inmc news issue 3

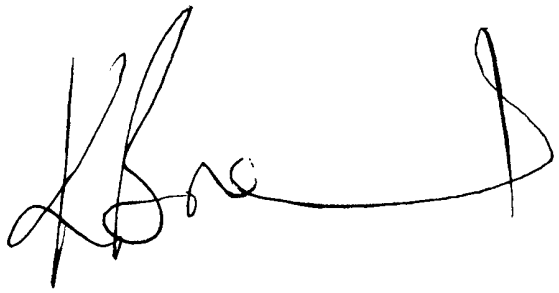
I must say that the immediate response to the second newsletter was very encouraging and 99% of the people were enthusiastic. The usual three people did not like it but I hope they will like this one.

The Committee has so far only really produced the newsletters but I hear rumour that they are about to generate some new ideas. One small problem with the Committee is that in January, when it was formed, its members were independent users but since that time the majority of them have become Nascom employees. They are still as critical but slightly more biased. Therefore, I ask again for people to write in who are willing to be part of either the main Committee under David Hunt's guidance as Chairman, or who would help on a regional capacity as, for example, the INMC Officer in their local computing club. Many clubs that I know of are now getting a large number of Nascom users but none of them, as far as I am aware, have an INMC Official.

Also, I do feel that, sceptical or not, more of you could have made an effort to support Hunt and his cronies and sent in more letters or ideas or programs or engineering tips. Trying to produce a magazine without this kind of help from the outside is bloody difficult and if you want to have an INMC then start helping yourselves.

On a happier note, I was really pleased with the response to the competition in which we got some incredible programs and you will see the results and the winner's name in this issue. I am quite prepared to put up prizes for competitions as long as people are willing to enter them sensibly.

Lastly, I would suggest that Nascom users when going to buy product buy it from their local Distributor and support him, otherwise in many areas you will not have a local Distributor at all. If you all want to buy from one central warehouse, I feel that you will lose the personal value that you all seem to feel is most important.



Kerr Borland
INMC President

UNFORTUNATELY

THIS COULD CONCERN YOU

INMC 3 also contains monosodium glutamate, permitted colouring matter, ion exchange reagents, toluene di-isocyanate, office coffee (very nasty), methanol, nitromethane, non-milk fats and traces of wisdom, wit and humour (less than 3 parts per million)

INMC 3 has been awarded a U certificate by the British Board of Software Censors and an X certificate by the Society for the Prevention of Cruelty to Computers.

STANDARDS: All hardware projects, modifications etc. published in INMC 3 have nothing whatever to do with MIL SPEC 999/X/43 , DEF 95(classified), VDE, IEEE, S1OO, US NAVY STD. 7094, ICCC or the Construction and Use Regulations,1964.

No liability can be accepted for brain damage, CPU malfunctions, hebrephrenic schizophrenia or indigestion caused by reading INMC3.

LETTERS TO THE EDITOR

Dear Sir,

There must be a number of NASCOM 1 owners who, like myself thought they were purchasing a kit for a complete and usable micro system. This system offered the possibility of software exchange via cassette.

Now I am reading in the INMC News and elsewhere about NASBUG T1/T2,T4, B-BUG, Tiny Basic, 3K Tiny Basic, Level A, Level B, etc. No doubt NASCOM et al. are hoping to tempt owners such as myself to expand our systems and run more sophisticated programs.

From the NASCOM owner's angle this is most unsatisfactory. There is a notable lack of information as to the interdependency of the various developments, e.g. is 3K Basic compatible with Nasbug T1? Does 3K Basic require 8K of memory? How compatible are 2K & 3K Basic? What comes after T4? What happened to 8K Basic?

The hobbyist needs to be able to be confident before he starts expanding that

(i) His expansion will serve his needs.
(ii) Costs will be controlled.
(iii) His old software will still run.
(iv) Software from other sources will run.
(v) Further developments will not make his system obsolete.

The name of the game is "information".

Can we have some?

Yours faithfully,

J.Griffiths
West Yorkshire.

Dear Mr.Griffiths,

UPWARD COMPATIBILITY

The INMC promises to do its best to ensure that you have a computer system which can be expanded and improved, without hopeless incompatibility resulting.

/

(a) HARDWARE

Hardware expansion is made standard by the use of NASBUS. This is a bus standard, and this means that the signals between all different expansion boards will meet this standard. Note that the new NASCOM 2 computer plugs directly into NASBUS, without the need for a buffer board, which is an improvement. The NASCOM 2 still has the same screen formatting and the same Z80 CPU, so it is highly compatible with NASCOM 1.

(b) SOFTWARE - MONITORS

I hope that the article on monitors (later in this newsletter) explaining the development which has resulted in NAS-SYS, has answered many questions. Here is a brief summary. The following group of monitors are all highly compatible.

NASBUG T1: This was replaced by T2, and no one has a NASBUG T1 any longer. (We hope!)

NASBUG T2: Standard 1K monitor.

B-BUG: 2K monitor, fully compatible with T2. It will run all programs written for T2, without modification. It can read and write tapes in LOAD/DUMP format, like T2, or in READ/WRITE format. This monitor is not a NASCOM product.

NASBUG T4: 2K monitor, improved version of B-BUG. Same comments apply as for B-BUG. One tiny difference - SRLOUT routine was moved by one byte.

- - - - -

NAS-SYS 1 This new monitor is not directly compatible with the earlier monitors. It is called NAS-SYS, meaning NASCOM OPERATING SYSTEM, and has in fact been designed to provide for easy upward compatibility in the future. Quite a few earlier programs have been converted to NAS-SYS, (even including the 8K Basic!), and it has proved very easy in every case. When NAS-SYS and its full documentation and listing are available, I am sure it will be generally agreed that any inconvenience is made worth while by the enormous improvements. If a user has a large number of old programs, an old monitor can always be plugged in to run them. In fact, even NASCOM 2 could be run under NASBUG T2, if you really want to do this! In the future, all programs will be NAS-SYS compatible, and there won't be any problems.

(c) SOFTWARE - TINY BASIC FROM NASCOM

The 2K Tiny Basic runs under NASBUG T2, B-BUG or NASBUG T4. If NASBUG T2 is in use, programs are saved on cassette using LOAD/DUMP format. If B-BUG or T4 are plugged in, then the Basic automatically uses READ/WRITE format, which is, of course,

/

four times faster. The 3K Basic is the 2K Basic with an extra 1K EPROM to provide extra commands, so it works exactly like the 2K Basic. The 3K Basic, like the 2K Basic, requires some memory expansion to run, either 8K, 16K or 32K.

(d) SOFTWARE - TINY BASIC LEVEL A AND B

These Tiny Basic interpreters are not provided by NASCOM, and no information about compatibility is available. We believe that level A is a 2K Basic interpreter, which replaces the monitor program. Therefore, no monitor is available when level A is in use. Also, various features of the Tiny Basic have had to be left out, including FOR - NEXT loops. The only advantage is that no memory expansion is required to use it. Obviously only very short programs can be run, without memory expansion. Level B seems to be normal 2K Tiny Basic. It is not known how programs are stored on tape, or with which monitors it is compatible.

(e) SOFTWARE - 8K BASIC

The 8K Basic is on its way. It has been well tested, and really does exist and it works! It is coming from NASCOM in two versions - in ROM at E000 - FFFF with workspace and program starting at 1000, and also on tape, at 1000 - 2FFF, with workspace and program starting at 3000. Apart from the different location of the program area, these products are virtually the same. Programs can be saved on tape. Furthermore, NASCOM 8K Basic works with B-Bug, Nasbug T4 or NAS-SYS!!! (The tape version also works with Nasbug T1/T2). Only when using NAS-SYS are full program editing facilities available, making it very easy to change bits of programs easily and quickly. We (the INMC committee) recommend that you use every possible inducement to get your hands on NAS-SYS and the 8K Basic, because we think it makes the whole system far more useful, and very impressive even when compared to certain boring machines which come ready built in plastic boxes from the USA, at far greater cost.

(f) SOFTWARE - ZEAP

The ZEAP Editor/Assembler runs under NASBUG T2, or NASBUG T4. It is going to be released in an improved form to run under NAS-SYS.

(g) SOFTWARE - Z80 PROGRAMS FROM OTHER SOURCES

There are always problems when moving programs from one system to another, because of different I/O capabilities and different assignment of memory. However, the 2K and 8K Basic interpreters have been converted for NASCOM without great difficulty, and so have many other programs. NAS-SYS uses standard ASCII codes, which will help make future conversions easier. (Also, the 8K Basic will run most published Basic programs, often without any modification.)

(h) SOFTWARE -DISC OPERATING SYSTEM

NASCOM have announced their intention to provide the CP/M operating system, for disc users. This will open up the possibility of running hundreds of CP/M programs which already exist, including larger more powerful versions of BASIC, as well as other languages. We hope you will support the committee's belief that this is the way that we should go for large systems.

We hope that we have answered all the questions asked by Mr.Griffiths. If not, write back to us. All queries are welcome, although we can never hope to answer all of them.

The Editor

Dear Sir,

I thought NASCOM users might be interested in some additional Z80 opcodes that I have discovered. They all operate on IX or IY and their effectiveness hinges on the fact that IX and IY are functionally similar to HL. In general, instructions operating on HL will operate on IX or IY if preceeded by DD or FD. From this it can be deduced that the internal microcode of the Z80 addresses HL, IX, IY indirectly by a 2 bit register address pointer which is cleared at the start of each new instruction. When this pointer (call it P) is 00 then instructions operate on HL when it is 10 they operate on IX and when it is 11 they operate on IY. The effect of the instruction prefix DD is to set P to 10 and the effect of FD to set it to 11. Similarly, there must be other flip flops to select between the alternate register and accumulator sets.

Because of this any instruction normally accessing the H or L registers can be used to access the high or low order bytes of IX or IY if prefixed by DD or FD. This gives 80 new instructions. I have tested these on a NASCOM using a Zilog processor. I assume that similar instructions exist for rotate and shift, test, set, reset, which would give 124 more instructions. I have tested EX DE and IX and found that they do not work.

Yours faithfully,

W.P.Cockshott
Edinburgh.

Dear Mr.Cockshott,

Congratulations on finding the hidden bonus in the Z80. It is unfortunate that we can never be certain that every Z80 in the world has these instructions. Some might execute them unreliably.

There are 8 more hidden instructions which are prefixed by CB. These were found not to work as originally intended when the Z80 was built, so they were simply written out of the specification. See if you can find them and work out what they do wrong! Also, there may be some ED prefix instructions that have not yet been found. Good hunting.

The Editor.

Dear Sir,

I just had to write and congratulate you on Issue 2 of the INMC news. I enjoyed every page and every article and found the tips most invaluable to a striving amateur like myself. Your publication is a shining light in the NASCOM wilderness. Please! Please! keep it up.

P.S. When I have mastered the art sufficiently to contribute - I will have a go!

Yours faithfully

N.A.Lincoln
Loughborough.

Dear Mr.Lincoln,

What can we say except thank you. We will not be able to "keep it up" however, without some support from the vast membership. The four of use who make up the INMC committee wrote the last issue and this issue in their entirety and yet we all have full-time jobs. I am sure that it is not beyond each member's scope to write just one short article and send it to us. <u>If we can, then so can everyone else</u>. We look forward to hearing from you when you have "mastered the art".

Yours sincerely

The Editor.

GAMES COMPETITION RESULTS.

Thank you for your competition entries. We didn't get the quantity of response that we expected, but if we had I imagine the judging would have taken a week!

Anyway, one evening recently the INMC committee sat down with beer in one hand, sandwiches in the other and keyboard between teeth to judge the games that we had received. The programs had been entered, and stored on cassette during the previous week by our noble secretary. She double checked for errors and so we knew that any programs that didn't run were either incorrect or illegible.

So the judging began with "Burst the Balloon". In this game balloons are released, one at a time, from the bottom of the screen and make an erratic journey upwards, at the mercy of random breezes. The aim of the game is to burst as many balloons as possible by firing your gun, which is situated on the left hand side of the screen. This is an infuriating game as the balloons seem to tease you, pausing just below your line of fire, and then leaping past it just as you shoot. Quite entertaining, although we felt it could be improved by allowing movement of the gun.

We then came to two versions of "Life". Both interesting although unfortunately not necessarily for the correct reasons. We were a little uncertain whether or not to include these, as "Life" is a simulation rather than a game. Anyway, we had them on tape, so we tried them. The first one made clever use of different ASCII characters to give an effective display of 30x48. Various standard patterns were tried, and bred correctly. The speed of the program was impressive too. The second version, however, was not quite as successful, for whatever pattern we loaded seemed to make no difference and just resulted in a varied pattern moving across the screen! We could only assume that a transcription error had occurred somewhere along the line.

We continued with various other games. These included "Darts" which seemed to show promise, but also suffered from some bug; "Submarines" which allowed you to drop depth charges from your ship in a race against the clock to hit 9 submarines; and "Walled chase" a game for two players, with one chasing the other, but with invisible walls in the way.

"Lollypop lady trainer" appeared in the competition, and as many of you will have discovered, this was our "Mystery Program" in the last newsletter. From the same competitor we had "Road Race." This is a very entertaining game where you have to steer along a narrowing road in a bid to get home. There is a choice of three speeds, and the game ends with a comment related to the amount of damage that you did to your car. A must for learner drivers!

/

Two games had a "Space" theme. One was "Space Invasion", a game very similar to that seen in many amusement arcades. Very entertaining, and a game that gets progressively more difficult as you improve. We did manage to discover a way of cheating, but we'll keep quiet on that one.

The other space game was "Moon Lander." This game has appeared many times on one computer or another, and we have seen several versions written for the NASCOM. This version, however, we decided had a little extra and you will find a full description of it, and machine code listing elsewhere in this issue.

After each game we each awarded it points out of ten. At the end of the evening we added them up. The totals clearly indicated the winner as "Moon Lander," entered by Nick Ray but written by Arthur Norman and Chris Webster, with help from Philip Gladstone, Jim Cownie and Nick Ray. To them we will be sending a Zeap package. (We were thinking of sending them ½K of Zeap each, and not telling them who has got which bit!).

We also decided that, as Marcus Parker-Rhodes had got two games in the final list, he should be awarded a prize also, so a Super Tiny Basic set goes to him. Finally, to all those listed below we will be sending a selection of programs from the INMC library.

Thank you everybody for an entertaining evening.

Winner	Moon Lander	Nick Ray
Runners-up (in order)	Road Race	Marcus Parker-Rhodes
	Space Invasion	Graham Clarke
	Lollypop Lady Trainer	Marcus Parker-Rhodes
	Burst the Balloon	John Waddell
	Life	J. Haigh
Special Prize		Marcus Parker-Rhodes

Games Competition Winner - Moon Lander

Reproduced below is the object code listing of the winner of our games competition.

The aim of the game is to land a spacecraft on the moon's surface with a vertical velocity of less than 30, as near to the target (as indicated by the Distance parameter) as possible. However, you must not land on a boulder.

/

```
>  MOON LANDER PROGRAM
>T C50 FF0
  0C50 D5 EB 21 00 00 A7 ED 52 23
  0C58 D1 C9 21 0A 08 11 10 00 52
  0C60 3E 20 0E 10 06 30 77 23 B8
  0C68 10 FC 19 0D 20 F6 C9 EF 74
  0C70 D8 17 FC 9B FF F5 FF FE F3
  0C78 FF FF AF 01 2D 00 CB 7C A6
  0C80 28 04 CD 50 0C 41 E5 26 20
  0C88 00 D9 C6 6F 6F 26 0C 4E 91
  0C90 0C 28 29 23 46 23 E3 3E A6
  0C98 2F 3C 09 38 FC ED 42 E3 5E
  0CA0 D9 12 E6 0F 28 0E 67 78 A1
  0CA8 A7 28 05 1B 12 13 06 00 CE
  0CB0 13 D9 18 DB 84 20 F9 3E 76
  0CB8 20 12 18 F4 D9 7C A7 20 1E
  0CC0 05 1B 3E 30 12 13 E1 C9 29
  0CC8 E1 4E 06 00 23 ED B0 E9 B2
  0CD0 06 05 18 08 CD D7 0C CD 84
  0CD8 DA 0C 06 00 CD 35 00 10 E2
  0CE0 FB C9 0E 00 A7 F2 EA 0C 4D
  0CE8 0C AF FE 30 FA F2 0C 0C E1
  0CF0 3E 2F 5F 3E 0E 90 F2 F8 91
  0CF8 0C 0C AF FE 0F FA 03 00 E2
  0D00 0C 3E 0E 6F AF 67 57 CD 0E
  0D08 53 0F 19 11 0A 08 19 46 12
  0D10 3A 48 08 77 C9 3A 49 08 72
  0D18 A7 C0 67 6F 22 3D 08 22 EB
  0D20 43 08 18 14 CB 2F 00 00 9E
  0D28 6F 07 9F 67 CC 50 0C ED C6
  0D30 5B 46 08 19 7C 17 30 06 C8
  0D38 21 6F EF 22 48 08 22 46 9E
  0D40 08 C9 CD D0 0C 3E 02 CD D4
  0D48 4A 00 D9 11 00 00 D5 D5 33
  0D50 DD E1 FD E1 D9 06 08 C5 A5
  0D58 3E 01 CD 4A 00 DB 00 EE 84
  0D60 BF 21 7C 0F 16 00 58 19 5F
  0D68 5E EB 29 29 29 EB 4A 06 74
  0D70 00 50 1F 30 07 FD 09 DD 06
  0D78 19 D9 13 D9 11 08 00 19 95
  0D80 E5 01 AD 0F A7 ED 42 E1 E6
  0D88 38 DE C1 10 CA DD E5 E1 E9
  0D90 CB 3C CB 1D D9 D5 D9 D1 E4
  0D98 AF 8B 28 13 3E EA CD 74 83
  0DA0 0F C5 FD E5 E1 29 29 3E D4
  0DA8 FC CD 74 0F D1 18 04 57 45
  0DB0 5F 47 4F C5 DD 21 3A 08 B7
  0DB8 CD 2F 0E D1 DD 21 40 08 E6
  0DC0 CD 2F 0E 2A 43 08 11 F8 55
  0DC8 FF 19 22 43 08 3A 3F 08 0B
  0DD0 CD 24 0D 3A 45 08 CD 24 53
  0DD8 0D 11 CA 0B CD C8 0C 0A 83
  0DE0 44 69 73 74 61 6E 63 65 18
  0DE8 20 20 2A 3B 08 CD 7A 0C F5
  0DF0 CD C8 0C 08 20 48 65 69 DC
  0DF8 67 68 74 20 2A 41 08 CD A8
  0E00 7A 0C CD 24 0E 2A 3D 08 02
  0E08 CD 69 0F 3E 02 01 0D 09 B2
  0E10 CD 7E 0C CD 24 0E 2A 43 E1
  0E18 08 CD 69 0F 3E 02 01 0B BF
  0E20 5E CD 7E 0C CD 2A 0E CD B5
  0E28 2A 0E 3E 20 12 13 C9 3A F4
  0E30 47 08 17 38 32 DD 7E 05 6E
  0E38 CB 2F 83 DD 77 05 5F 07 82
  0E40 9F 57 DD 66 04 DD 6E 03 D9
  0E48 19 DD 74 04 DD 75 03 7C 95
  0E50 07 9F DD 56 01 DD 5E 00 73
  0E58 19 DD 74 01 DD 75 00 DD 00
  0E60 6E 02 8D DD 77 02 C9 AF 39
  0E68 18 D4 31 00 10 01 10 00 B4
  0E70 11 3A 08 21 DD 0F ED B0 7B
  0E78 CD 5A 0C 3E 3D 21 8A 0B EA
  0E80 77 11 8B 0B 01 2F 00 ED C9
  0E88 B0 21 0A 08 E5 CD 42 0D 7A
  0E90 E1 36 20 2A 41 08 CD 69 7E
  0E98 0F CD 5F 0F 7D F5 CD 5A 89
  0EA0 0F CD 69 0F 7D C6 20 C1 26
  0EA8 CD E2 0C E5 AF 81 C4 15 5F
  0EB0 0D 2A 41 08 11 C4 09 ED 09
  0EB8 52 30 D2 E1 CD 5A 0C 21 4F
  0EC0 0A 08 E5 CD 5A 0F 11 10 1C
  0EC8 01 19 11 F0 FF 7D E6 E0 33
  0ED0 6F 19 54 5D 22 7A 08 D9 94
  0ED8 21 CA 0B 11 C0 FF D9 06 8B
  0EE0 0C 4E 23 C5 06 04 EB E5 0A
  0EE8 C5 01 00 08 7E 23 E6 07 52
  0EF0 81 4F 10 F8 60 69 CD 69 D5
  0EF8 0F 7D 3C D9 E5 47 19 36 22
  0F00 7F 10 FB E5 DD E1 E1 23 40
  0F08 D9 C1 79 87 38 09 87 30 A9
  0F10 06 DD 36 00 0E 18 04 DD 3F
  0F18 36 00 5F 4F E1 23 EB 10 0A
  0F20 C5 C1 10 BD CD 42 0D E1 7F
  0F28 36 20 2A 41 08 CD 5F 0F 3B
  0F30 7D F5 CD 5A 0F 11 00 01 F9
  0F38 19 ED 5B 7A 08 A7 ED 52 10
  0F40 7D C1 CD E2 0C E5 78 FE A3
  0F48 20 20 62 79 A7 28 D5 E1 F7
  0F50 C3 78 0E CD 56 0F 29 29 2C
  0F58 29 C9 2A 3B 08 18 07 CD B2
  0F60 69 0F 54 5D 29 19 CD 69 10
  0F68 0F CD 6F 0F CD 6F 0F C8 E7
  0F70 2C CB 1D C9 06 FF 4F ED 9D
  0F78 52 D8 03 18 FA 6B 44 4A BF
  0F80 49 48 47 46 45 4B 24 2A 8B
  0F88 29 28 27 26 25 8A 83 89 F0
  0F90 88 87 86 85 84 46 23 81 27
  0F98 82 61 62 42 43 2B 06 80 22
  0FA0 60 40 41 21 22 6A 63 69 09
  0FA8 68 67 66 65 64 FE 0E 28 E9
  0FB0 16 2A 43 08 11 10 FF A7 11
  0FB8 ED 52 38 0B 11 D0 08 CD FF
  0FC0 C8 0C 02 4F 4B 18 10 E1 48
  0FC8 36 0C 23 36 2F 2B 2B 36 2D
  0FD0 5C 11 C1 FF 19 36 0C CD 3A
  0FD8 D4 0C C3 6A 0E 00 80 C1 43
  0FE0 A0 0F F0 00 38 4A 0C FE 1A
  0FE8 0A 20 4E 07 00 DD 3F 98 2A
.
> EXECUTE AT 0E6A
>
> NOTE THAT THE RIGHT HAND COLUMN
> OF THIS LISTING IS THE CHECKSUM
> AND MUST NOT BE TYPED IN!
>
```

The craft is controlled by means of the keyboard, which varies the rate of change of thrust. If no keys are pressed the thrust automatically decays to zero. All the keys may be used - the idea is to press in the direction you want thrust - dead centre is between the H and J keys and the force applied is proportional to the weighted distance from there.

You have a limited amount of fuel and should you run out the spacecraft will fall to the surface. You will lose all your fuel if you collide with the 'edge of the moon'. The spacecraft is represented by a bell (07) or if your fuel becomes exhausted the 'flames' go out and the spacecraft becomes a lower case 'o'.

As you approach the surface, the program gives you an expanded view showing the surface in detail, including boulders (which must be avoided).

An unhealthy landing will result in a minor explosion!

The game executes at OE6A. Note that the last column of figures in the listing is the checksum and should not be entered.

SO.. WHAT ABOUT ZEAP ??

ZEAP stands for Z80 Editor Assembler Package.

OK, so what is an assembler?

An assembler is a program which is used to take the 'donkey work' out of writing machine code programs. When you are converting your tediously written mnemonics into instruction codes (and fumbling through the book, 'cos you can't remember what they are) and at the same time assigning them to addresses, you are "assembling the program". Now an assembler does this for you, not only that, but it keeps track of where you put the subroutines and workspaces, converts lines of text into ASCII, and tells you when you started making up instruction mnemonics.

Fair enough, but what does the editor bit do?

Well most programs don't work straight off, because you got something wrong, or left something out, and these errors usually mean inserting or deleting some mnemonics, which of course sods law dictates that program will either be longer or shorter. Now the editor allows you to change, insert or delete at will, and when you re-assemble, all the addresses assigned to subroutine etc. will all come out in the right places; because the assembler part keeps track of where you put them. No more ploughing your way through 2K of object code, changing all the subroutine calls, just because you left out two bytes at the start.

Well, I've seen the output of ZEAP, but I still get lost, why is that?

Firstly, you've got to remember that ZEAP is only about 2¾K long and that is pretty short as assemblers come. ZEAP was designed to run in a NASCOM with the minimum (8K) expansion possible, and because of this a few short cuts were taken. The short cuts all concern what are known as 'assembler directives' which are instructions to the assembler (not program instructions to the Z80) to do certain things with certain bytes of the program. For instance a 'defined byte directive' (DEFB), now this directive instructs the assembler to put the byte you have defined into the program. So DEFB £1A means put 1A into the program (the £ sign means the byte is expressed in hexadecimal). The assembler will assemble it like this; 0C50 is the address by the way 0010 is the line number:

```
0C50 1A          0010          DEFB £1A
```

Now for the short cut bit - suppose you have a string of defined bytes, followed by an instruction, the assembler would assemble them like this:-

```
0C50  1A          0010          DEFB £1A, £2C, £44, £5F
0C54  3E1F        0020          LD   A, £1F
```

Notice that ZEAP only displayed the first defined byte, but the addresses worked out right. This is because ZEAP assembled them, it just does not display them. Notice also the second instruction, there is no space between the 3E and the 1F. Remember, if you were typing that into a NASCOM, you would have to put a space between the two.

The same point about defined bytes applies to defined words (DEFW) and defined messages (DEFM) thus;

```
0C50 1A          0010          DEFB £1A, £2C, £44, £5F
0C54 350C        0020          DEFW £0C35, £0035, £0124
0C5A 54          0030          DEFM /THATS THE LOT/
0C67 3E1F        0040          LD   A, £1F
0004             0050          DEFS 4
0C6D 00          0060          NOP
```

Notice that with a defined space (DEFS) ZEAP simply put 0004 in the address field, and advanced the address counter by four.

So the basic message is, when loading programs assembled on ZEAP, watch the address counter.

Of course, lets not create a wrong impression, these short cuts in the display only occur on the monitor screen or on a printer, if you were using ZEAP yourself, these deficiences need not cause problems, because you would ultimately assemble the program to memory or tape, in which case these bytes would be correctly assembled even though they are not displayed.

/

Other points to remember;

Anything following a semicolon is a comment, and is ignored by the assembler.

The other directive ZEAP accepts is 'ORG', which is the location where ZEAP is to start the assembly. Sometimes there may be several ORG directives where parts of programs are assembled at different locations.

An 'EQU' is used where lable or symbol not included in the program has been used. Most commonly these refer to routines in NASBUG/B-BUG which are used as part of the program. As these do not appear in the program, you have to tell ZEAP where they are. Also symbols like CR (for carriage return/new line) may be defined by an EQU as, you never know, someone may come along with a new monitor which redefines the symbols. In this way, ZEAP can be used as a sort of 'cross assembler'.

MEMORY PLAGUE !!

What is it?
How do I identify it?
How do I cure it?

"Memory Plague" is a euphamism for the unexplained failure of NASCOM memory boards. "Memory Plague" seems to affect about 10% of NASCOM issue 1 memory boards, and its causes are, to say the least, obscure. It would seem that noise caused by switching transients from the data latches (81LS97) and the address multiplexers (74LS157) may be breaking onto the data bus via IC2. Poor board layout would seem to be primarily to blame, in conjunction with chips that just happen to be on the low side of average for noise immunity.

Identifying "Memory Plague" is not as easy as it would at first appear because a memory suffering from mild plague will pass both memory test programs in the construction manuals with flying colours and will run Tiny Basic without problem. This is because both memory tests and Tiny Basic are loading operands to memory, and not actually executing M1 (op-code fetch) cycles. M1 cycles are more critical on timing and hence more susceptible to corruption caused by noise. Likewise "Memory modify" and "Copy" commands are unlikely to cause problems except in severe cases. So the only thing likely to reveal "Memory Plague" is a program with lots of M1 cycles and filling a sizeable chunk of memory. ZEAP is ideal for this. If you do not have ZEAP, then write a simple relocatable program that may be copied throughout memory, finishing with the printing of an * then looping back to the start. Leave this running as long as possible (preferably overnight). If the program "crashes", then provided you have eliminated next-doors' fridge (never your own, of course) as the cause, then it's likely that "Plague" has struck.

/

So to the cures:

None of these is technically elegant, but they do work. They should be tried in order until the problem is cured. Don't go in for overkill, as this is unnecessary, and undesirable.

1). Go for a National 81LS97 in the IC2 position (AMD devices seem to have lower noise immunity although AMD deny this). You have 7 81LS97s to play with, 3 on the NASCOM, 1 on the buffer, and 3 on the memory, one at least is likely to be made by National. Swap these ICs about for the best results in the IC2 position.

2). Grid off the Ground and +5 volt supplies. On the underside of the pcb it will be noted that the GND and +5 rails supplying the TTL ICs terminate at the end of each row. Wire links can be fitted to connect these rails to the equivalent rails supplying the RAM chips, thus completing the "grid" on the power supply rails, thereby reducing power supply noise. Take care not to short out the power supply rails by "gridding" to the wrong tracks.

3). The 74LS04 on the buffer board may be replaced with 74S04. Bit of a naughty one this, as far as loading goes, but it does tidy up the MREQ waveform.

4). On ICs 4-11 only, fit a 4K7 resistors from pin 9 to pin 14 of each chip, thus pulling the outputs of the RAMs to +5V.

5). On ICs 4-11 only, in addition to 4 (above), fit 47pF ceramic capacitors from pin 14 to pin 16, thus producing a time constant on the RAM output.

Various combinations of these cures have been tried with 100% success on the few boards that have come our way, and although not 'elegant' solutions have transformed recalcitrant RAM boards into perfect working members of the species.

Please write to the INMC if you have come across any other oddities in the RAM or buffer boards.

STOP PRESS!

We have recently discovered that noise on the NASCOM 1 itself can cause problems with expansion, but fortunately this is easily cured. If you look at the corner of the board where the modulator is situated you will see an issue number. If it is "ISS.C" ignore these comments. Otherwise you may find it worthwhile to add a few links to the back of the board along the long edge of the board where the power supplies are connected. The links should "bus up" the ground and 5V supply rails and the easiest way to do this is to connect the ground of each decoupling capacitor at the edge of the board to the next decoupling capacitor at the edge. Similarly connect the 5V sides. Be careful not to get them twisted!!

If you are at all uncertain about any of these modifications, please contact your distributor, NASCOM, or the INMC. If in doubt, stop. If you return the system for repair you should include the NASCOM, memory and buffer boards to ensure that the system is totally operational when returned to you.

SPECIAL OFFERS

We have received offers from the two companies below for special reductions to INMC members.

Cassettes

D.J.M.Services offer quality C12 cassettes manufactured by Racal Zonal Ltd. Their normal prices are 5 for £3.00, 10 for £5.00 and 50 for £22.50. These prices include postage and VAT, CWO only. INMC members qualify for 10% discount on these prices. All queries to:-

D.J.M.Services
82 Hilden Park Road
Hildenborough
Kent
TN11 9BN TEL: 0732-832815/357721

Soldering Irons.

Future Electronics offer Adamin Model 15 miniature soldering irons. These irons are described as being "probably the slimmest and lightest available and make construction of kits like the NASCOM 1 easy, giving professional results with little effort". A 1/16" bit is supplied, and a full range of 14 bits is available in copper or long-life forms. A catalogue of soldering products will be sent to all enquiries and orders, and goods will be despatched first class, by return of post. Future Electronics also hope to shortly be offering hand tools.

The recommended retail price of these irons is £4.27 inc.VAT - INMC members can obtain them for £3.50 including p & p and VAT.

All enquiries to: Future Electronics
Unit B1
Parkhall Trading Estate
Martell Road
London SE21 TEL: 01-660-0747

VAT Please note that both of the above offers were made prior to the recent budget - please check with the companies for details of changes.

APOLOGIES

In Issue 2 of INMC News we referred to a NASCOM Club being started by Frank Butler in Mansfield. We said that he would like to hear from other NASCOM users with a view to starting a club in North Wales. This should of course said North Notts and we apologise for any inconvenience that this might have caused.

For those of you to whom this is the first newsletter, the address again is:

Frank M Butler,
8A Church Side,
Mansfield,
Notts.
NG18 1AD Telephone: 0623-29237

SOFTWARE SECTION

1. Beginners' Corner

How to move data around-quickly!

I have seen a program which put a message on the screen with a piece of code like this:

```
LD    HL, address on Screen
LD    A, "M
LD    (HL), A
INC   HL
LD    A, "E
LD    (HL), A
INC   HL
```

and so on ..... and on ..... and on .....

This works very well, but it takes a long time and a lot of program just to do this simple task. You should see the last newsletter for an easy way to display a message using code EF, then the message, then a zero.

But quite often, one wants to move lots of data around and not just output a message at the current cursor address. By far the easiest way is by the LDIR instruction. Here is an explanation:

1. Set HL to the address of the data you want to move.

2. Set DE to the address you want to move it to.

3. Set BC to the length of the data you want moved.

4. LDIR instruction - this does the work.

For example, suppose you had 48 bytes of data at address OE8O, and you wanted to put this on the top line of the screen. The address of the top line is OBCA.

SO:

```
1.  21 8O OE        LD  HL,OE8O  (from)
2.  11 CA OB        LD  DE,OBCA  (to)
3.  O1 3O OO        LD  BC,OO3O
```

(Length of data, OO3O = 48 decimal)

```
4.  ED  BO          LDIR
```

/

2. Plea for help

In the last newsletter we said that it was your newsletter. Please will you send in helpful tips to this software section of the newsletter. I promise to consider printing anything that isn't actually wrong!

3. Correction

The last newsletter said that to end a program you could jump to PARSE. I did not write that - it was edited by our super software? expert (Ta for the compliment Richard. Just wait until you want some hardware sorted!! D.H.) and INMC chairman David Hunt, who got it wrong! This is a complicated situation, and the ONLY 100 percent safe way that ALWAYS works is to jump to address 0. If you don't want to clear the screen, put in a 76 (HALT), and press RESET to carry on when you are ready.

4. Solution to Puzzle (in the last newsletter)

Nobody sent in a correct explanation of the problem, but I give half marks to those who made the program work by putting in code B7, which is OR A, before the INC A. The reason for the problem is that although INC A and ADD A, 1 might seem the same, INC A does not set the Carry flag, or change it at all. Once the carry flag gets set, the DAA instruction gets completely confused! To understand all this, use the S command to step through, keeping an eye on the Carry flag. Also, have a look at the Z80 programming manual, under INC, ADD, and DAA. If you don't understand the table describing DAA, you are not alone! By the way, perhaps the Z80 should set Carry when INCing - but if it did it wouldn't be 8080 compatible, so you really have to blame Intel for the 8080 design. Anyway, these little quirks make programming interesting.

5. NASCOM Monitor Programs - a personal view

Here is a brief history of the monitors - also known as operating systems - which have controlled my NASCOM 1 computer. By now I suspect that there maybe quite a few confused NASCOM owners, because so much change has occurred in the last year, so this may also help to clear up misunderstandings.

In the beginning I had NASBUG 1. This lasted about two minutes, then I blew it up - it got very hot, I burnt my fingers, and pulled its little legs off getting it off the board! I then had a replacement NASBUG 1, which worked, except for the tape loading problem. This was cured when NASBUG 1T2 was produced, and most NASCOMS probably use this monitor, which is the only standard 1K monitor.

Eventually the magic of NASBUG wore off, and frustration with slow tape loading set in, so I wrote a monitor called B-BUG, which was NASBUG compatible but with extra features, the most important of which was the read and write commands, for tape loading at four times the speed. B-BUG used about 2/3 of the second 1K EPROM, so there was still room for improvement.

/

Next I wrote NASBUG T4, which used up all the rest of the available 2K, and contained many improvements and new features. Most important, it was still NASBUG compatible, so programs written for the other monitors would run under it.

At this point it was apparent that there was no further scope for significant improvement, because of the need to retain complete compatibility with the original old NASBUG 1. At this time I got my memory expansion and ZEAP, and re-assembled NASBUG T4 using ZEAP, which was fun because all the previous work had been done by hand in machine code! NASBUG T4 is the best monitor currently available for the NASCOM 1. But .....

Some of you have probably heard about NASCOM 2, and you may be wondering about its monitor. What happened was this: Using my NASCOM 1, with 32K board and ZEAP, I developed a completely new and somewhat incompatible monitor, by starting with NASBUG T4 and steadily improving it. Several major changes were made:

1. The Cursor on the screen is made to blink on and off. It is possible to move it anywhere on the screen. Lines on the screen can be edited, and even characters inserted and deleted by moving the rest of the line to left or right.

2. The line to be processed is always the line where the cursor was when enter (or newline) is pressed.

3. Output starts at the top and scrolls down.

4. The extended NASCOM 2 keyboard is fully supported including up, down, left and right arrows to move the cursor. (The old keyboard can do these too.)

5. The memory used by the monitor has to occupy addresses OCOO to OC7F.

6. All monitor commands perform comprehensive error checking on the line entered. Also, all routines are callable by the user program, NOT by a CALL instruction which has an actual address, but by a restart instruction followed by a (magic) subroutine number.

7. Various errors have been corrected, and lots of extra features added. There are very full input/output options, to make it easier to use printers, teletypes, etc. Also, the machine uses ASCII codes for carriage return, line feed, back space and clear screen.

This monitor is called NAS-SYS 1, and is the only monitor for the NASCOM 2. No changes are planned, and it will be completely standard. Even if changes are made there are no compatibility problems because of the use of subroutine numbers (see 6 above).

Now the crucial question - can NAS-SYS be used on a NASCOM 1, making it very like NASCOM 2? - the answer is yes, and I use my NASCOM 1 under NAS-SYS most of the time, BUT - will NASCOM provide if for you? Marketing Director, Kerr Borland, says "No plans at present." If you want it, why not write and tell him what you think about it - he might pay attention (he thinks I'm crazy).

/

By the way, the 8K Basic works best under NAS-SYS, because you can correct lines of program using its editing facilities. Otherwise, you have to re-enter the whole line in error. I have got a version of the 2K Tiny Basic which runs under NAS-SYS, but it's not really worth bothering with - the 8K Basic is so tremendous. I am patiently waiting for ZEAP to be converted to run under NAS-SYS - come on!!

Send in any queries about monitors and what they do or don't have, and I will try to remember the answers and print replies, Good Luck out there!

Richard Beal

Note: Address contributions to "Software Section, INMC"

Idea for a Program

Suppose you have a program from a magazine which is assembled to run at address 0000-0100, and it uses a data area at F000-F020. You want to run it on a NASCOM, with the program at 0C80-0D80 and the data area at 0E00-0E20. Now you could convert all the addresses, effectively reassembling the program. But suppose you don't have the source listing, and don't even know that it uses a data area. It would still be possible to disassemble and examine the program, but this would take a long time.

What I propose is a program to watch over the program that is being run. The unaltered program would be loaded into the NASCOM at 0C80. The control program would be told:-

All addresses 0000-0100 are to be converted to 0C80-0D80. Any other addresses are to be reported out before being executed.

The control program would then be run. It would look at each instruction in turn and then execute it. Addresses in instructions would be examined, and depending on their value, either left alone, be adjusted according to instructions given to the control program, or else be reported out and the program paused.

This suggested control program would also be excellent for debugging ordinary programs. For example, on an unexpanded NASCOM, it would be told: Run this program. Report any attempt to use an address in the range 1000-FFFF.

The control program could carry out the single step execution of the other program by copying the instruction to another part of RAM and following it with a RET, and then calling it as a subroutine. Or perhaps NMI single step feature could be used.

Please send in any ideas for how to do this, or indeed if it is possible.
Better still, try writing a program to do it!

Richard Beal
Software Co-ordinator
INMC Committee

SOFTWARE LIBRARY

Below is a list of programs that are now available from the INMC Software Library. Copies cost 8p per sheet, and the number of sheets for each program is given alongside its description. Please also enclose postage, as detailed on the enclosed order form.

For a variety of reasons, we have decided to remove the previous list of 12 programs from the library, and for simplicity, this list starts at number 20. Therefore, please do not write and ask us about numbers 1 to 19, they don't exist.

Program No	Description	No. of sheets
20	Robots Approx 900 bytes Six homicidal robots chasing you across a field of electrified pylons. The robots head directly towards you and so by careful movement you can persuade them to electrocute themselves on the pylons. Speed and difficulty selectable from cinch to impossible! Good instructions, teletype assembly listing, no comments. By H Birkett	8
21	T.V. Test Pattern 26 bytes Displays test pattern of "+" symbols on screen to set up colour T.V. Well commented. By R Cogliatti	1
22	NIM Approx 700 bytes The classic game of Nim Nicely commented with good instructions. By N A Purver	15
23	Attack Approx 350 bytes Shoot the descending aliens before they get you! Commenting fair. By M R Perry	6
24	Octal to Hexadecimal Convertor Approx 180 bytes Rejects all illegal input characters. Well commented. By J Hill	3
25	Space-Invasion Approx 900 bytes Very similar to the game seen in many amusement arcades. As you get better, it gets harder. Addictive. Well commented. By G Clarke	15

Program No.	Description	No. of Sheets
26	Digital Clock Approx 130 bytes Simple 24 hour digital clock (hours, mins., secs.). Nice demo of arithmetic and counter manipulation. Well commented but ties up processor 100%. By B C Winch	4
27	Roadrace Approx 900 bytes Steer your car home with minimum damage. Choice of speeds. Addictive. Well commented. By M Parker-Rhodes	11
28	Decimal to Hexadecimal Convertor Approx 120 bytes Nicely commented. Poor input validation. Positive numbers only. By G Harriman	2
29	Life Approx 750 bytes The ubiquituous simulation of life, using special characters to give an expanded Universe. Works well, very fast, good instructions, well commented. By J Haigh	8
30	Moon-Lander Approx 1000 bytes! Ingenious moon-landing program using keyboard as "crashpad joystick". Below 2500 ft display expands to terrain map, and when you land, look out for those boulders. Well commented, good instructions, teletype assembly listing. INMC games competition winner - object code listing in INMC issue 3. Submitted by N Ray	19
31	Compact Ascii Editor Approx 130 bytes Crude editor allowing Ascii characters to be located and changed. Well commented, good instructions. (Nice demo, not very practical). By A Fountain	2
32	Carre Chinois We have no idea what this game is or what it does (we ain't French is we?) Totally written (and presumably (?) beautifully commented) in French. By G Bochent	21

Program No.	Description	No. of Sheets
33	Lollypop Lady Trainer Approx 800 bytes See if you can get the chickens across the road without getting them run over. Well commented. By M Parker-Rhodes	9
34	Random I-Ching Character Generator Approx 180 bytes Great mystical significance (for those who understand these things). Nicely commented. By C Blackmore	4
35	Walled Chase Approx 490 bytes For two players, one chases the other but there are invisible walls in the way, which appear when you hit them. Fascinating game, well commented By S Montgomery-Smith	12
36	Sub-Search Approx 300 bytes Ship traverses screen dropping random depth charges on randomly moving submarines. By T Bailie	6
37	Random Display 37 bytes Displays random pattern of asterisks. Nicely commented. By G Harriman	1
38	Submarines Approx 500 bytes Hit the randomly displayed submarines with your steerable depthcharges. Good instructions, no comments. By N D Wallbridge	7
39	Burst the Balloon Approx 600 bytes Shoot the balloons as they make their way up the screen at the mercy of random breezes. Well commented, good instructions By J Waddell	9

FOUR FREE PROGRAMS!

As a special bonus, here are four programs which are fun to have. We hope to include proper assembler listings of these programs in the software library soon, so that you will be able to see how they work. Some of you who have bought games tapes from Henry's Radio will have seen some of them before, but there should be some which you haven't yet seen.

You should find all the programs self explanatory, given the notes printed with them. The last column of numbers on the listings is a checksum, so don't type it in!

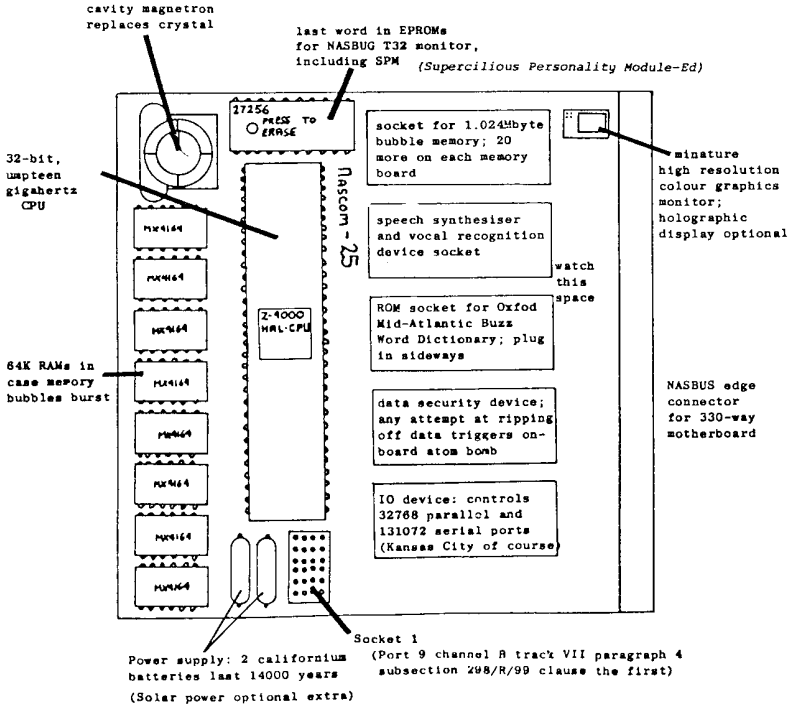
The programs will run under NASBUG T2, B-BUG or NASBUG T4. We hope to reassemble them for NAS-SYS one day.

One point about REACT. It uses its own special keyboard scan routine, and it is essential that the spare keyboard input pin be tied to +5V, not to earth. (This is the same modification required for the Tiny Basic.) Connect pins 7 and 8 to pin 16 of the keyboard socket to solve the problem. This prevents the spare input lines floating.

To play OTHELLO you need to know the rules, but you may be able to deduce them since the program doesn't let you cheat and shows you where you can move. It scores the game and provides quite good opposition. Remember to press the space bar to let it move. It can also simply supervise a two human player game.

Richard Beal
Software Co-ordinator
INMC Committee

STOP PRESS: This preliminary board layout was found by our own Investigative Journalist on the desk of NM's ace designer Heath Robinson-Crusoe. Concealed beneath a copy of International Times was also a breadboard prototype, but this disintegrated before it could be photographed. Robinson-Crusoe was unavailable for comment, but sources close to the industry suggested that the device was actually available some time last year.



```
   REACT  BY DAVID HUNT
>T C50 F00
  0C50 31 FF 0F CD 00 0F D9 06 56
  0C58 05 D9 21 78 0C 18 01 D9 D9
  0C60 11 4F 0B 01 25 00 ED B0 9A
  0C68 CD 40 02 CD 40 02 D9 10 7B
  0C70 EE D9 CD 1D 0F C3 31 0D 3D
  0C78 22 52 45 41 43 54 22 20 57
  0C80 74 65 73 74 73 20 79 6F C7
  0C88 75 72 20 72 65 61 63 74 AA
  0C90 69 6F 6E 73 20 21 20 20 D6
  0C98 41 66 74 65 72 61 20 20 37
  0CA0 72 61 6E 64 6F 6D 20 20 6D
  0CA8 64 65 6C 61 79 20 61 20 64
  0CB0 73 69 6E 67 6C 65 20 20 7E
  0CB8 64 69 67 69 74 20 77 69 D5
  0CC0 6C 6C 62 65 20 64 69 73 CB
  0CC8 70 6C 61 79 65 64 2C 20 9F
  0CD0 79 6F 75 20 6D 75 73 74 22
  0CD8 20 70 72 65 73 73 20 74 C5
  0CE0 68 65 20 73 61 6D 65 64 E3
  0CE8 69 67 69 74 20 20 6F 6E BE
  0CF0 20 74 68 65 20 6B 65 79 C6
  0CF8 62 6F 61 72 64 20 61 73 00
  0D00 20 73 6F 6F 6E 20 61 73 E0
  0D08 20 79 6F 75 63 61 6E 2E F2
  0D10 20 50 72 65 73 73 20 61 CB
  0D18 6E 79 20 6B 65 79 20 77 0C
  0D20 68 65 6E 20 79 6F 75 27 0C
  0D28 72 65 20 72 65 61 64 79 41
  0D30 2E 21 58 0D 11 5C 0D 7D E8
  0D38 FE 5B 20 04 2B 2B 2B 2B 6E
  0D40 23 7B FE 65 20 0A 1B 1B AE
  0D48 1B 1B 1B 1B 1B 1B 1B 1B 20
  0D50 13 CD 69 00 30 E1 18 0E DD
  0D58 02 03 04 05 30 35 32 33 3D
  0D60 34 35 36 37 38 39 7E 32 64
  0D68 9D 0D 1A 32 71 0D 18 04 05
  0D70 37 38 92 07 CD 00 0F 21 82
  0D78 CA 09 22 18 0C EF 41 6E 3C
  0D80 79 20 73 65 63 6F 6E 64 A2
  0D88 20 6E 6F 77 20 20 20 3E A7
  0D90 00 CD 1D 0F 06 80 CD 35 1E
  0D98 00 10 FB D9 06 04 D9 06 72
  0DA0 FF CD 69 00 38 08 10 F9 2B
  0DA8 D9 10 F3 D9 18 20 CD 00 6F
  0DB0 0F 21 DA 09 22 18 0C EF 05
  0DB8 57 41 49 54 20 46 4F 52 01
  0DC0 20 49 54 20 21 20 21 00 0C
  0DC8 CD 1D 0F C3 9F 0F 21 71 D1
  0DD0 0D 7E 32 DF 09 11 00 00 93
  0DD8 3E 55 3D 20 FD 7B C6 01 14
  0DE0 27 5F 7A CE 00 27 57 DA 13
  0DE8 D9 0E 3E 02 D3 00 AF D3 71
  0DF0 00 06 08 3E 01 D3 00 AF CC
  0DF8 D3 00 DB 00 FE FF 20 04 D4
  0E00 10 F1 18 D4 ED 53 72 0D BA
  0E08 CD 69 00 32 70 0D 21 71 8D
  0E10 0D BE CA 54 0E CD 00 0F F1
  0E18 21 DA 08 22 18 0C EF 4E AC
  0E20 6F 74 20 76 65 72 79 20 17
  0E28 63 6C 65 76 65 72 00 CD 84
  0E30 1D 0F CD 23 0F 21 D9 09 6C
  0E38 22 18 0C EF 74 6F 20 67 E5
  0E40 65 74 20 69 74 20 57 52 ED
  0E48 4F 4E 47 20 21 00 CD 1D 65
  0E50 0F C3 63 0F CD 00 0F 21 9F
  0E58 DE 08 22 18 0C EF 43 6F 33
  0E60 72 72 65 63 74 2C 00 CD 87
  0E68 1D 0F CD 23 0F 21 CA 09 95
  0E70 22 18 0C EF 42 65 74 74 42
  0E78 65 72 20 74 68 61 6E 20 48
  0E80 30 2E 36 20 73 65 63 6F EC
  0E88 6E 64 73 20 69 73 20 67 5E
  0E90 6F 6F 64 2C 20 31 20 69 E6
  0E98 73 20 61 76 65 72 61 67 AF
  0EA0 65 2C 20 20 6D 6F 72 65 32
  0EA8 20 74 68 61 6E 20 31 2E 00
  0EB0 33 20 69 73 20 68 6F 70 54
  0EB8 65 6C 65 73 73 3B 20 68 A5
  0EC0 61 64 20 74 6F 6F 20 6D 92
  0EC8 75 63 68 20 62 6F 6F 7A F0
  0ED0 65 3F 00 CD 1D 0F C3 63 A1
  0ED8 0F 21 CA 09 22 18 0C EF 1E
  0EE0 54 68 61 74 20 74 72 69 EE
  0EE8 63 6B 20 77 6F 6E 27 74 D3
  0EF0 20 77 6F 72 6B 20 21 00 22
  0EF8 CD 1D 0F C3 9F 0F 00 00 70
  0F00 3E 1E CD 3B 01 CD 1D 0F 6D
  0F08 21 DB 0B 22 18 0C EF 04 57
  0F10 20 04 20 52 45 41 43 54 D2
  0F18 20 04 20 04 00 2A 18 0C BD
  0F20 36 20 C9 21 56 09 22 18 08
  0F28 0C EF 79 6F 75 20 74 6F 92
  0F30 6F 6B 20 20 20 20 20 20 D9
  0F38 20 73 65 63 6F 6E 64 73 56
  0F40 00 CD 1D 0F 3A 73 0D 21 23
  0F48 60 09 22 18 0C CD 44 02 19
  0F50 3A 72 0D CD 44 02 CD 1D 15
  0F58 0F 21 60 09 7E 2B 77 23 43
  0F60 36 2E C9 21 8A 0A 22 18 8B
  0F68 0C EF 44 69 73 70 6C 61 CF
  0F70 79 65 64 20 20 20 3E 00 5F
  0F78 CD 1D 0F 21 0A 0B 22 18 F0
  0F80 0C EF 48 65 79 20 70 72 B5
  0F88 65 73 73 65 64 20 3E 00 09
  0F90 CD 1D 0F 3A 71 0D 32 99 1B
  0F98 0A 3A 70 0D 32 19 0B 21 0F
  0FA0 8A 0B 22 18 0C EF 50 72 3B
  0FA8 65 73 73 20 61 6E 79 20 8A
  0FB0 6B 65 79 20 74 6F 20 74 9F
  0FB8 72 79 20 61 67 61 69 6E D2
  0FC0 2E 00 CD 1D 0F CD 69 00 2C
  0FC8 30 FB C3 50 0C 00 00 1F 40

EXECUTE AT 0C50
```

OTHELLO BY RICHARD BEAL (Game against Computer)

```
TC50 FE8
0C50 21 E7 0F 3E 08 CD 7A 04 04
0C58 C6 30 32 54 0B 3E 08 CD FE
0C60 7A 04 C6 30 32 55 0B C9 38
0C68 20 31 32 33 34 35 36 37 00
0C70 38 20 31 2E 2E 2E 2E 2E EB
0C78 2E 2E 2E 31 32 2E 2E 2E FB
0C80 2E 2E 2E 2E 2E 32 33 2E 05
0C88 2E 2E 2E 3D 2E 2E 2E 33 18
0C90 34 2E 2E 2E FF 00 3D 2E C4
0C98 2E 34 35 2E 2E 3D 00 FF 03
0CA0 2E 2E 2E 35 36 2E 2E 2E 2B
0CA8 3D 2E 2E 2E 2E 36 37 2E 44
0CB0 2E 2E 2E 2E 2E 2E 2E 37 35
0CB8 38 2E 2E 2E 2E 2E 2E 2E 3E
0CC0 2E 38 20 31 32 33 34 35 51
0CC8 36 37 38 20 4D 6F 76 65 30
0CD0 20 4E 6F 20 20 31 20 34 7E
0CD8 20 63 68 6F 69 63 65 73 E2
0CE0 20 FF 20 74 6F 20 6D 6F 0A
0CE8 76 65 9D 0C 02 02 00 00 7C
0CF0 01 09 0A 0B F5 F6 F7 FF FC
0CF8 00 58 08 94 00 00 00 00 F8
0D00 CD C8 0D 21 D4 0C CD 80 FD
0D08 0D 00 00 00 21 E1 0C 7E AE
0D10 2F 77 CD 45 0E 01 0D 0A FB
0D18 11 68 0C CD B0 0D 21 D6 2B
0D20 0C 06 02 CD 94 0D 20 1C EB
0D28 CD 90 0F 21 FF 0C 34 7E 7F
0D30 3D CA 03 0D EF 20 47 61 0B
0D38 6D 65 20 4F 76 65 72 21 F4
0D40 00 C3 88 0F AF C3 0D 0F 35
0D48 00 EF 20 52 6F 77 20 43 FF
0D50 6F 6C 3F 00 CD DB 01 21 41
0D58 54 0B CD 70 0D DA 9E 0E 94
0D60 21 68 0C 06 00 09 CD 80 5E
0D68 0E CA 9E 0E C3 03 0D 00 CC
0D70 CD A0 0D D8 CD 8C 0D CD 02
0D78 A0 0D D8 81 4F C9 00 00 A3
0D80 3E 39 23 34 BE D0 36 30 4F
0D88 2B C3 88 0E 17 4F 17 17 AD
0D90 81 4F C9 00 AF B7 C3 90 EF
0D98 0E 00 23 C6 D0 10 F6 C9 3B
0DA0 7E 23 FE 20 28 FA D6 31 95
0DA8 D8 FE 09 3F 3C C9 00 00 D7
0DB0 CD B2 0E 2A F9 0C C5 1A 58
0DB8 13 77 23 36 20 23 10 F7 F2
0DC0 0E 2C 09 C1 0D 20 EF C9 B6
0DC8 21 FE 0C 7E 2F 77 32 D0 26
0DD0 0E 00 21 73 0C 01 2E 08 C2
0DD8 3E 08 71 23 3D 20 FB 23 3A
0DE0 23 10 F5 21 20 30 22 D4 7C
0DE8 0C 22 D6 0C AF 32 FF 0C F1
0DF0 67 00 32 E1 0C 2F 6F 22 43
0DF8 94 0C 6C 67 22 9E 0C C9 0D
0E00 AF 32 FC 0C 11 F0 0C 1A 1E
0E08 13 B7 20 05 3A FC 0C B7 FE
0E10 C9 4F 17 9F 47 E5 CD 20 05
0E18 0E CC 30 0E E1 18 E8 00 1F
0E20 09 3A E1 0C 2F BE C0 09 14
0E28 BE 28 FC 2F BE C9 00 00 CE
0E30 22 FB 0C 3A FD 0C B7 C0 21
0E38 3A E1 0C 2F B7 ED 42 BE 40
0E40 2F 77 28 F7 C9 AF 2F 32 EC
0E48 FD 0C 21 20 30 22 D6 0C 04
0E50 21 73 0C 0E 08 06 08 C5 E7
0E58 01 3D 2E 7E B8 28 04 B9 ED
0E60 20 10 70 CD 00 0E 28 0A 1B
0E68 36 3D E5 21 D6 0C CD 80 1E
0E70 0D E1 23 C1 10 E1 23 23 87
0E78 0D 20 DA AF 32 FD 0C C9 40
0E80 3E 3D BE CA 00 0E BF C9 27
0E88 34 7E FE 21 C0 36 31 C9 57
0E90 CD 8C 0D 7E FE 20 20 02 C2
0E98 C6 10 81 C3 9A 0D EF 1E 74
0EA0 20 45 72 72 6F 72 21 00 F9
0EA8 06 80 CD 35 00 10 FB C3 0C
0EB0 15 0D 21 D0 0E CB FE 00 A8
0EB8 EF 1E 00 21 CA 0B 22 18 03
0EC0 0C EF 2A 20 4F 54 48 45 43
0EC8 4C 4C 4F 20 2A 20 43 3D A7
0ED0 80 20 20 53 63 6F 72 65 9A
0ED8 3A 20 20 FF 20 20 30 20 EF
0EE0 20 20 20 80 20 20 30 1F 5D
0EE8 00 C5 D5 06 64 1A 13 FE 25
0EF0 FF 20 06 21 E5 0B CD 80 81
0EF8 0D FE 00 20 06 21 ED 0B 50
0F00 CD 80 0D 10 E8 D1 C1 3E 31
0F08 20 32 4A 0B C9 21 FF 0C B3
0F10 77 3A FE 0C 4F 3A E1 0C 50
0F18 B9 C2 49 0D CD 90 0F 21 85
0F20 DD 0F 11 A0 0F 46 23 E5 29
0F28 AF 32 EC 0C 26 0C 1A FE 5A
0F30 00 20 01 76 13 6F 7E FE D4
0F38 3D 20 0E AF 32 ED 0C D5 61
0F40 11 F0 0C 1A 13 B7 20 11 71
0F48 D1 10 E3 E1 3A EC 0C FE 2C
0F50 00 28 D2 2A EA 0C C3 66 A2
0F58 0D C5 4F 17 9F 47 E5 09 73
0F60 7E E1 C1 FE 00 28 04 FE B7
0F68 FF 20 D8 E5 21 ED 0C 34 A1
0F70 7E 00 2B BE 38 07 77 E1 7D
0F78 22 EA 0C 18 01 E1 18 C3 74
0F80 00 00 00 00 00 00 00 00 8F
0F88 CD 98 0F C3 00 0D 00 00 DB
0F90 CD 3E 00 FE 20 20 F9 C9 AA
0F98 CD 3E 00 FE 2F 20 F9 C9 C1
0FA0 73 7A C0 B9 75 8E BE A5 7B
0FA8 87 BB AC 78 98 BC 9B 91 9D
0FB0 76 A2 77 BD 89 AA 8C A7 71
0FB8 8B 96 A8 A9 9D A0 8A 93 93
0FC0 8D B4 AB B1 A6 7F 82 88 9B
0FC8 80 A1 92 9C B2 97 81 B3 A3
0FD0 BF BA 74 7D 79 84 B6 AF AB
0FD8 7E B0 B5 83 00 04 08 08 61
0FE0 04 08 10 08 04 00 00 00 17
```

```
> EXECUTE AT 0D00
> CHANGE 0F19 FROM C2 TO C3 FOR 2 PLAYER GAME.
>
    PRESS SPACE FOR NEXT MOVE OR TO CONTINUE GAME.
>   PRESS / FOR NEXT GAME.
>
```

> REVERSE By Howard Birkett

```
>T C60 F90
  0C60 C5 D5 E5 57 ED 5F 47 21 F6
  0C68 8B 0C 0E 03 3A 8D 0C E6 D5
  0C70 42 C6 3E 17 17 CB 16 23 F4
  0C78 0D 20 FA 10 EA 3A 8B 0C 76
  0C80 CB BF 92 30 FD 82 3C E1 74
  0C88 D1 C1 C9 DB 8A 1D 00 00 71
  0C90 06 0F CD 3C 02 10 FB 06 CD
  0C98 09 21 50 0C 7E CD 4D 02 C4
  0CA0 CD 3C 02 23 10 F6 CD 40 ED
  0CA8 02 CD 40 02 C9 00 00 00 8E
  0CB0 21 50 0C 06 0A 36 00 23 A2
  0CB8 10 FB 11 50 0C 0E 09 3E 91
  0CC0 09 CD 60 0C 21 50 0C 06 91
  0CC8 09 BE 23 28 F2 10 FA 12 F4
  0CD0 13 0D 20 EB CD 90 0C EF 5F
  0CD8 20 20 52 65 76 65 72 73 9B
  0CE0 65 20 3F 20 00 CD 3E 00 DB
  0CE8 FE 3A 30 F9 FE 32 38 F5 B2
  0CF0 32 95 0B D6 30 47 05 11 31
  0CF8 50 0C 21 50 0C 23 10 FD 0D
  0D00 47 CB 38 4E 1A 77 79 12 C1
  0D08 13 2B 10 F7 CD 40 02 CD 36
  0D10 90 0C 3A 59 0C C6 01 27 46
  0D18 32 59 0C 3E 01 06 09 21 2B
  0D20 50 0C BE C2 D7 0C 23 3C 4B
  0D28 10 F8 EF 20 20 59 6F 75 A9
  0D30 20 77 6F 6E 20 69 6E 20 C8
  0D38 00 3A 59 0C F5 E6 F0 28 D7
  0D40 06 F1 CD 44 02 18 04 F1 64
  0D48 CD 4D 02 EF 20 6D 6F 76 D2
  0D50 65 73 20 21 20 20 00 21 D7
  0D58 7B 0D E5 3A 59 0C FE 07 76
  0D60 DA A3 0D CA AF 0D FE 09 84
  0D68 DA BB 0D CA C2 0D FE 12 C0
  0D70 DA CC 0D FE 14 DA D6 0D FF
  0D78 C3 E6 0D EF 1F 1F 20 20 A8
  0D80 54 79 70 65 20 61 20 73 43
  0D88 70 61 63 65 20 74 6F 20 51
  0D90 70 6C 61 79 20 61 67 61 9C
  0D98 69 6E 1F 1F 00 CD 88 0F 1E
  0DA0 C3 F2 0D EF 50 75 72 65 FA
  0DA8 20 6C 75 63 6B 00 C9 EF 3C
  0DB0 56 65 72 79 20 67 6F 6F C8
  0DB8 64 00 C9 EF 47 6F 6F 64 6A
  0DC0 00 C9 EF 4E 6F 74 20 62 38
  0DC8 61 64 00 C9 EF 41 76 65 6E
  0DD0 72 61 67 65 00 C9 EF 4E 82
  0DD8 65 65 64 20 70 72 61 63 09
  0DE0 74 69 63 65 00 C9 EF 46 90
  0DE8 6F 72 67 65 74 20 69 74 13
  0DF0 00 C9 3E 1E CD 3B 01 21 4C
  0DF8 D4 0B 22 18 0C EF 09 20 42
  0E00 20 09 20 20 52 45 56 45 A9
  0E08 52 53 45 20 45 53 52 45 4F
  0E10 56 45 52 20 20 0D 20 20 98
  0E18 0D 00 CD 40 02 EF 54 6F F4
  0E20 20 77 69 6E 20 52 45 56 A9
  0E28 45 52 53 45 2C 20 61 72 84
  0E30 72 61 6E 67 65 20 74 68 47
  0E38 65 20 6C 69 73 74 20 28 CF
  0E40 31 09 39 29 20 69 6E 1F 00
  0E48 6F 72 64 65 72 2E 1F 54 13
  0E50 6F 20 6D 6F 76 65 2C 20 F0
  0E58 74 65 6C 6C 20 6D 65 20 29
  0E60 68 6F 77 20 6D 61 6E 79 91
  0E68 20 6E 75 6D 62 65 72 73 92
  0E70 20 74 6F 20 72 65 76 65 53
  0E78 72 73 65 2E 1F 46 6F 72 44
  0E80 20 65 78 61 6D 70 6C 65 9A
  0E88 2C 20 69 66 20 74 68 65 12
  0E90 20 6C 69 73 74 20 69 73 76
  0E98 3A 2D 1F 1F 20 20 32 20 DD
  0EA0 33 20 34 20 35 20 31 20 FB
  0EA8 36 20 37 20 38 20 39 20 14
  0EB0 1F 1F 61 6E 64 20 79 6F 37
  0EB8 75 20 72 65 76 65 72 73 F2
  0EC0 65 20 34 2C 20 74 68 65 14
  0EC8 20 72 65 73 75 6C 74 20 B5
  0ED0 77 69 6C 6C 20 62 65 3A B7
  0ED8 2D 1F 1F 20 20 35 20 34 1A
  0EE0 20 33 20 32 20 31 20 36 3A
  0EE8 20 37 20 38 20 39 1F 20 3D
  0EF0 1F 4E 6F 77 20 69 66 20 60
  0EF8 79 6F 75 20 72 65 76 65 35
  0F00 72 73 65 20 35 2C 20 79 76
  0F08 6F 75 20 77 69 6E 20 21 AA
  0F10 1F 1F 20 20 31 20 32 20 40
  0F18 33 20 34 20 35 20 36 20 79
  0F20 37 20 38 20 39 1F 20 54 AA
  0F28 79 70 65 20 61 20 73 70 09
  0F30 61 63 65 20 77 68 65 6E 3A
  0F38 20 79 6F 75 20 61 72 65 1C
  0F40 20 72 65 61 64 79 20 74 18
  0F48 6F 20 73 74 61 72 74 00 14
  0F50 CD 88 0F 06 0F CD 40 02 E7
  0F58 10 FB EF 20 20 48 65 72 C0
  0F60 65 20 77 65 20 67 6F 20 E6
  0F68 2E 20 2E 20 2E 20 2E 20 AF
  0F70 54 68 65 20 6C 69 73 74 7C
  0F78 20 69 73 1F 1F 00 C3 B0 34
  0F80 0C 00 00 00 00 00 00 00 9B
  0F88 CD 3E 00 FE 20 20 F9 C9 A2
```

EXECUTE AT 0DF2

```
   JJ  BY HOWARD BIRKETT
>?C50 0E0
 0C50 EF 20 61 6E 64 20 00 C9 87
 0C58 EF 20 77 65 6E 74 20 75 C6
 0C60 70 20 74 68 65 20 68 69 2E
 0C68 6C 6C 1F 5A 6F 20 66 65 19
 0C70 74 63 68 20 61 20 00 C9 25
 0C78 EF 20 6F 66 20 00 C9 EF 40
 0C80 20 66 65 6C 6C 20 64 6F 42
 0C88 77 6E 20 61 6E 64 20 62 4E
 0C90 72 6F 6B 65 20 68 69 73 B1
 0C98 20 00 C9 EF 20 63 61 6D CD
 0CA0 65 20 74 75 6D 62 6C 69 BE
 0CA8 6E 67 20 61 66 74 65 72 BB
 0CB0 2E 00 C9 EF 1E 47 69 76 E6
 0CB8 65 20 6D 65 20 61 20 72 33
 0CC0 6F 72 64 20 66 6F 72 20 98
 0CC8 74 68 65 20 66 6F 6C 6C E2
 0CD0 6F 77 69 6E 67 20 3A 1F 39
 0CD8 41 20 63 6F 6E 74 61 69 C3
 0CE0 6E 65 72 20 3F 1F 00 C9 78
 0CE8 EF 4D 61 6C 65 20 6E 61 51
 0CF0 6D 65 20 3F 1F 00 C9 EF 04
 0CF8 46 65 6D 61 6C 65 27 73 E8
 0D00 20 3F 20 1F 00 C9 EF 41 A4
 0D08 20 70 61 72 74 20 6F 66 E1
 0D10 20 74 68 65 20 62 6F 64 D3
 0D18 79 20 3F 1F 00 C9 EF 41 15
 0D20 20 6C 69 71 75 69 64 20 F5
 0D28 3F 1F 00 C9 DD 21 DB 0D 42
 0D30 21 E5 0D 06 05 CD B3 0C E7
 0D38 CD 9F 0D CD E8 0C CD 9F EB
 0D40 0D CD F7 0C CD 9F 0D CD 70
 0D48 06 0D CD 9F 0D CD 1E 0D D9
 0D50 CD 9F 0D 2A DD 0D CD D1 88
 0D58 0D CD 50 0C 2A DF 0D CD 7E
 0D60 D1 0D CD 58 0C 2A DB 0D 8E
 0D68 CD D1 0D CD 78 0C 2A E3 7E
 0D70 0D CD D1 0D EF 1F 00 2A 60
 0D78 DD 0D CD D1 0D CD 7F 0C 72
 0D80 2A E1 0D CD D1 0D EF 1F 5E
 0D88 00 2A DF 0D CD D1 0D CD 23
 0D90 9B 0C CD 3E 00 FE 2F 20 9C
 0D98 F9 C3 2C 0D EF 1F 00 CD 75
 0DA0 DB 01 11 4B 0B 1A FE 20 20
 0DA8 28 F5 DD 75 00 DD 23 DD 01
 0DB0 74 00 DD 23 F5 78 FE 04 A0
 0DB8 28 04 FE 03 20 03 F1 18 1E
 0DC0 03 F1 C6 20 77 23 13 1A 6E
 0DC8 FE 20 20 F6 36 00 23 05 67
 0DD0 C9 7E FE 00 C8 CD 3B 01 F3
 0DD8 23 18 F6 E5 0D E9 0D EE EC

>  EXECUTE AT 0D2C
>

   ENTER / FOR A NEW GAME.
>
```