

NASCOM OPERATING SYSTEM – NAS-SYS 1

NAS-SYS 1 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.

Facilities include:-

Modification and tabulation of memory.

Normal execution of programs, or single stepping through programs, or execution of a program with break-points set. During single stepping or at a breakpoint a full display of the machine registers is provided.

Comprehensive screen editing by using a blinking, non-destructive cursor. 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 support for all features with 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. Paper tape can be used to store programs in ASCII format.

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

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.

In February 2001 this document was scanned and OCR'd by OmniPage 5.1)

In August 2023, this document was converted to a pdf using LibreOffice.

LIST OF COMMANDS

A xxxx YYYY	Arithmetic - in hexadecimal
B xxxx	Breakpoint set or cleared
C xxxx yyyy zzzz	Copy - move data
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
L	Load a paper tape
M xxxx	Modify or examine data
N	Normal I/O to be resumed
O xx yy	Output data to port
Q xx	Query data from port
R	Read a cassette tape
S xxxx	Single step
T xxxx yyyy zzzz	Tabulate data or write a paper tape
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
Z	Jump to address FFFD - BASIC warm start

(D, F, P, Y commands do not exist)

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' ('New Line') 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 three 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 hexadecimal 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 completely.
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/O	Move cursor left.

CUR	17	Right arrow or CTRL/R	Move cursor right.
CUU	13	Up arrow or CTRL/S	Move cursor left.
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 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 use 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.)

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 1" 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.

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. A breakpoint cannot be inserted into a program which is in Read Only Memory, such as NAS-SYS itself. Therefore entering 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 E7H is automatically inserted at the breakpoint address. 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 and target areas, 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, make the value yyyy one greater than xxxx. Then put the value into memory at xxxx, and set zzzz to the number of bytes into which the value is to be copied.

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 MAS-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 executing a program at address FFFAH. This is normally the cold start for the 8K 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.

L

LOAD COMMAND

The Load command is used to load data stored on paper tape. The data must be in the format used by the Tabulate command for writing a paper tape.

The data loaded is carefully checked by NAS-SYS, and any lines containing errors are scrolled up on the screen, so that they can be reloaded or corrected. All correct lines are displayed for a moment, and then vanish from the screen.

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.

Invalid values entered are detected, and an error message displayed. In this case the command continues automatically at the address at 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 0FH to port 07H. To learn how to use the parallel ports, read the PIO technical manual.

Q xx QUERY COMMAND

The Query command obtains the value 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 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.

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 (0=256).

After the last block, block 00, has been read correctly, the Read command automatically ends. During 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. 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 keys on the keyboard 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.

Note that it is not possible to single step through certain parts of NAS-SYS itself, including the execution of the input and output routines. It is best to set a breakpoint immediately after any call to a NAS-SYS routine, and enter E to perform the routine. Single Stepping can then continue in the user program. This is more productive in case, since NAS-SYS should not contain any errors!

T xxxx vvvv zzzz TABULATE COMMAND

The Tabulate command displays a block of memory, starting at address xxxx and continuing up to but not including address vvvv. Each line shows the address followed by the values of the eight bytes starting at that location. zzzz lines are displayed at a time. Press any key to display the next group of lines, or press ESC or SHIFT/ENTER to end the command.

By setting zzzz to 0, all of the data will be output without a pause. By entering the command X 0 and then entering the Tabulate command with zzzz set to 0, a continuous block of data is transmitted to the serial output port, where it can be punched onto paper tape. A checksum is output at the end of each line. The Load command can be used to load this data back into the computer.

U

USER I/O COMMAND

The User I/O command activates user specified output routines in addition to the normal keyboard input and screen output. 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 at 0C79H, and the user input routine must have its address stored at 0C7BH. 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 Load, 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 (FFH).
SS SS	Start address. low order first.
LL	Length of data (0=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 are 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, as follows:-

- 0 Support a terminal in full duplex mode.
 Every character typed is sent back to the terminal.
 Line feed is automatically supplied after carriage return.
 All output is even parity.
- 10 Same as 0, but line feed is not supplied.
- 20 Same as 0, but half duplex.
 Characters entered are not sent back.
 However, line feed is always supplied after carriage return.
- 30 Same as 20, but line feed is not supplied.
 This option makes the Nascom into a half duplex terminal.
- 1, 11, 21, 31 Same as 0, 10, 20, 30, respectively, but the output
 parity is odd instead of even.
 (Input parity is always ignored.)

To store data on paper tape, use the X command (usually option 0), then the Tabulate command. Then use the Load command to load this data back into the computer.

When a character is received from the terminal, unless it is Escape (1BH), Null (0), or Del (7FH), 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.

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 Load, 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 Nascon keyboard and display, and a Teletype keyboard and printer, and a parallel printer, all working at the same time.

Z

Z COMMAND

The Z command starts executing a program at address FFFDH. This is normally the warm start for the 8K Basic.

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 HALT instruction (76H) 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 RST 0 (C7H) 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 code 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 1000H.)
4. The method of ending a program at a place which it shouldn't ever get to is to execute code RST 20H (E7H) in the program. This stores the program registers and displays them, before returning control to NAS-SYS. This can be useful to indicate in 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.

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 RST 18H (DFH) 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'.

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 10H	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 18H	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 20H	BRKPT	Store and display the program registers, then return control to NAS-SYS. This is used by the Breakpoint command.
EF	RST 28H	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
			message. The A register is set to 0.
F7	RST 30H	ROUT	Output the character in the A register.
FF	RST 38H	RDEL	Wait for a period of time dependent on the value in the A register. A is set to 0.

a

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, simply put the codes DF 63 into 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 (up to three) values 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 to the start of the data to be written, and DE 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.

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 address 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 which is called at address 000D, by the codes CD 0D 00. This routine, called STMON, reinitialises NAS-SYS and clears the screen. The point is that if the computer is designed so that an

Reset,

it starts execution at an address other than 0, then the program which starts can set the Stack Pointer and then call STMON. It can then continue just as 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 ARG0 (0C0A) and then execute code DF 5C. The routine will be called for you.
DF 5D	TBEL	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 'New Line' to be pressed. The DE register is set to the address of the start of the line where the cursor was when the line was entered. The A register is modified.

<u>CODES</u> =====	<u>NAME</u> =====	<u>FUNCTION</u> =====
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 >FFFFH), 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.

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. Registers E and A are modified.

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 ZEAP, 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 properly, 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 also 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 E7H (RST 20H) is executed in a program, the program registers are displayed as follows:-

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

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 what is on the stack, or what the memory locations pointed to by the HL register contain.

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 F7H (RST 30H) 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 CFH (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.

INPUT AND OUTPUT

This section describes 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 show you how to control terminals and printers using NAS-SYS. The NAS-SYS 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, CRLF, ERRM and TX1 provide several easy 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 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 address 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 and \$OUT 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	let 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. 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.
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 workspace. This occupies locations 0C00 to 0C7F, so the first free area for user programs is at 0C80. The first part of the workspace, from 0C00 to 0C6A is initialised to 0 on reset. The second part, from 0C6B to 0C7D is initialised from a table in NAS-SYS. 0C7E to 0C7F is also initialised by NAS-SYS. 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 part 0.
0C01	9	KMAP	Map of state of keyboard.
0C0A	1	ARGC	Command letter or routine number last processed for command execution or input/output.
0C0B	1	ARGH	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	0C	ARG49	Fourth 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	35	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.
-----	Initialised by table from here on -----		
0C6B	2	RSP	Register save area. Stack pointer.
0C6D	2	\$KTABL	Length of keyboard table.
0C6F	2	\$KTAB	Start of keyboard table.
0C71	2	\$STAB	Start of table of routine addresses, for routine 00. Since the first routine is in fact number 41H, the actual table starts 82H 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 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	0BBF
		=====		=====	
		LEFT SIDE		RIGHT SIDE	