nascom

NAS - SYS 3

ADVANCED 2K

MONITOR

The Nascom Microcomputers Division of Lucas Logic Limited reserves the right to amend/delete any specification in this brochure in accordance with future developments.

© Copyright Lucas Logic Limited

Nascom Microcomputers
Division of Lucas Logic Limited
Welton Road Wedgnock Industrial Estate
Warwick CV34 5PZ
Tel: 0926 497733 Telex: 312333

Lucas Logic

NAS-SYS 3 - IMPROVED VERSION OF NAS-SYS 1

1. All the keys on the keyboard automatically repeat when held down. (Not the @ key.) The initial repeat delay and the repeat speed are adjustable.

2. All routines are interruptable, so that interrupts can be used while executing NAS-SYS routines.

3. The CRT routine allows data to be output anywhere in memory, so headings can be output to the top line of the display.

4. The Read command has an optional parameter which allows cassette tapes to be read into any memory locations.

5. The Tabulate command is enhanced in three ways. Firstly, ASCII values of the bytes are output as well as the usual hexadecimal tabulation. Secondly, a fourth parameter can be used to specify the output of additional values on each line, to allow for printers of different widths. Thirdly, a fifth parameter allows suppression of either the hexadecimal output or the ASCII output.

6. All NAS-SYS routines can be single stepped, which makes it easier to test a program which uses NAS-SYS routines. By using the repeat keyboard feature, high speed single stepping is possible.

7. The register display is enhanced so that it shows the two byte value pointed to by each of the main registers.

8. Output from the Modify command is displaced two characters to the right, to improve readability.

9. The External (X) command has additional options, and no longer fails to output nulls. This enables the NULL command in BASIC to work correctly.

10. There are three new commands. P displays the stored user program registers, D executes a program at #D000, and Y executes a program at #B000.

11. The cursor blink rate is adjustable.

12. There are three new routines. Repeat keyboard scan, Output two spaces, and a new routine which can execute any other routine.

13. Both on breakpoint and on NMI, control passes through the $NMI jump before displaying the registers, allowing a program to take alternative action.

14. The B 0 command turns off the breakpoint completely, so that with appropriate hardware NAS-SYS can be executed in RAM.

15. Support in NAS-SYS itself for the use of paper tape has been removed, so there is no longer a Load command and the Tabulate command does not output a checksum.

NOTE: Because of improvements to the Read command, ARGN must be set to 0 before calling the Read routine from a program. Therefore when using the Nascom Tape Basic, you MUST enter the command POKE 3083,0 after each cold or warm start.

NASCOM OPERATING SYSTEM - NAS-SYS 3

NAS-SYS 3 is a 2K operating system for the Nascom 1 and Nascom 2 microcomputers. It makes it easy to enter, test and run machine code programs. It also provides a comprehensive set of routines which can be called by user programs. These routines are also used by the high level languages on the Nascom such as the 8K BASIC.

SUMMARY OF FEATURES

The contents of memory can be tabulated in hexadecimal and ASCII.

Memory locations can be modified with values entered in hexadecimal or ASCII.

Programs can be executed normally, or single stepped, or executed with a breakpoint set. During single stepping or at a breakpoint a full display of the machine registers is provided.

Commands are entered and edited using a blinking, non-destructive cursor. Comprehensive screen editing includes insertion and deletion of characters. This makes it easy to correct and reenter incorrect commands.

The keyboard routine allows every possible code to be entered when using the expanded keyboard, and provides automatic repeat when a key is held down. It also supports all features when using the original Nascom 1 keyboard.

Cassette tapes can be used to store programs, using a fast and fully checked method of recording the data. Tapes can be written which automatically load and execute machine code programs without any commands being entered at the keyboard.

All parallel I/O ports can be controlled and checked by direct commands.

Full support of serial terminals and printers is included, so that the computer can be controlled by a Teletype or equivalent device.

The computer can be used as a terminal, by connecting it with an acoustic coupler to a timesharing service, or even to another Nascom computer.

All internal codes are in ASCII, making it easy to attach other peripherals such as parallel printers.

(This document was produced on a Nascom computer, using the NASPEN word processing package, and printed on a Teletype 43 terminal.)

CONTENTS

LIST OF COMMANDS

A xxxx yyyy	Arithmetic - in hexadecimal
B xxxx	Breakpoint set or cleared
C xxxx yyyy zzzz	Copy - move data
D	Jump to address #D000 - run ZEAP
E xxxx	Execute a program
G xxxx yyyy zzzz	Generate a self loading cassette tape
H	Half duplex terminal
I xxxx yyyy zzzz	Intelligent copy - move data safely
J	Jump to address #FFFA - BASIC cold start
K xx	Keyboard option
M xxxx	Modify or examine contents of memory
N	Normal I/O to be resumed
O xx yy	Output data to port
P	Display program registers
Q xx	Query data from port
R xxxx	Read a cassette tape
S xxxx	Single step
T xxxx yyyy zzzz vv hhaa	Tabulate contents of memory
U	User specified I/O routines activated
V	Verify cassette tape is readable
W xxxx yyyy	Write a cassette tape
X xx	External serial device activated
Y	Jump to address #B000
Z	Jump to address #FFFD - BASIC warm start

(F and L commands do not exist)

Note. All hexadecimal numbers in this manual are preceded by "#".

HOW TO ENTER A COMMAND

Type the command letter in the first position on the line, followed by any values required. Each value must be separated from the others by one or more spaces. Then press the ENTER key. The line entered is ALWAYS the line where the cursor is blinking when the ENTER key is pressed. If the first position on the line is blank, no command is processed.

ERRORS WHEN ENTERING A COMMAND

If an invalid command is entered, the message "Error" is displayed on the next line. Very careful checking is performed by NAS-SYS, so that disastrous mistakes are less likely. The following types of error are detected:-

Invalid command character. (If the command character is blank, the line is simply ignored.)

More than ten values entered after the command character. (Only five values are ever needed, but a program can use up to ten values entered after the E command.)

Any value which is not a valid hexadecimal number. (Digits 0 to 9, and letters A to F.)

A value greater than #FFFF.

HOW TO CORRECT ERRORS

Simply press BACKSPACE to correct errors which you notice while typing the command.

However the comprehensive screen editing facilities allow you to go back and reenter an incorrect line provided it is still on the screen. Simply move the cursor back up the screen, and edit the line using the cursor movement keys, including deleting and inserting characters if required. Then press ENTER to enter the line.

For details of the screen editing facilities, read the next section.

SCREEN EDITING

The screen editing facilities are available with both the expanded keyboard, and the original Nascom 1 keyboard.

NAME	HEX CODE	KEYS PRESSED	FUNCTION
NULL	#00	CTRL/SHIFT/@	Ignored on display.
BS	#08	BACKSPACE	Move back and make position blank.
LF	#0A	LF or CTRL/J	Ignored on display.
FF	#0C	CS or SHIFT/BS	Clear screen and start at top left.
CR	#0D	ENTER or NEWLINE	Carriage return, line feed. Scroll up if at bottom.
CUL	#11	Left arrow or CTRL/Q	Move cursor left.
CUR	#12	Right arrow or CTRL/R	Move cursor right.
CUU	#13	Up arrow or CTRL/S	Move cursor up.
CUD	#14	Down arrow or CTRL/T	Move cursor down.
CSL	#15	SHIFT/Left arrow or CTRL/U	Delete character at cursor, move rest of line to the left.
CSR	#16	SHIFT/Right arrow or CTRL/V	Move rest of line to the right.
CH	#17	CH or CTRL/W	Move cursor to start of line.
CCR	#18	CTRL/X	If cursor at start of line, ignore. Otherwise do CR.
ESC	#1B	ESC or SHIFT/ENTER	Delete current line, and place cursor at start of line.

The cursor blinks and is non-destructive, which means that it can be moved over existing text without changing it. The cursor blink speed can be altered by changing the value KBLINK (#0C32-#0C33) in the NAS-SYS workspace.

THE KEYBOARD

The expanded keyboard has the ability to generate any of the 256 possible 8 bit codes, as follows:-

Hex #20 to #5F are the normal ASCII codes and are available as marked keys.

The letters are upper case, unless SHIFT is held down, when they become lower case. (Also see K command.) The @ key requires SHIFT to be held down, to produce an @.

Both the CTRL and the @ key operate as CTRL keys. This enables original Nascom 1 keyboards to make use of the screen editing facilities. When one of these keys is held down, and another key pressed, bit 6 is altered. This gives codes #00 to #1F and #60 to #7F.

The GRAPHICS key sets bit 7 on while it is held down, and this can be used to give graphic characters directly, with the Nascom 2 graphic option. When used in conjunction with the other keys, codes #80 to #FF may be obtained. (Also see K command.)

All keys (except the @ key) automatically repeat if held down. The delay before repeating starts and the repeat speed can be altered by changing the values KLONG (#0C2E-#0C2F) and KSHORT (#0C30-#0C31) in the NAS-SYS workspace.

THE RESET BUTTON

The Reset button is quite different to any other key. It sends a signal to the computer telling it to reinitialize itself. Pressing Reset does not result in the loss of whatever programs are stored in memory, but the operating system resets itself and takes control. The message "-- NAS-SYS 3 --" is displayed at the top left of the screen, and the computer waits for you to enter a command.

SCROLLING DISPLAY

When Reset is pressed or the screen is cleared, the cursor is positioned at the top left, and as commands are entered, the display scrolls down the screen. When the bottom line is reached, the whole display automatically starts to scroll upwards, showing the last 15 lines.

The top line of the display is never scrolled, making it a convenient place to display headings.

Note that user programs must not change the margins in the Video RAM.

DETAILED DESCRIPTION OF EACH COMMAND

A xxxx yyyy ARITHMETIC COMMAND

The Arithmetic command performs simple hexadecimal arithmetic. Three results are displayed, as follows:-

SSSS DDDD JJ

SSSS is the sum of the two values.

DDDD is the difference of the two values.

JJ is the displacement required in a Jump Relative instruction which starts at xxxx, to cause a jump to yyyy. If such a jump is not possible, then ?? is displayed instead.

B xxxx BREAKPOINT COMMAND

The Breakpoint command is used to insert a trap code into the program at address xxxx specified. Clearly a breakpoint cannot be inserted into a program which is in ROM (Read Only Memory). The command B 0 turns off the breakpoint.

Initially the breakpoint is turned off, since Reset deactivates it. The breakpoint command may be entered at any time. Once it has been entered, NAS-SYS remembers the breakpoint address and also keeps a record of the value at that address. Then, when an Execute command is entered, code #E7 is automatically inserted at the breakpoint. If the code is encountered during execution, then the program registers are saved, and are also displayed. The original value is replaced at the breakpoint address. Any command can then be entered, so that for example the program can be modified. If an Execute or Step command is then entered without specifying an address to start execution, the program will automatically restart where the breakpoint was. If an Execute command is entered, then the original instruction at the breakpoint is executed, and the breakpoint will only stop execution again the next time that it is encountered.

Note that the original value is put back into the program no matter which way the program is ended. (See "How to end a program".)

Note that the breakpoint must only be set at the first byte of an instruction in the program.

C xxxx yyyy zzzz — COPY COMMAND

Copy a block of data, length zzzz from xxxx to yyyy. One byte is copied at a time, starting with the first byte, so if there is an overlap between the source area and the target area, data may be destroyed. (Also see Intelligent copy command.)

The command can also be used to fill an area of storage with a single value. To do this put the value into memory at xxxx using the Modify command, make yyyy one greater than xxxx, and set zzzz to the number of bytes into which the value is to be copied.

D — D COMMAND

The D command starts execution of a program at address #D000. This is normally the cold start for the ROM version of ZEAP.

E xxxx — EXECUTE COMMAND

The Execute command executes a program, starting at address xxxx.

If address xxxx is not entered, then the stored program counter is used. This means that to continue execution after encountering a breakpoint, it is only necessary to enter E.

G xxxx yyyy zzzz — GENERATE COMMAND

The Generate command writes a cassette tape, which when read back in, loads a program and automatically executes it.

Data from address xxxx up to but not including address yyyy is written to the tape, and zzzz is the address at which execution is to start. Start the tape mechanism before entering the command. The LED is only on during output of the program.

When reading in the tape, there is no need to enter any commands at all. Simply start the tape, and stop it when the program has started execution.

The data on the tape is as follows:-
(CR) E0(CR) R(CR)
then the data in the same format as the Write command, then
Ezzzz (CR)

H HALF DUPLEX TERMINAL

The H command executes a very simple program in NAS-SYS which waits for input characters, and outputs them. The program can only be ended by pressing Reset.

To make the Nascom act as a half duplex, ASCII, computer terminal, enter the command X 30 and then enter the H command.

I xxxx yyyy zzzz INTELLIGENT COPY COMMAND

The Intelligent copy command is identical to the Copy command, in that it copies a block of data, length zzzz, from xxxx to yyyy.

However it is intelligent enough to start the copying process at whichever end is required to ensure that data in an overlapping region is never destroyed.

J JUMP COMMAND

The Jump command starts execution of a program at address #FFFA. This is normally the cold start for the 8K ROM BASIC.

K xx KEYBOARD COMMAND

The Keyboard command sets the method of operation of the keyboard.

K 0 is the normal option. This is the condition following Reset.

K 1 reverses the effect of the SHIFT key on the letters.

K 4 reverses the effect of the GRAPHICS key.

K 5 has the effect of both K 1 and K 4.

M xxxx — MODIFY COMMAND

The Modify command enables locations in memory to be examined and modified, starting at address xxxx.

The address of the location to be modified is displayed, followed by the current value. This value may be changed, and when the line is entered the new value is set at that location. Several values may be entered on the line, in which case subsequent locations also have their values changed.

Enter "." at the end of the line to end the Modify command.

Enter ":" at the end of the line to go back to the previous address instead of forward to the next.

Enter "/yyyy" at the end of the line to continue the Modify command at address yyyy.

Normally, hexadecimal values must be entered. However it is also possible to enter ASCII characters directly into memory by entering a comma followed by the character. For example ",A,B,C" would enter ABC into memory.

Any invalid values entered are detected, and an error message is displayed. In this case the command continues automatically at the address at the start of the last line.

N — NORMAL COMMAND

The Normal command resets the pointers to the input and output tables to their normal values. This has the effect of turning off an X or U command. All input will then be from the Nascom keyboard or the serial input, and output will be only to the screen.

O xx yy — OUTPUT COMMAND

The Output command sends data to a port. The value yy is sent to the port xx. For example "O 7 F" would send #0F to port #07. To learn how to use the parallel ports, read the PIO technical manual.

P = DISPLAY PROGRAM REGISTERS

The P command displays the stored user program registers. See "Display of Program Registers" for a description of this display.

Q xx QUERY COMMAND

The Query command obtains data from a port and displays it in hexadecimal. The port queried is the port xx. To learn how to use the parallel ports, read the PIO technical manual.

R xxxx READ COMMAND

The Read command reads a cassette tape which was written by the Write command. See the Write command for the format of the data. Normally the value xxxx is not entered, and the data is read back into the memory locations from which it was written. If xxxx is entered, then it is added to the address at which the data is stored. For example if the data on the tape was written from address #1000, and the command entered is "R 0800", then the data is stored at #1800.

As each block of data is read, the header information is displayed:-

SSSS BBLL .

SSSS is the start address of the data, BB is the block number, and LL is the length (00=256). After the last block, block 00, has been read correctly, the Read command automatically ends. During execution of the command, the LED is switched on.

The start of each block is recognised by reading the four start of block characters. All data is ignored until the start of a block is detected. If the checksum for the header data is incorrect, then a question mark is displayed, and the program waits for the start of the next block. The data following is not loaded. If the checksum for the actual data loaded is incorrect, then instead of a full stop, a question mark is displayed, and the program waits for the start of the next block. In this case, invalid data will have been loaded, but only at the correct locations. This technique eliminates the need for a buffer.

Do not press any keys on the keyboard while reading data since this will cause errors. (Which will be detected.)

A visual check of the display is required to ensure that all blocks have been loaded correctly. If any errors are encountered, simply rewind the tape for about two blocks and carry on.

To stop the Read command in the middle, press ESC (SHIFT/ENTER) four times. This only works between blocks, so if necessary first press any keys until end of block is reached.

S xxxx — SINGLE STEP COMMAND

The Single Step command executes a single instruction of a program, at address xxxx.

If address xxxx is not specified, then the stored program counter is used. This means that to execute a single step after encountering a breakpoint, it is only necessary to enter S.

If the previous command was a Step command, then even S need not be entered.

After the instruction has been executed, the program registers are saved, and are also displayed.

It is possible to single step through NAS-SYS routines called by a program, although since NAS-SYS should not contain any errors it is more efficient to set a breakpoint at the next instruction after the call and enter E to execute the routine.

High speed single stepping is possible, using the repeat keyboard feature. Enter the S command and then hold down the ENTER key. The repeat speed may be altered by changing the value KSHORT (#0C30-#0C31).

T xxxx yyyy zzzz vv hhaa — TABULATE COMMAND

The Tabulate command displays a block of memory, starting at address xxxx and continuing up to but not including address yyyy. Each line shows the start address followed by the values of several bytes starting at that address, in hexadecimal, followed by the same values in ASCII. Bytes with values in the ranges #00-#1F, #7F-#9F and #FF are output as ".".

zzzz lines are displayed at a time. Press any key to display the next group of lines, or press ESC (SHIFT/ENTER) to end the command. By setting zzzz to 0, all of the data will be output without a pause.

If vv is 0, then 8 bytes are output on each line. Otherwise 8+vv bytes are output. For example set vv to 8 for 16 values on each line. Fewer than 8 values can be output. For example set vv to #FC for 4 values on each line.

If aa is not 0, then the ASCII part of the display is suppressed. If hh is not 0, then the hexadecimal part of the display is suppressed.

Note that since the Tabulate command is the only command to have more that three parameters, it is not normally necessary to enter the fourth and fifth values, since they will remain unchanged, except on Reset when all values are set to 0.

Note that the repeat keyboard feature may be used to scroll up the tabulated data quickly. Set zzzz to 1 and hold down the ENTER key to scroll up.

U — USER I/O COMMAND

The User I/O command activates user specified input and output routines, in addition to the keyboard input and screen output, which continue to operate normally. These routines can for example control a parallel printer. The Normal command can be used to turn these routines off again.

The user output routine must be pointed to by an address stored at #0C78-#0C79, and the user input routine must have its address stored at #0C7B-#0C7C. On Reset these addresses point to a Return instruction within NAS-SYS, so if these addresses are not altered, entering the U command will have no effect.

The U command is automatically deactivated during execution of Read, Write and Generate commands, and reactivated afterwards.

V — VERIFY COMMAND

The Verify command is identical to the Read command, except that the data read from the cassette tape is not loaded into memory. The purpose of the command is to check that a tape can be read without error.

W xxxx yyyy — WRITE COMMAND

The Write command writes data to a cassette tape. Data from address xxxx up to but not including address yyyy is written to the tape.

Data is output in blocks, each of 256 bytes, except the last block, which may have less. The format of each block is as follows:-

00	Null (0).
FF FF FF FF	Four start of block characters (#FF).
SS SS	Start address, low order first.
LL	Length of data (00=256).
BB	Block number. This is one less for each block. The last block is block 00.
CC	Checksum for the header data.
DD DD	Data.
EE	Checksum for the data.
00 00 00 00 00 00 00 00 00 00	Ten nulls (0).

As each block is written, the header data, consisting of the start address, block number and length is displayed as follows:-

SSSS BBLL

When the command is entered, the LED is switched on, there is a brief delay, 256 nulls are output, and then each block is output. At the end, the LED is switched off and the command is ended.

The extra nulls at the end of each block ensure that even if several characters are lost in a block, the next block can still be read correctly. The extra null before the start of each block ensures that an initial spurious start of block character is ignored. The 256 nulls at the start ensure that error correction is always possible merely by rewinding the tape and playing it again.

X xx — EXTERNAL COMMAND

The External command activates input and output routines contained within NAS-SYS, which provide comprehensive capabilities for communication with external devices such as ASCII terminals (e.g. Teletype), and mainframe computers, through the serial input/output ports. The Normal command can be used to turn these routines off again.

The value entered after the X specifies the X option, each bit of which has a specific meaning described below. However the most commonly used options are as follows:-

22 Support a terminal in full duplex mode.
Line feed is automatically supplied after carriage return.
All output is even parity.

32 Same as 22, but line feed is not supplied.

20 Support a terminal in half duplex mode.
When a character is received from the terminal, unless it is ESC (#1B), or Null (0), it is assumed that the program will try to output the character. Therefore an indicator is set so that the next character output by the program is not sent to the serial output.
Line feed is always supplied after carriage return on input and output.

30 Same as 20, but line feed is not supplied.
This option makes the Nascom into a half duplex terminal.

23, 33, 21, 31 Same as 22, 32, 20, 30, respectively, but the output parity is odd instead of even.
(Input parity is always ignored.)

To input a BACKSPACE on a terminal which does not have this key, use CTRL/H.

The X command is automatically deactivated during execution of Read, Write and Generate commands, and reactivated afterwards.

The X command also activates the U command output routine, if one has been supplied. (See the U command.) This makes it very easy to have the Nascom keyboard and display, and a Teletype keyboard and printer, and a parallel printer, all working at the same time.

X OPTION BIT	FUNCTION
0	Parity of output characters. 0 = even, 1 = odd.
1	Suppress echo of input characters by user program, by setting bit 7 to 1 for each input character other than ESC (#1B) or NUL (0). 0 = suppress, 1 = do not suppress.

X OPTION BIT	FUNCTION
4	Supply LF (#0A) after CR (#0D) on output. Also output LF when CR is input unless bit 1 = 1, and regardless of the state of bit 5. 0 = output LF after CR, 1 = do not output LF after CR.
5	Make the input routine echo all input characters. 0 = echo, 1 = do not echo.
7	Indicator used to suppress the next output character. Set to 0 by output routine. See description of bit 1. 0 = do not suppress, 1 = suppress next output character.

Y Y COMMAND

The Y command starts execution of a program at address #B000.

Z Z COMMAND

The Z command starts execution of a program at address #FFFD. This is normally the warm start for the 8K ROM BASIC.

HOW TO ENTER AND TEST A PROGRAM

1. Write out the program on paper, in Z80 Assembler language. You will need to refer to the Z80 programming manual, and the sections of this manual describing the NAS-SYS facilities you can use for input and output, and how to end the program. There are also many useful routines in NAS-SYS which you may wish to make use of.

2. Then you can assemble the program by hand. This means that you must convert each instruction into machine code. The Z80 programming manual gives all the codes. The NAS-SYS Arithmetic command makes it easy to calculate the values to put into relative jump instructions.

3. If you have ZEAP, the Nascom Editor and Assembler package, then you can type in your program in Assembler, and ZEAP will generate all the machine code for you.

4. If you don't have an assembler, then use the Modify command to enter the program into memory.

5. Since there might be an error in the program, and this error might change any part of memory and quite possibly wipe out the program or corrupt it, it is wise to save the program on tape before starting to test it. Use the Write command to write the program to cassette tape. Unless you are confident that your tape recorder and tape are working reliably, use the Verify command to check that the tape can be read back.

6. To test the program, use the Execute command to run it. You can set a breakpoint first by using the Breakpoint command. You can also use the Single Step command to execute small parts of the program which are causing trouble. You can use the Modify or Tabulate commands to examine the program and areas of memory used by the program, after a breakpoint or during single stepping.

7. If you crash the program, you can always use the Reset button to regain control of the computer, and then use the Read command to reload the program for another attempt.

8. Although Assembler and machine code programming may seem slow and difficult at first, it is also fascinating, and it provides the only way to get the most out of the machine, in terms of both speed of execution of programs, and also in the depth of understanding which you will gain about the Z80 microprocessor and computing in general. Good luck!

DISPLAY OF PROGRAM REGISTERS

When the Single Step command is used, or when a breakpoint set by the Breakpoint command is encountered, or if code #E7 (RST #20) is executed in a program, the program registers are displayed as follows:-

```
-SP- nnvv   -PC- nnvv   -AF- nnvv   -HL- nnvv
-DE- nnvv   -BC- nnvv   I  -IX- -IY- Flags
```

The value nnvv after each of the six main registers is the two byte value at the memory locations pointed to by that register. vv is the actual byte pointed to, and nn is the next byte after that.

The flags are a decoded representation of the F register. The following characters may be displayed, indicating which flag bits have been set:-

S Z H P N C

The register display is often an essential aid when debugging a program. The Program Counter (PC) shows the address of the next instruction to be executed, and the Stack Pointer (SP) shows the position on the stack. The other registers show the effect of the program instructions on them. When the registers have been displayed, it is often necessary to investigate the actions of the program in more detail, by using the Modify command to determine the contents of various memory locations. For example you can find out what is on the stack.

SIMPLE INSTRUCTIONS FOR INPUT AND OUTPUT

NAS-SYS uses a most powerful method of controlling input and output. This is described in the section "Input and Output".

However, to simply output the contents of the A register to the screen, enter the code #F7 (RST #30) in your program. You simply type in F7 as part of your program.

To simply wait for an input character from the keyboard or from the serial input, enter the code #CF (RST 8) in your program. You simply type in CF as part of your program. The input character will be returned in the A register.

In both cases, no other registers will be affected.

Even if you use only this simple method provided by NAS-SYS for input and output, you can use the X or U commands to control terminals and printers. See the descriptions of these commands.

HOW TO END A PROGRAM

One of the following methods should be used to end each of your programs - and no other!

1. Press the Reset button at any time, to restart the system. A #76 (HALT) instruction may be placed in the program to make the Halt LED light, to indicate that the program has finished, and then you can press the Reset button to continue. This is a very primitive solution.

2. Execute code #C7 (RST 0) in the program. This is equivalent to pressing the Reset button. Like the first method this is simple but not very clever. Both these methods have the disadvantage that the screen is cleared so that you can't see what was output before the program ended.

3. The normal and recommended method of ending a program is to execute codes #DF #5B in the program. This provides a controlled return to NAS-SYS, so that you can enter commands. (The program registers are not saved, and the user stack pointer is set back to #1000.)

4. The method of ending a program at a place which it shouldn't ever get to is to execute code #E7 (RST #20) in the program. This stores the program registers and displays them, before returning control to NAS-SYS. This can be useful to indicate an abnormal end of program, as well as providing useful information for debugging. Execution of the program may then be continued by entering an Execute or Single Step command, but this command must specify the address at which execution is to start.

5. Generate a Non-Maskable Interrupt (NMI). If the computer has a hardware feature to allow the user to generate a single NMI by pushing a button, then this will have the same effect as method 4 above, except that execution can be continued by simply entering E.

Note that the action of NAS-SYS on encountering a breakpoint or an NMI is to jump to $NMI (#0C7D), which contains a jump to a NAS-SYS routine which stores and displays the program registers, and returns control to NAS-SYS so that commands can be entered. However it is possible to make a program change this instruction so that it causes a jump to a diagnostic routine of your own.

NAS-SYS RESTARTS AND ROUTINES

NAS-SYS provides many useful routines which can be called by user programs. These fall into three categories:-

1. Restart instructions. These functions are called by using the one byte Z80 RST instructions.

2. Routines which are called by special two byte instructions which in fact consist of a #DF (RST #18) instruction followed by a number which indicates the routine to be called. NAS-SYS routines should always be called by this method and never by a CALL instruction, because if NAS-SYS was changed, the routine numbers would be the same, but the absolute addresses would be different.

3. There is one routine which is called by a normal CALL instruction. This routine allows another program to perform NAS-SYS initialisation without losing control.

The routines in each of these three categories will now be described in detail, except for the low level input and output routines, which are described in the section "Input and Output".

Note that all NAS-SYS routines are interruptable, which means that at no time is data held on the stack at an address less than the stack pointer. Therefore programs which use interrupts can use NAS-SYS routines without any difficulty. Interrupt routines must of course preserve the state of all registers, and the stack space must be sufficient.

NAS-SYS RESTART INSTRUCTIONS

CODE	ASSEMBLER	NAME	FUNCTION
#C7	RST 0	START	Reset computer. Initialise NAS-SYS.
#CF	RST 8	RIN	Obtain an input character in the A register.
#D7	RST #10	RCAL	Relative Call. Follow this code with the displacement to the routine to be called. This is similar to the Z80 Jump Relative instruction, and it allows relocatable code to be written.
#DF	RST #18	SCAL	Subroutine Call. Follow this code with the number of the routine to be called. This is the method used to call the NAS-SYS routines. See the next section.
#E7	RST #20	BRKPT	Store and display the program registers, then return control to NAS-SYS. This is used by the Breakpoint command.
#EF	RST #28	PRS	Output the string of characters following this code, until a 0 is encountered. Then continue execution with the next instruction. This provides a very simple way of displaying a message. The A register is set to 0.
#F7	RST #30	ROUT	Output the character in the A register.
#FF	RST #38	RDEL	Wait for a period of time dependent on the value in the A register. A is set to 0.

NAS-SYS ROUTINES

These routines are called simply by putting the codes listed beside them into your program. For example routine ERRM outputs an error message. So to output an error message, enter the codes #DF #6B in your program. (You simply type in DF 6B as part of your program.)

It is also possible to call the routines which are the NAS-SYS commands. To do this use the code #DF followed by the ASCII code for the letter of the command. Registers HL, DE and BC should be set to the values stored in ARG1, ARG2 and ARG3 which would normally be typed in after the command. For example codes #DF #57 would call the NAS-SYS Write command routine. You would need to set HL and ARG1 to the start of the data to be written, and DE and ARG2 to the next address after the end of the data, before calling the routine. You may need to examine the listing of NAS-SYS to learn exactly how the commands work, and this is essential in the case of the Read, Verify and Tabulate commands.

You might want to define your own set of commands and routines, or add functions that are not in NAS-SYS. You can do this because the address of the start of the table of routine addresses is stored at $STAB (#0C71-#0C72). You can change this value to point to your own table of routine addresses. Examine the listing of NAS-SYS to learn how it works.

Before the list of the routines you can call, here is the description of a special routine called STMON which may be called at address #000D, by the codes #CD #0D #00. This routine reinitialises NAS-SYS and clears the screen. The point is that if the computer is set up so that on Reset it starts execution at an address other than 0, then the program which starts executing can set the Stack Pointer and then call STMON. It can then continue just as if it had been executed from NAS-SYS, and can use the NAS-SYS routines.

The listing of the NAS-SYS routines follows. You should find them easy to incorporate into your programs, and they should allow you to develop working programs more quickly.

CODES	NAME	FUNCTION
#DF #5B	MRET	This is not a normal routine, but is instead used to end a program and return control to NAS-SYS. See "How to end a program".
#DF #5C	SCALJ	This is not a normal routine, but is instead used to call any other routine, when the routine number is not known when the program is written. Store the routine number at address ARGC (#0C0A) and then execute codes #DF #5C. The routine will be called for you.
#DF #5D	TDEL	Wait for about 1 second (at 4 MHz). Registers A and B are set to 0.
#DF #5E	FFLP	Flip one or more bits of output port 0, then immediately flip them back again. Register A must have a 1 in each bit position to be flipped. A is modified.
#DF #5F	MFLP	Alter the state of (turn on or off) the tape drive LED. Register A is modified.
#DF #60	ARGS	Load the contents of ARG1 into HL, ARG2 into DE and ARG3 into BC. ARG1, 2 and 3 are the first three values entered after a NAS-SYS command. You could use this routine to pick up extra values typed in as part of the E command used to run the program.
#DF #62	IN	Scan for an input character. Instead of waiting for an input, like code #CF (RST 8), it just checks to see if there has been an input. If there has been, then the Carry flag is set and the character is in A. The A register is modified.
#DF #63	INLIN	Obtain an input line. This is the routine used by NAS-SYS to get commands. It provides a blinking cursor and waits for ENTER or NEWLINE to be pressed. The DE register is set to the address of the start of the line where the cursor was when when the line was entered. The A register is modified.

CODES	NAME	FUNCTION
#DF #64	NUM	Examine an input line and convert a hexadecimal value from ASCII to binary. Set DE to point to the start of the line. Leading blanks are ignored. The value is ended by a blank or null (0). DE is returned pointing to the next position. If the value is invalid (not 0-9, A-F, or >#FFFF), then the Carry flag is set, and DE points to the invalid character. The resulting value is placed in NUMV (#0C21-#0C22) and the number of characters in the ASCII value is placed in NUMN (#0C20). The HL and A registers are modified.
#DF #66	TBCD3	Output the value in the HL register in ASCII, followed by a space. Also add H and L into the C register. The A register is modified.
#DF #67	TBCD2	Output the value in the A register in ASCII. Also add A into the C register. The A register is modified.
#DF #68	B2HEX	Output the value in the A register in ASCII. The A register is modified.
#DF #69	SPACE	Output a space. The A register is set to a space.
#DF #6A	CRLF	Output a Carriage return/line feed. A is set to a CR.
#DF #6B	ERRM	Output the message "Error" followed by a CR. A is set to a CR.
#DF #6C	TX1	Output HL in ASCII, then a space, then DE, then another space. Also, H, L, D and E are added into the C register. Register A is modified.
#DF #6D	SOUT	Send a string of characters directly to the serial output port. HL must be set to point to the start of the string, and B must be set to the length. The C register is set to 0 and then all the characters are added into it. Registers HL, B and A are also modified.
#DF #6F	SRLX	Send the character in the A register directly to the serial output port.

CODES	NAME	FUNCTION
#DF #79	RLIN	Examine an input line and convert up to ten hexadecimal values separated by spaces from ASCII to binary. Set DE to point to the start of the line. The line must end with a null (0) character. RLIN uses the NUM routine to convert the values. If an invalid value is found or if there are more than ten values, the Carry flag is set. ARGN (#0C0B) is set to the number of values found. ARG1 (#0C0C-#0C0D) to ARG10 (#0C1E-#0C1F) are set to the values. NAS-SYS uses this routine to get the values from commands entered. The HL, DE, BC and A registers are modified.
#DF #7A	B1HEX	Output the low order (right) half of the A register in ASCII. The A register is modified.
#DF #7B	BLINK	Obtain an input character in the A register. While waiting for the input, blink the cursor on the screen. The display is not modified. Registers HL and DE are modified.
#DF #7C	CPOS	The HL register must be set to point to a position on the screen. Then the routine is called. It sets HL to point to the address of the first character on that line on the screen. Register A is modified.
#DF #7E	SP2	Output two spaces. The A register is set to a space.
#DF #7F	SCALI	This is not a normal routine, but is instead used to call any other routine, when the routine number is not known when the program is written. Store the routine number in register E and then execute codes #DF #7F. The routine will be called for you.

INPUT AND OUTPUT

This section describes in detail the powerful method which NAS-SYS uses to control input and output, so that you can make best use of it. However, it is not necessary to read or understand this section to make extensive use of NAS-SYS and its input and output facilities.

The descriptions of the N, U and X commands tell you how to control terminals and printers using NAS-SYS. The Restart instructions RIN, ROUT and PRS enable you to input characters and output single characters and messages. The NAS-SYS routines IN, BLINK and INLIN provide various input options, and TBCD3, TBCD2, B2HEX, B1HEX, SPACE, SP2, CRLF, ERRM and TX1 provide several simple ways to output data. (NUM and RLIN go a stage further and help you to get data from input lines.)

So at this point you should know how to specify input and output in your program, and how to control where this data is to go to by the N, U and X commands.

However, you can control where the data is to come from for input and go to for output, much more flexibly, if you understand the following description of how NAS-SYS works.

Both input and output work the same way, as follows:-

Every time an input or output is requested, a special routine (called ATE) in NAS-SYS calls each of a number of input/output handling routines in turn. Like all other NAS-SYS routines, these have routine numbers. There is a table of routine numbers for the input routines, and another for the output routines. Each of these tables is ended by a null (0). The address of the input table is stored at $IN (#0C75-#0C76), and the address of the output table is stored at $OUT (#0C73-#0C74).

As usual in NAS-SYS, the routine numbers are converted to actual addresses by referring to the table of subroutine addresses. The address of the start of this table is stored at $STAB (#0C71-#0C72).

The ATE routine which calls each of the input/output routines in turn, automatically preserves the HL, DE and BC registers. Output routines must preserve the AF register. Input routines must return with the Carry flag set and the input character in the A register if there is an input, otherwise the Carry flag must be reset.

The way that the NAS-SYS N, U and X commands work is to change the addresses of the input and output tables, at $IN and $OUT. NAS-SYS contains alternative input/output tables and other input and output routines for this purpose.

Now that you know how the input and output works, you will see that it is possible to set up your own tables of routine numbers, and then change the addresses at $IN (#0C75-#0C76) and $OUT (#0C73-#0C74) to point to your tables. You may also wish to add your own routines, and this is done most easily by using the jump instructions provided for the U command. NAS-SYS provides some routines to make it easier for you to change the addresses of the tables. These are as follows:-

CODES	NAME	FUNCTION
#DF #71	NOM	Set HL to the address of the new output table, then call this routine. It changes the address for you, and returns with the previous address in HL.
#DF #72	NIM	Set HL to the address of the new input table, then call this routine. It changes the address for you, and returns with the previous address in HL.
#DF #77	NNOM	Set the output table back to normal. It returns with the previous address in HL.
#DF #78	NNIM	Set the input table back to normal. It returns with the previous address in HL.

You now need to know the routine numbers to put in your tables. You must remember to put a 0 at the end of each table. The routine numbers are as follows:-

INPUT ROUTINES
==============

CODE	NAME	FUNCTION
#61	KBD	Scan Nascom keyboard.
#7D	RKBD	Scan Nascom keyboard and provide repeat key feature.
#70	SRLIN	Scan serial input port.
#74	XKBD	Scan external ASCII keyboard. (See X command.)
#76	UIN	User specified input routine. (See U command.)

OUTPUT ROUTINES
===============

CODE	NAME	FUNCTION
#65	CRT	Display on Nascom screen. See "Screen Editing" for details of facilities.
#6F	SRLX	Output to serial output port.
#6E	XOUT	Output to external ASCII device. (See X command.)
#75	UOUT	Output to user specified output routine. (See U command.)

NAS-SYS WORKSPACE

NAS-SYS requires an area of memory to use as a workspace. This occupies locations #0C00 to #0C7F, so the first free area for user programs is at #0C80. The workspace is initialised on Reset. User programs may choose to make use of or alter certain values in the workspace, so the following table is included to supplement the listing of NAS-SYS. The table shows the address and length of each value, in hexadecimal, and its name and function.

ADDR	LEN	NAME	FUNCTION
#0C00	1	PORT0	Copy of current state of output port 0.
#0C01	9	KMAP	Map of state of keyboard.
#0C0A	1	ARGC	Routine number processed by routine SCALJ.
#0C0B	1	ARGN	Number of values in input line.
#0C0C	2	ARG1	First value entered.
#0C0E	2	ARG2	Second value entered.
#0C10	2	ARG3	Third value entered.
#0C12	2	ARG4	Fourth value entered.
#0C14	2	ARG5	Fifth value entered.
#0C16	8	ARG69	Sixth to ninth values entered.
#0C1E	2	ARG10	Tenth value entered.
#0C20	1	NUMN	Number of characters in value examined by routine NUM.
#0C21	2	NUMV	Value returned by routine NUM.
#0C23	2	BRKADR	Breakpoint address.
#0C25	1	BRKVAL	Stored value from the breakpoint address.
#0C26	1	CONFLG	Normally 0, but set to -1 for the E command.
#0C27	1	$KOPT	Keyboard option. See K command.
#0C28	1	$XOPT	X option. See X command.
#0C29	2	CURSOR	Position of the cursor.
#0C2B	1	ARGX	Last command letter entered.
#0C2C	2	KCNT	Keyboard repeat counter.
#0C2E	2	KLONG	Keyboard initial repeat delay.
#0C30	2	KSHORT	Keyboard repeat speed.
#0C32	2	KBLINK	Cursor blink speed.
#0C34	#2B	MONSTK	NAS-SYS stack.
#0C61	2	RBC	Register save area. Register BC.
#0C63	2	RDE	Register save area. Register DE.
#0C65	2	RHL	Register save area. Register HL.
#0C67	2	RAF	Register save area. Register AF.
#0C69	2	RPC	Register save area. Program counter.
#0C6B	2	RSP	Register save area. Stack pointer.
#0C6D	2	$KTABL	Length of keyboard table.
#0C6F	2	$KTAB	Address of last byte in keyboard table.
#0C71	2	$STAB	Start of table of routine addresses, for routine 00. Since the first routine is in fact number #41, the actual table starts #82 beyond this address.
#0C73	2	$OUT	Start of table of output routines.
#0C75	2	$IN	Start of table of input routines.
#0C77	3	$UOUT	Jump to user specified output routine.
#0C7A	3	$UIN	Jump to user specified input routine.
#0C7D	3	$NMI	Jump to breakpoint/NMI routine. NAS-SYS sets this to display the registers.

ADDRESSING OF VIDEO RAM

The video RAM is addressed as shown in the diagram below. The top line is addressed after the other 15, and it is not scrolled by NAS-SYS. There is a 10 byte margin on the left of each line, then a 48 byte line, then a 6 byte margin on the right. When NAS-SYS clears the screen, all visible locations are made blank, and all the margins are set to nulls (0), except for the first 10 bytes, which are to the left of line 1, and the last 6 bytes, which are to the right of line 16, the top line.

LINE NO.	LEFT MARGIN	LEFT SIDE	RIGHT SIDE	RIGHT END
16	0BC0	0BCA 0BCB	0BF9	0BFF
1	0800	080A	0839	083F
2	0840	084A	0879	087F
3	0880	088A	08B9	08BF
4	08C0	08CA	08F9	08FF
5	0900	090A	0939	093F
6	0940	094A	0979	097F
7	0980	098A	09B9	09BF
8	09C0	09CA	09F9	09FF
9	0A00	0A0A	0A39	0A3F
10	0A40	0A4A	0A79	0A7F
11	0A80	0A8A	0AB9	0ABF
12	0AC0	0ACA	0AF9	0AFF
13	0B00	0B0A	0B39	0B3F
14	0B40	0B4A	0B79	0B7F
15	0B80	0B8A 0B8B	0BB9	0B8F
		LEFT SIDE	RIGHT SIDE	