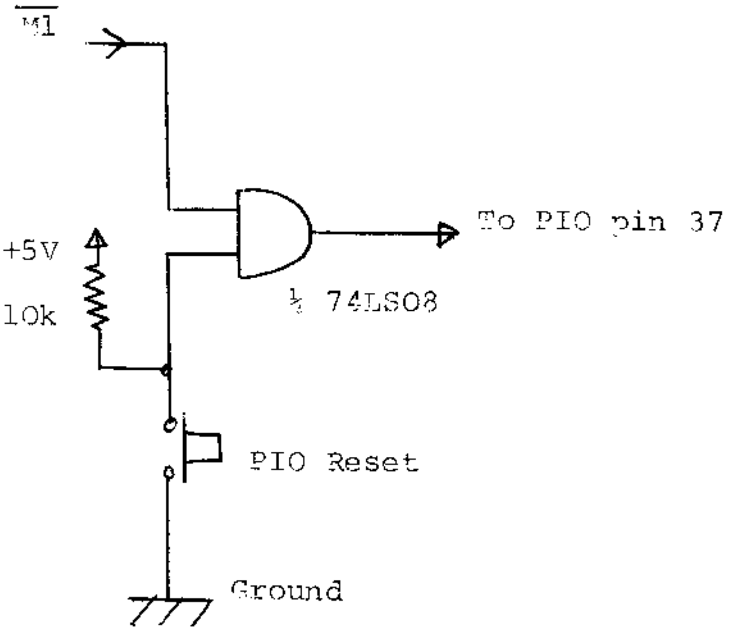

Figure 3.
Reset circuit for PIO.
(Switch is Push to Make.)

Nascom Users Group

We hear that Merseyside Nascom owners have formed a users group which meets on the first Wednesday of every month in Liverpool. All enquiries should be made by contacting Graham Myers on 051-677-9340 (after 7.00 p.m.)

QUERIES AND FAULTS.

The Nascom distribution network has been set up in order to give the customer a more personal and efficient back up service. If you have any queries on any aspect of your Nascom you should contact your distributor who will be

/ .. ..

willing to assist you with the problem.

If you have a hardware fault many of the distributors are able to offer a repair service. If your distributor does not offer a repair service, then the unit can be returned direct to Nascom's service department.

Finally, if you feel that you are not getting anywhere with either your distributor or Nascom, then write to the Editor, INMC, and we will see what we can do.

APPLICATIONS

We would be interested to hear from anybody who is using, or would like to use, their Nascom 1 for any specialised purpose.

Amateur Radio - We hear that John Wilson, G8HUN, is compiling details of amateur radio applications and he would like to talk to anyone interested in this field. So far he knows of people who are investigating using their Nascom for RTTY, transmitting and receiving morse, satellite tracking, controlling synthesizers etc. Anyone else interested in these or similar applications should contact John, in the first instance c/o INMC.

Whilst on this subject, it should be noted that the 2K monitor, Nasbug T4, contains amongst its routines one that enables radio amateurs to transmit and receive ASCII data with no additional software and minimal hardware modification.

MICROCOMPUTER BOOKS

Mine-of-Information of St.Albans stock a range of microcomputer books and will offer members of the INMC a discount of 10%. Enquiries should be made to:

Mine of Information Ltd
1 Francis Avenue
St.Albans
AL3 6BL

/ .. ..

MYSTERY PROGRAM

This program has been written by an anonymous INMC member - would you please identify yourself! As we don't know who the programmer is, we won't tell you what the program does - try it! All we will say is that the space bar runs the program, it executes at OC60 and you really need the "snow-plough".
P.S. Apologies to the writer for the mods. we've made.

```
0C60 3E AA 32 51 0E 21 00 00      0E00 00 CD EE 0E 00 CD 50 0D
0C68 22 D8 0D 01 05 0D 21 AA      0E08 CD B0 0D C1 0D D9 C0 D9
0C70 0B 11 19 60 D9 CD D6 0C      0E10 0E 05 C5 CD A5 0D CD 80
0C78 00 00 CD C0 0D 11 7A 08      0E18 0E C1 FE AA 28 1C 00 3E
0C80 CD 00 0D 11 FA 09 CD 00      0E20 19 BB 11 27 19 28 03 11
0C88 0D CD 5C 0D CD 28 0D CD      0E28 19 60 00 00 05 28 05 D9
0C90 1A 0D CD 37 0D 00 00 00      0E30 C9 00 00 00 11 19 19 CD
0C98 CD D8 0D 3A 51 0E FE 96      0E38 90 0D 11 19 19 21 00 00
0CA0 CC 58 0E 00 00 00 00 00      0E40 22 D8 0D 06 0D 2A 51 0E
0CA8 00 00 00 00 00 CD 70 0D      0E48 2B 2B 2B 2B 22 51 0E D9
0CB0 FE 11 28 06 FE 03 28 07      0E50 C9 AA 0B 00 00 00 00 00
0CB8 18 19 11 7A 08 18 03 11      0E58 3E AA 32 51 0E 21 7C 0F
0CC0 FA 09 CD 00 0D 3E 20 32      0E60 CD DC 0C CD 69 00 30 FB
0CC8 89 0A 3E 20 32 FA 08 00      0E68 CD C0 0D 21 0A 08 11 0B
0CD0 CD 5C 0D C3 8C 0C EF 1E      0E70 08 01 80 00 ED B0 CD D9
0CD8 00 21 60 0F 11 D5 0B 01      0E78 0C D9 21 AA 0B D9 C9 00
0CE0 19 00 ED B0 C9 00 00 00      0E80 00 00 00 E5 3E 20 23 BE
0CE8 00 00 00 00 00 00 00 00      0E88 20 1A 2B 2B 2B BE 20 14
0CF0 00 00 00 00 00 00 00 00      0E90 CD A5 0D BE 20 0E 23 BE
0CF8 00 00 00 00 00 00 00 00      0E98 20 0A 23 BE 20 06 23 BE
0D00 00 00 00 21 00 0F 0E 05      0EA0 20 02 E1 C9 00 E1 00 00
0D08 C5 01 10 00 ED B0 E5 21      0EA8 00 CD 37 0D CD 5C 0D 11
0D10 30 00 19 EB E1 C1 0D 20      0EB0 FA 09 CD 00 0D 11 80 0A
0D18 EF C9 21 C9 09 11 C8 09      0EB8 21 50 0F 01 09 00 ED B0
0D20 01 42 01 ED B0 C9 00 00      0EC0 11 40 0A 21 20 0F 01 0A
0D28 21 B8 09 11 B9 09 01 40      0EC8 00 ED B0 21 20 0F 11 FE
0D30 01 ED B8 C9 00 00 00 21      0ED0 09 01 0C 00 ED B0 CD 5C
0D38 7A 08 01 05 00 3E 20 77      0ED8 0D 3E AA C9 00 00 00 00
0D40 11 40 00 19 0D 20 F6 3E      0EE0 00 00 00 00 00 00 00 00
0D48 0B BC C8 21 C9 09 18 EA      0EE8 00 00 00 00 00 00 3A 0A
0D50 0E 03 CD 35 00 0D 20 FA      0EF0 0B FE 20 C8 CD B0 0D C9
0D58 C9 00 00 00 0E 13 C5 CD      0EF8 00 00 00 00 00 20 0B FF
0D60 28 0D CD 1A 0D CD D8 0D      0F00 20 20 20 20 20 5F 5F 5F
0D68 CD 37 0D C1 0D 20 EF C9      0F08 5F 5F 5F 20 20 20 20 20
0D70 00 3E 20 21 87 0D C5 47      0F10 20 20 20 20 AF 20 20 28
0D78 ED 5F 86 38 01 3D 77 90      0F18 29 20 20 5C 20 20 20 20
0D80 30 FD 80 3C C1 00 C9 67      0F20 FF FF FF FF FF FF FF FF
0D88 00 00 00 00 00 00 00 00      0F28 FF FF FF FF FF FF FF FF
0D90 E5 D5 C5 7B 77 2B 7A 77      0F30 FF FF FF FF FF FF FF FF
0D98 CD A5 0D 3E 28 77 23 3C      0F38 FF FF FF FF FF FF FF FF
0DA0 77 C1 D1 E1 C9 0E 40 2B      0F40 20 20 20 5C 5F 2F 20 20
0DA8 0D 20 FC C9 00 00 00 00      0F48 20 20 5C 5F 2F 20 20 20
0DB0 E5 3E 20 77 2B 77 CD A5      0F50 41 4D 42 55 4C 41 4E 43
0DB8 0D 77 23 77 E1 C9 00 00      0F58 45 00 00 00 00 00 00 00
0DC0 21 AA 0B 0E 05 11 19 19      0F60 2A 20 4C 4F 4C 4C 59 50
0DC8 E5 CD 90 0D E1 2B 2B 2B      0F68 4F 50 20 4C 41 44 59 20
0DD0 2B 0D 20 F1 C9 00 00 00      0F70 54 52 41 49 4E 45 52 20
0DD8 00 00 00 00 00 00 00 00      0F78 2A 20 20 20 50 72 65 73
0DE0 00 00 00 00 00 00 00 00      0F80 73 20 73 70 61 63 65 20
0DE8 00 CD 69 00 FE 20 28 04      0F88 49 4E 4D 43 20 52 55 4C
0DF0 CD 50 0D C9 21 18 20 22      0F90 45 53 20 4F 4B 00 00 00
0DF8 D8 0D D9 CD 90 0D C5 00
```

/ .. ..

NASBUG T4 Extended 2K Monitor for Nascom I

NASBUG T4 incorporates the best facilities of NASBUG T2 and B-BUG, and has been further extended to allow keyboard access to the PORTS, use as an intelligent terminal, keyboard shift options, and the Z80 restarts to be of more use to the user.

Command Table

A Hexadecimal arithmetic to calculate the sum, difference and relative jump of two addresses.

B Breakpoint as NASBUG T2, but also automatically relocates the cursor to the bottom left of the screen if it has been moved by the user. Breakpoint is set to zero on Reset.

C Copy as in NASBUG T2.

D Dump as in NASBUG T2, but with extra features for error erradication.

E Execute as in NASBUG T2.

G Generate. On reading a tape recorded in this format, the tape enters its own Read and Execute commands and automatically executes itself.

I Intelligent Copy will copy data up or down without corruption which can occur under certain conditions using the C command. C command has been left in the command table as the corruption caused can be deliberately used to profit under certain conditions.

K KO Nascom keyboard as normal but shift now gives lower case letters. KO is automatically set on RESET.
K1 Letters shift is inverted from KO (typewriter mode).
K2 As KO, but holding down the space bar causes the ASCII representation of the character typed to be displayed.
K3 As K1 but in ASCII mode as K2.

L Load as NASBUG T2.

M Memory examine/modify as NASBUG T2, but is additionally capable of backwards stepping through memory, and immediate jumps to different locations in memory.

N Reverts 'X' to normal.

O Output to a port.

Q Input from a port.

R Read as in B-BUG. Four times faster than 'L'.

S Single-step as NASBUG T2, but relocates cursor as in 'B' (above).

T Tabulate as in NASBUG T2.

/ .. ..

NASBUG T4 (Continued)

W Write as in B-BUG. Four times faster than 'D'.

X Multiple option external mode, which converts Nascom to a full ASCII intelligent terminal. Capable of supporting paper tape with odd or even parity, with or without automatic CR/LF, Teletype as above in half or full duplex, external mainframe timesharing systems through a telephone modem in full or half duplex, odd or even parity, with or without automatic CR/LF, and of course multiple Nascom configurations. This command is possibly the most powerful of all.

Z Directs the Nascom to accept a new command table at the argument supplied by 'Z'.

? Prints out the current command table in the following format:

A B C D E G I K L M N O Q R S T W X Z ?

Restart vectors (Z80 page O)

RST O (C7) Restart NASBUG T4

RST 8 (CF) 'Soft restart' NASBUG T4. As RST O but does not clear screen.

RST 16 (D7 xx) Allows relative subroutine calls to be made using displacement (xx). Note that this feature is supported by NASBUG T4 and not by the Z80, and therefore cannot be used in Z80 based systems not using NASBUG T4.

RST 24 (DF xx) Allows a direct call to location OEOO plus a displacement (xx), the displacements are in 3's, allowing the user to locate tables, reflective jumps etc. in this area. Note; not supported by Z80 as RST 16.

RST 32 (E7) Breakpoint as in NASBUG T2.

RST 40 (EF) String print as in NASBUG T2.

RST 48 (F7) Direct call to $-CRT

RST 56 (FF) Calls part of KDEL as in NASBUG T2, allowing KDEL to be shortened proportional to the value in A, allows for accurate timing in increments of approx. 50 uS.

SOFTWARE LIBRARY

The original intention of the INMC software library was to gather together user programs and offer them to members for a minimal photocopying charge - they would be unchecked and untested by the INMC as the originator would presumably have debugged them. However, from the programs that the INMC have so far received we can see that this system will not work - unless you want 8 different versions of Mastermind and 5 Hangmans! We are,

/ .. ..

therefore, sorting through the programs at the moment and putting them into various categories - e.g. runs on unexpanded Nascom, runs on expanded Nascom, runs under Tiny Basic, Super Tiny Basic etc. We hope to have a list available shortly but meanwhile you'll find a machine code program and some Super Tiny Basic examples elsewhere in this newsletter.

However, it is obvious to the committee that everybody lost interest during 1978. This, of course, includes us. We have taken on the task of trying to re-establish the INMC on the assumption that most users, like ourselves, would rather have it working than not have it at all. Therefore, we need your help, your support, your programs and your ideas and hardware additions that we can publish in our newsletter. Now that many people have expanded Nascoms we hope that we will start to see significant numbers of programs of a more interesting nature than perhaps was possible before when one had to create the whole thing in machine code.

This first three months is critical not only from the point of view of you starting to believe in us, but also to confirm in our minds that the INMC is practical. The whole thing is now in our control and having objected strongly to the way that Nascom ran it last year, we rely on your support so that we can run it properly.

COMPETITION.

We have decided to hold a competition to see what sort of nutty games programs you are all writing. The rules are outlined below. First prize will be either a Super Tiny Basic or a Zeap editor/assembler cassette along with a selection of the programs submitted. There will also be five runners-up prizes, each being a selection of the programs submitted to the competition. So send in your programs - don't worry what your coding is like, we won't be judging that!

Rules

1. All entries must be received by 27th May 1979

2. Winners will be notified by post and will be listed in a future newsletter.

3. Programs must run in an unexpanded Nascom and must be Nasbug T1/T2 compatible.

4. Programs will be primarily judged on "entertainment value".

5. Additional consideration will be given to original and to neatly written and well commented programs.

/ .. ..

6. All entries must be made on paper - no cassettes or alternative formats will be judged.

7. All entries become the property of the INMC and may be added to the software library.

8. The judges will consist of the members of the INMC along with their families and any passers by.

9. The final (after the fighting has finished!) decision of the judges is absolute and no correspondence on it will be answered!

10. Any number of programs may be submitted by an entrant.

11. Programs that have already been submitted to the INMC may be entered but this must be done by sending in a new copy.

12. No correspondence for the INMC or any part of the Nascom organisation should be included with the entry.

13. Alongside the Name and Address of the Entrant the preference for Zeap or Super Tiny Basic should be indicated.

14. All entries should be addressed to:

INMC Games Competition
c/o Nascom Microcomputers
121 High Street
Berkhamsted
Herts HP4 2DJ

ZEAP

Quite a few people have written to Nascom detailing "errors" that they have found in Zeap. In actual fact we know of very few incidences of faulty tapes being supplied or of any major operational bugs - errors have usually been found to be caused by incorrect entry of source programs, or by faulty memory boards. Please read the Zeap manual carefully to ensure that you are operating Zeap correctly. If you are in any doubt as to if your memory board is functioning correctly, then contact either your distributor or Nascom.

All of the members of the INMC committee have been running Zeap for some time now, and many programs have been written using it - this includes the 2K monitor, Nasbug T4. We are, therefore, in no doubt that the Zeap package is an extremely powerful and worthwhile Nascom product.

/ .. ..

Situations Vacant - Software

Nascom Microcomputers are looking for a programmer to work on disc operating systems and languages. If you would like to be considered please send a brief career resume to Tony Rundle, Software Director, 121, High Street, Berkhamsted, Herts.

Another Nascom Users Group

Frank M. Butler would like to hear from other local NASCOM users with a view to starting a club in North Wales. Enquiries to:

Frank M Butler,
8A, Church Side,
Mansfield,
Notts.
NG18 1AD

Telephone: Mansfield (0623) 29237

Double Mastermind

A code guessing game for the Nascom 1. By D. Ritchie.

This programme was included in the first batch of programmes issued by the INMC. Unfortunately, it did not copy very well, and a number of customers were unable to read the object code listing. We are, therefore, including a copy of the object listing in the newsletter. The source listing will continue to be available from the INMC library in the normal way.

Notes on the game

Codes are made up of any combination of four of the octal digits (0 - 7).

The score for each guess is given as 2 digits. The first is the number of correct digits in correct position. The second is the number of correct digits in the wrong position.

You and the machine take alternate guesses at each others code. You first enter a guess at the machines code, 'newline' gives your score. Another 'newline' gives the machines guess at your code. After entering its score, 'newline' lets you enter your next guess, and so on until both codes are found. Pressing R will re-start the game at any time. Backspace can be used to correct entries.

Notes on Programme

Start address is OD22
OF65 to OFAO approx. are used for storage.

Double Mastermind by D. Ritchie

Executes from OD22

```
 M 0C50,0F5B
0C50 11 6D 0F 21  69 0F AF 08  06 04 1A BE  CC 7F 0C 13
0C60 23 10 F7 08  C9 21 69 0F  AF 08 0E 04  06 04 11 6D
0C70 0F 1A BE CC  7F 0C 13 10  F8 23 0D 20  EF 08 C9 B7
0C80 F8 2F 12 7E  2F 77 08 3C  08 C9 21 68  0F 06 04 23
0C90 7E B7 F2 96  0C 2F 77 10  F6 C9 21 69  0F 0E 04 ED
0CA0 5F 07 07 07  07 E6 07 77  23 08 2F 47  10 FE 0D 20
0CB0 EE C9 01 04  00 21 65 0F  11 69 0F 30  01 EB ED B0
0CC0 C9 21 1B 0F  7E 23 FE 04  28 05 CD 3B  01 18 F5 11
0CD0 D9 0B 01 0D  00 ED B0 3E  20 32 8A 0B  C9 78 B9 F0
0CE0 04 DD 36 00  20 DD 36 FB  20 DD 2B DD  2B C9 D9 19
0CF0 E5 DD E1 D9  C9 CD 3E 00  FE 52 CA 22  0D C9 CD F5
0D00 0C FE 1D C8  FE 30 38 F6  BD 30 F3 C9  FD 34 01 FD
0D10 7E 01 FE 0A  38 06 DD 36  F2 31 D6 0A  C6 30 DD 77
0D20 F3 C9 31 00  10 21 2D 08  11 20 00 D9  FD 21 E0 0F
0D30 AF FD 77 FF  FD 36 00 75  FD 77 01 CD  C1 0C CD 9A
0D40 0C 37 CD B2  0C CD EE 0C  FD CB FF 46  C2 C7 0D AF
0D50 CD B2 0C 0E  04 41 2E 38  CD DD 0C DD  36 00 5F CD
0D60 FE 0C FE 1D  28 F2 DD 77  00 DD 23 DD  23 10 EC CD
0D70 F5 0C FE 1D  28 E2 FE 1F  20 F5 21 70  0F DD 2B DD
0D80 2B DD 7E 00  D6 30 38 04  77 2B 18 F1  CD 50 0C F5
0D90 C6 30 DD 77  0C CD 65 0C  C6 30 DD 77  0E F1 FE 04
0DA0 20 25 FD CB  FF C6 06 0B  21 CA 0E FD  7E 01 FE 04
0DB0 38 0C FE 06  30 05 21 D5  0E 18 03 21  E0 0E 7E DD
0DC0 77 43 DD 23  23 10 F7 FD  7E FF FE 03  28 03 B7 20
0DD0 0E CD F5 0C  FE 1F 20 F9  FD 7E 01 FE  0E 28 F2 CD
0DE0 EE 0C CD 0C  0D FD CB FF  4E C2 C7 0E  DD 36 F9 3F
0DF0 CD 9A 0C D9  01 01 10 D9  D9 0B CB 78  D9 28 0B 21
0E00 0B 0F 11 0C  00 DD 19 C3  B8 0E CD 8A  0C 06 04 34
0E10 CB 5E 28 05  36 00 2B 10  F6 21 75 0F  22 73 0F 7D
0E20 FD BE 00 28  22 11 6D 0F  01 06 00 ED  B0 CD 50 0C
0E30 EB BE 20 C4  CD 65 0C EB  23 BE 20 BC  CD 8A 0C 2A
0E40 73 0F 0E 06  09 18 D5 EB  21 69 0F E5  01 04 00 ED
0E50 B0 E1 06 04  7E C6 30 DD  77 F9 23 DD  23 DD 23 10
0E60 F3 0E 02 41  CD DD 0C B7  28 02 2E 35  DD 36 FB 5F
0E70 CD FE 0C FE  1D 28 ED DD  77 FB D6 30  67 DD 23 DD
0E80 23 05 78 B7  28 05 7D 94  6F 18 E1 CD  F5 0C FE 1D
0E90 28 D2 FE 1F  20 F5 FD 7E  00 C6 06 FD  77 00 7C 13
0EA0 12 DD 7E F7  D6 30 1B 12  FE 04 20 1B  21 EB 0E FD
0EB0 CB FF 46 28  03 21 FB 0E  FD CB FF CE  06 10 7E DD
0EC0 77 2B DD 23  23 10 F7 C3  45 0D 41 4D  41 5A 49 4E
0ED0 47 20 21 20  20 06 20 56  45 52 59 20  47 4F 4F 44
0EE0 59 45 53 2C  41 54 20 4C  41 53 54 59  4F 55 20 4D
0EF0 41 59 20 43  4F 4E 54 49  4E 55 45 22  52 22 20 46
0F00 4F 52 20 52  45 2D 53 54  41 52 54 4D  41 52 4B 49
0F10 4E 47 20 45  52 52 4F 52  20 5E 20 1E  20 20 20 20
0F20 20 20 59 4F  55 52 53 20  20 20 20 20  20 20 20 20
0F30 4C 49 4E 45  20 20 20 20  20 20 20 20  4D 49 4E 45
0F40 1F 1F 1F 1F  1F 1F 1F 1F  1F 1F 1F 1F  1F 1F 04 4D
0F50 41 53 54 45  52 4D 49 4E  44 20 49 49
```

TINY BASIC PAGE

FIRSTLY, HAVE YOU NOTICED THAT SETTING UP THE ARRAY TO A CERTAIN VALUE ALWAYS SEEMS TO TAKE A LONG TIME. WELL IF YOU HAVE THE 3K TINY BASIC, YOU CAN MAKE USE OF THE MCU COMMAND TO SET THE ARRAY BY A MACHINE CODE UP COPY. THIS SHORT SUBROUTINE (COURTESY OF HOWARD) MAKES FULL USE OF THIS FACILITY, AND EVEN ALLOWS SETTING PARTS OF THE ARRAY.

```
B-BASIC V1.1
OK
>LIST
  10 REM                  FAST ARRAY SETUP SUBROUTINE
  20 REM                             ***************
  30 REM   SETS THE FROM @(N) TO @(L) TO THE VALUE K
  40 REM   ENTER WITH K, L AND N SET
  50 REM   ALSO USES VARIABLES J AND M
  60 REM   RETURNS WITH J=1 IF A COMBINATION OF L AND N ARE ILLEGAL,
      OTHERWISE J=0
  70 IF (N<0)+(L<=N)+(L>S./2) L. J=1; RET
  80 L. J=K, M=4096
  90 MCK
 100 L. M=K-2+(2*N), L=2*(L-N), K=J, J=0
 110 MCW
 120 L. N=M, M=M-2
 130 MCU
 140 RET
OK
>
```

ALSO WHEN USING THE MACHINE CODE FACILITIES OF THE 3K TINY BASIC, SOME NEAT TRICKS WITH THE MCI AND MCP COMMANDS ARE POSSIBLE. ONE IS TO FIND THE LENGTH OF L AFTER AN MCI INPUT.

```
B-BASIC V1.1
OK
>LIST
  10 REM   TO FIND L WHEN USING AN MCI COMMAND
  20 REM              *****************
  30 REM   SET L AND M AS IN THE MANUAL AND INPUT THE STRING
  40 L. L=20, M=16000; MCI
  50 REM   NOW FIND THE REAL LENGTH OF L
  60 M=M+L-1
  70 MCL; IF M=32  L. L=L-1, M=M-2; G.70
  80 REM   L IS NOW EQUAL TO THE LENGTH OF THE STRING
OK
>
```

FURTHER, WHEN USING MULTIPLE STRINGS, THE ARRAY MAY BE USED TO KEEP TRACK OF THE ADDRESSES AND LENGTHS OF THE STRINGS.

```
B-BASIC V1.1
OK
>LIST
  90 REM   USING THE ARRAY TO HOLD STRING LENGTHS AND ADDRESSES
 100 REM             ****************************
 110 REM   THE FIRST LOCATION CONTAINS THE NUMBER OF STRINGS
 120 REM   AND IS INCREMENTED AFTER EACH MCI INPUT
 130 L. @(0)=@(0)+1
 140 REM   THEN THE VALUE OF M, AND THE NEW L
 150 L. @(1)=16000, @(2)=L
 160 REM    'ODDS' CONTAIN THE START OF THE STRING, AND 'EVENS' THE LENGTH
OK
>
```

The following little program demonstrates what can be done with the strings and the array.

```
  10 P. $↑
  20 P. "Good day, I'm a Nascom, what is your name ?"
  30 L. M=16000, L=20, @(0)=0, @(1)=M
  40 MCI; M=M+L-1
  50 GOS. 500
  60 REM   NOW PRINT STRING 1, USING K AS THE STRING NUMBER
  70 P. "Well ",; K=1; GOS.610; MCP; P." It's nice to know you."; P.
  80 P."Tell me (in a couple of words) what the weather is like.  ",
  90 REM   NOW CALCULATE THE NEXT M
 100 GOS. 710; L=20; MCI; M=M+L-1
 110 GOS. 500
 120 P.; P. "I see. As this is a demo program, I'm going to  let you enter
     any old rubbish you like now."
 130 GOS. 710; L=47; MCI; M=M+L-1
 140 GOS. 500
 150 P.; P.; P.; P.;  P. "Fine, I hope you feel better, now just to prove I
      can do it, I've printed the strings."; P.
 160 P. "The rubbish you typed was"
 170 K=3; GOS. 610; MCP; P.; P.
 180 P. "The weather is ",; K=2; GOS. 610; MCP; P. "."; P.
 190 P. "Bye ",; K=1; GOS. 610; MCP; P.", have a nice day."; P.
 200 S.
 500 REM   SUBROUTINE TO FIND REAL L
 510 MCL; IF K=32 L=L-1, M=M-2; G.500
 520 L. @(0)=@(0)+1, @(2*@(0))=L
 530 RET
 600 REM   SUBROUTINE TO FIND L AND M, USING K AS A MESSAGE NUMBER.
 610 L. M=@(2*K-1), L=@(2*K); RET
 700 REM   CALCULATE NEW M
 710 L. M=@(2*@(0)-1)+@(2*@(0)), @(2*@(0)+1)=M; RET
OK
>
```

Situations Vacant

Nascom Microcomputers are looking for an Electronics Technician to work at their Berkhamsted office. The position will include building prototypes, answering technical queries and assisting in the Repair Department. All enquiries should be made to: Mr W J Bulman at Nascom Microcomputers Limited. 92 Broad Street, Chesham, Bucks.

Note from the INMC Committee

Well, that's the end of this newsletter. We hope you like it. In the next issue we hope to have details of the programs in the Software Library, further Software and Hardware hints, and the solution to the little puzzle in this issue. We also look forward to receiving letters, criticism and information from you to include in YOUR newsletter.

Logically Yours,

THE INMC COMMITTEE