CONTENTS
---------

IMPORTANT NOTES:-

1.  Never connect the keyboard connector to the serial interface - the keyboard will be damaged.

2.  Do not attempt to connect any external electrical equipment to your computer unless you have read this manual.

Rev 1.2     17 May 1981

# 1. INTRODUCTION
----------------------

Without a means to input instructions and data and a means to
output results a computer would be of very limited use. Your Nascom
computer provides several different methods of data input and
output. The most obvious of these are the video/televison output,
the keyboard input and the cassette drive interface, instructions
on the use of which are contained in the main body of your Nascom
manual. If you do not wish to use any other input and output
equipment you will not need to read the remainder of this manual.
However, as you become more ambitious and wish to use printers,
discs, direct input from switches, voltmeters etc, or outputs to
control motors, heaters, etc then you should find the information
contained in this manual useful.

The manual contains several separate sections, each dealing with an
aspect of input and or output operations with your Nascom computer.
The sections are:

1. A general overview of the way in which input/output (i/o) is
handled by the computer.

2. A more specialised discussion of the use of unterupts.

3. A description of the interface for serial input/output (i/o)
devices operating to V24, RS232 or 20 milliamp standards. Many
printers are of this type.

4. A description of the parallel interface used for connecting
on/off signals - up to 16 independent signals at 5 volt levels can
be handled by one such interface.

5. Instructions on how to connect a variety of different types of
printer to your Nascom.

6. How to program i/o operations in machine code or assembler with
the aid of the NAS-SYS operating system.

7. How to program input/output operations from within BASIC
programs.


# 2. OVERVIEW
----------------

Computers operate only on electrical signals that are either 'on'
or 'off'. For internal calculations and memory storage these
on/off, (or binary, two-state) signals are grouped in eights, and
the group of eight is known as a byte. The individual signals are
known as bits, and therefore one group of eight bits is known as
one byte. External inputs and outputs linked to the computer must

also consist of groups of on/off signals. Since these signals can only have discrete values they are known as digital signals. The switches of a computer keyboard, for example, can easily form input signals, and the specific codes reqired to operate printer mechanisms can also be output. Potential input signals which are of a continuously variable nature, such as a temperature input from a thermocouple, must be converted to a digital form before they can be input to a computer. Devices called analog to digital converters (ADC's) are used for this purpose, and might typically produce an eight bit (1 byte) code to represent the measured value.

In order to access these input or output signals the computer requires to be connected to electrical interface components, and to have a means of addressing these interfaces to read data from them, or output data to them. Most microprocessor chip families include special purpose integrated circuits to simplify the electrical interface. Your Nascom computer contains two such chips - a UART serial interface and a PIO parallel interface. These will be described in more detail in subsequent sections of the manual. The state of the inputs and outputs can be read/written in a very similar way to that in which data is accessed in memory locations. Indeed some microprocessors do not distinguish between memory and external i/o addressing and instructions. This type of i/o operation is known as 'memory mapped i/o', since the inputs and outputs behave just like memory locations. It has the advantage that it is sometimes easier to implement in small computer systems, and the full range of computer instructions can be applied to the inputs and outputs. However in larger computer configurations the conflict with areas reqired for memory is a major handicap. The Z80A used in your Nascom is not limited to memory mapped i/o, but features a full repertoir of i/o operations which can be applied to up to 256 i/o ports (1 port = 1 8-bit byte of input or output). All input and output on the Nascom board itself and on Nascom products uses this form of input/output, although for anyone intending to design their own special purpose interface circuitry for connection direct to the NASBUS there is nothing to prevent the use of memory mapped i/o.

Many devices linked to computers for input and output are either electromechanical in operation (eg printers) or are operated by human beings (eg keyboards). As a result they are slow in opration, and consequently the computer could spend much of its time waiting for such equipment to finish operating. In many microcomputer applications this is no great loss, since the computer has nothing else which it can do until this operation is complete. However in some instances this does represent a waste of time, for example where the computer has to perform a set of lengthy calculations, output a result on a printer and then continue with the next calculation. The subsequent calculation could be proceeding while the computer waits for individual letters to be printed. In another case a microcomputer might be required to monitor the state of many position switches in an automated manufacturing plant, and perform control functions based on the operation of these switches. In order to determine when any switch may be operated it would be necessary for the computer to loop round looking at each switch in turn, and when one is found to be operated to perform the necessary control function. Unfortunately while the control function is in

progress the inspection (or 'polling' as it is known) of the other inputs is suspended, and this might have disasterous consequences to the plant.

For many applications waiting for the completion of each i/o operation, or suitably frequent polling of i/o, is sufficient. However, in those instances where more efficient use of the computer's power is required, or where very rapid response to external events is required, it is necessary that such i/o requests be sensed by the computer and used to interrupt its current operation. The interrupting event should then be processed, after which the previous operation can be resumed. This is in fact a description of how the computer interrupt system works.

An interrupt appears to the microprocessor as an electrical signal. This signal is generated within the i/o interface circuit. The microprocessor responds to this signal by 'remembering' where it currently is (by storing its program counter on the stack) and then going to the address of a program, known as the 'interrupt handler'. The actual address of the program which is executed is determined by the current interrupt mode, and the form of interrupt generated by the interface. These parameters must have been correctly set up by the programmer when the main program was initiated. The types of interrupt available and the way in which they are programmed are described in more detail in section 3 of this manual for those who require such information.

Computer systems can work perfecly satisfactorily without using interrupts, and indeed use of interrupts has the following disadvantages:

1. Complexity of programming.

2. Difficult to de-bug.

3. Interaction of interrupts can be troublesome.

4. Interrupt handling routines can be large.

5. Interrupt handlers can be slower to execute, although execution begins almost immediately, than polling in simple cases.

Your Nascom computer has all the interrupt capability of the Z80A microprocessor available. However the operating system NAS-SYS does not itself use the interrupt system as this would increase complexity and none of the normal i/o devices warrant its use. The more advanced NAS-SYS 3 operating system will support the use of interrupts, although it does not itself utilise them for i/o to the screen, keyboard, cassette and serial ports.

We will revert now to a discussion of the arrangement of input and output interfaces present in your Nascom system. As mentioned previously the i/o consists of up to 256 ports handling 8 bits each. Ports 0 to 7 are associated with interfaces on the main Nascom board.

Port 0 is dedicated to system functions associated with the keyboard, single step and cassette drive. Only 2 of these signals are likely to be of interest - input bit 7, which appears on pin 3

on the main circuit board, and which will be discussed in connection with printer interfacing in section 5, and output bit 4 which appears on pin 10 of the main board, and which may be used with a suitable interface to switch the cassette motor on and off (see Nascom hardware manual).

## Port 0
------

| Output bits | Input bits |
|---|---|
| 7  Not available | 7  Not used (pin 3) |
| 6  Not available | 6  S6 |
| 5  Unused | 5  S5 |
| 4  Tape drive led (pin 10) | 4  S4 |
| 3  Single step logic | 3  S3 |
| 2  Unused | 2  S2 |
| 1  Reset keyboard counter | 1  S1 |
| 0  Clock keyboard counter | 0  S0 |

Ports 1 and 2 are associated with the serial i/o interface, which is also used for cassette tape i/o. Port 1 handles the actual data, while port 2 is a control port used to indicate the staus of the interface. The use of the serial interface is described in detail in section 4.

## Port 1
------

| Output bits | Input bits |
|---|---|
| 7 - 0  Data TO interface | 7 - 0  Data FROM interface |

## Port 2
------

| Output bits | Input bits |
|---|---|
| 7 | 7  Data received |
| 6 | 6  Transmit buffer empty |
| 5 | 5  Not assigned |
| 4 | 4  Not assigned |
| 3 | 3  Frame error |
| 2 | 2  Parity error |
| 1 | 1  Overrun error |
| 0 | 0  Not assigned |

Port 3 is not assigned in the system.

Ports 4 - 7 are associated with the parallel i/o interface (PIO), which contains 2 independent 8-bit ports for data and 2 corresponding control ports. The two ports in each PIO are referred to as

Port A and Port B of the PIO. The actual addresses used by the computer are ports 4 and 5 for data related to the on-board PIO of the Nascom 2. The functions of these ports are described in detail in section 5.


Port 4
--------

Output bits                          Input bits

7 - 0  Data out port A of PIO    7 - 0  Data in port A of PIO


Port 5
--------

As port 4, but for port B of PIO.


Port 6
--------

Output bits                          Input bits

7 - 0  Control port A            7 - 0  Status port A
(See section 5 for details)


Port 7
--------

As port 6, but for port B of interface.


All other ports, 08 - FF, are available for use on expansion input/output boards.

NOTE: If the Nascom board is strapped for INTERNAL i/o the ports 08 - FF are merely reflections of ports 0 - 7. Therefore when using extension i/o make sure that the i/o strap, LSW2/8, is in the upper position.


3.  INTERRUPTS
-----------------

The subject of interrupts was introduced in the previous section. For further details of the different types of interrupt and the way that they can be handled the reader is referred to the Z80 Technical Manual which forms part of the documentation provided with your Nascom computer.

# 4. SERIAL INPUT/OUTPUT - THE 6402 UART
-----------------------------------------------

A serial i/o interface is extremely useful since this is the most common method of communication used for printers, visual idsplay units (VDU's) and linking computers. There are also other instruments such as graph plotters and ADC's which can be used with a serial interface. In many instances only a 4-wire link is required between the instrument and computer - a transmit, a receive, a signal ground and a protective ground. Wiring is reduced in this way, but on the other hand the speed of transmission of data is

reduced since each bit is transmitted separately, by comparison with a parallel interface in which 8 bits are transmitted simultaneously. The maximum speed of transmission of serial data is rarely more than 19 200 bits per second (usually expressed as 19.2 K baud). However as most of the devices with which this communication is associated are relatively slow this is not a serious restriction. In fact printers and VDU's are usually run at 1200 baud (120 characters per second) or less.

## 4.1 Hardware
------------------

The serial interface on the Nascom microcomputer uses the 6402 integrated circuit (IC20). This is an extremely common serial interface chip, as is known as a UART, an abreviation for Universal Asynchronous Receiver/Transmitter. This makes for very easy programming of serial i/o, particularly with the aid of the routines already provided in the NAS-SYS operating system. While the i/o is made simple in this way it should be noted that the UART cannot be run in an interrupt mode, and therefore in applications involving slow serial i/o of long strings, and requiring a rapid response under interrupt to external stimuli it will be necessary to perform the serial i/o in short strings, or character by character, with interrupts disabled to avoid possible loss of data.

The UART on the Nascom computer is used to interface to the cassette data storage system. When the cassette is not in use the serial port can be used for interfacing to other serial devices such as printers. Additional UART's can be used with your Nascom by connecting Nascom Input/Output interface cards to the computer via NASBUS.

The standards used for serial transmission vary in data rate, separation of 8-bit characters and check bits (parity). For this reason your Nascom has been designed to facilitate the selection of the appropriate standard for your particular application. The options are selected by means of the jumpers/ switches at LSW 2 and LSW 1/5. The options are as follows:

## 1.Transmit Speed
----------------------

A speed of 300 baud (30 characters per second) or 1200 baud (120 characters per second) may be selected.

LSW2/1 UP = 1200 Baud               DOWN = 300 Baud

## 2. Transmit Clock Select (1)

Timing of the speed of transmission can be based on either the on-board clock setting of 1. above, or by the external setting of switch LSW2/3 to be described in 3. below.

LSW2/2 UP = External switch     DOWN = On board setting 1.

## 3. Transmit Clock Select (2)

This switch is ignored if LSW2/2 is down. Otherwise it selects a 110 Baud rate (for conventional 10 character per second teletypes) or an external clock.

LSW2/3 UP = External clock     DOWN = 110 Baud teletype

## 4. Receive Speed Select

This selects the receive speed of 300 Baud or 1200 Baud in a similar way to LSW2/1 described in 1. above.

LSW2/4 UP = 1200 Baud     DOWN = 300 Baud

## 5. Receive Clock Select (1)

This selects between the on board timing switch LSW2/4 and the external/teletype timing switch LSW2/6, in a similar way to LSW2/2 described in 2.

LSW2/5 UP = External/teletype     DOWN = On board setting

## 6. Receive Clock Select (2)

This selects between an external timing input and a timing rate of 110 Baud for teletypes, in a similar way to LSW2/3. If LSW*/5 is down this setting is ignored.

LSW2/6 UP = External     DOWN = 110 Baud teletype

## 7. Serial Input Select

The serial input/output UART is used for both cassette i/o and for the operation of other serial devices. When using cassette input an additional input circuit is used to condition the signal before input to the UART. When serial input is from other serial devices, such as the keyboard of a VDU, this circuit is not used. This switch is used to select between cassette input via the additional circuit and serial input form some other terminal.

LSW2/7 UP = Serial terminal     DOWN = Cassette

## 8. Stop Bits

In serial transmission systems the end of a character us denoted by
the presence of stop bits. Normally only 1 stop bit is required
except when oprating at 110 baud, when 2 stop bits are required.

LSW1/5 UP = 1 stop bit      DOWN = 2 stop bits

1 start bit is also present. These start and stop bits explain the
apparent discrepancy between a baud rate of 300 baud and the
corresponding pront speed of 30 characters/second. ie Each charact-
er is transmitted as 8 data bits + 1 start + 1 stop bit = 10 bits.
30 characters/second at 10 bits/character = 300 baud.

All the electrical connections from the UART are taken via signal
conversion circuitry to socket PL2 on the rear edge of the main
Nascom board. Cassette signals are also taken via additional
circuitry to pins 6, 7 and 9. The signals provided at PL2 are
suitable for the 2 electrical standards used for serial i/o,
namely:

1. RS232/V24, in which levels of +12 and  -12 volts are used to
indicate 0 or 1 in abit.

2. 20 milliamp current loop, in which zero current or 20 milliamps
is used to indicate 0 or 1.

The connections on PL2 are as follows:

PIN                          PIN

1 Tape drive control          9 20 mA loop in
2 +5 volts                   10 +12 volts
3 RS232 in                   11 Ground
4 Ext transmit clock         12 20 mA loop out
5 Ext receive clock          13 Cassette out hi
6 RS232 out                  14 Cassette out lo
7 -12 volts                  15 Ground
8 Spare                      16 Cassette in

Serial i/o connections  on external devices are commonly by means
of the 25 pin Canon D type connector. If you fit one of these
sockets the normal connections used are:

Pin 1 - Protective ground
Pin 2 - Signal transmitted by Nascom
Pin 3 - Signal received by Nascom
Pin 7 - Signal ground

Conventions on the use of other pins are unfortunately less
accepted.

## 4.2 Programming

The 6402 UART is extremely easy to program, and in fact all the

necessary routines required to use the UART on the main Nascom board are included in the NAS-SYS operating system manual, to which you are referred.

When using additional UART's on extension i/o boards the serial i/o routines from NAS-SYS may be copied, using the appropriate new port addresses. The procedure is basically as follows:

1. Output

Output the required character to the lower address of the UART.
Read the higher address and continue to do so until bit 6 goes high, indicating that the character has been acceptedt.
Continue in this way until the character string to be output is exhausted.

2. Input

Read the higher address to determine status.

If bit 7 is zero, no character read, so continue.

If bit 7 was 1 then read the character from the lower address.

Additional Notes
-----------------------

1. IT IS VITAL that the keyboard should NOT be connected to PL2 - damage to the keyboard will result.

2. Before attempting to connect external serial devices make sure whether they are 20 mA or RS232, and connect accordingly. Also check the required transmission rate and number of stop bits.

3. Remember if you alter the settings of receive/transmit switches that these may affect tape operation. If so, reset them before using tape again.

4.. Serial RS232 comunication is used in linking computers and terminals over long distances using modems and the Post Office telephone network. However, the PO has very strict regulations governing equipment which can be connected via modems to its telephone network for safety reasons. Surprisingly this even applies to the use of acoustically coupled modems. You must not connect equipment in this way unless you have obtained Post Office approval for your system.

# 5. PARALLEL I/O - THE Z80 PIO (3881)
--------------------------------------------------------------

Your Nascom computer is fitted with one PIO (programmable input
output) chip, IC19, which is not utilised at all in the standard
configuration. This will allow for the connection of up to 16
individual on/off electronic signals. These signals must be at  TTL
(transistor-transistor) voltage levels - ie +5 volts and 0 volts.
For further details of electrical interfacing see section 9.

The PIO is a highly sophisticated circuit which allows many
different modes of operation. These modes of operation are achieved
by programming of the PIO itself. This programming is carried out
by outputting commands to the PIO. The flexibility permitted by the
PIO is unfortunately offset by the added complexity involved in
programming it before it can be used. However, if you read
carefully through the explanation which follows you should not
experience any problems. the PIO consists of 2 separate 8-bit i/o
posrts, each handling 8 parallel i/o lines. These will be referred
to as port A nd port B - on the main board of the Nascom these are
addreesed as ports 4 and 5 respectively. The 2 ports behave almost
identically. Associated with each of these data i/o ports is a
control port. Each port also has associated with it 2 additional
bits which can be used to 'handshake' data into and out of the
port. These are known as the 'strobe' (STB) and 'ready' (RDY)
lines.

On the main Nascom board we therefore have:

Port 4    Input/output data port A
Port 5    Input/output data port B
Port 6    Control for port A
Port 7    Control for port B

Because the PIO must be programmed to specify its mode of operation
it is essential that any program which uses the PIO should contain
the instructions to issue the control commands to the PIO. These
instructions must be in the user program - they are not carried out
by the operating system or BASIC automat- ically. Thus in BASIC the
control commands must have been issued at some time before IN and
OUT instructions are used on the data ports. The commands can of
course be issued from BASIC using the OUT instruction to the
appropriate control port.

## 5.1 Control of the PIO
--------------------------------

### 5.1.1 Input/output mode
--------------------------------

The first step in controlling the PIO is to select which lines
(bits) of the port are to be used for input and which for output.
To do this a byte of data is output to the control port (A or B,
corresponding to the data port we are using) of the following form:

| M1 | M2 | 0 | 0 | 1 | 1 | 1 | 1 |
|----|----|---|---|---|---|---|---|

1. Mask control

| X | Y | Z | F | 0 | 1 | 1 | 1 |
|---|---|---|---|---|---|---|---|

The mask control register determines whether interrupt is enabled
on the PIO concerned, and in mode 3 also allows particular bit
patterns to cause an interrupt to be generated. This is described
in more detail in section 4.3.

2. Interrupt enable

| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
|---|---|---|---|---|---|---|---|

This causes interrupts generated from the assicaited data port to
be enabled.

3. Interrupt vector

| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
|---|---|---|---|---|---|---|---|

The next byte sent to the control register will contain the
interrupt vector used to determine the address of the interrupt
handling program in memory. This is only applicable when the
computer has been programmed to work in interrupt mode 2 (see
section 3). The setting of this vector is described in more detail
in section 4.3.


5.2 Data Input and Output
------------------------------------

Data input and output via the PIO is achieved simply by means of
input and output instructions addressing the appropriate data port.
Programming can be carried out in either assembly language /machine
code or BASIC. Remember, however, that prior to data i/o the port
must have been programmed using the control instructions described
in section 4.2 - at least to specify whether ports and data lines
are to be used for input or output.

5.3 Interrupts
------------------

The form that an interrupt from a PIO will take depends on 4
different control instructions sent to it. The response of the
computer will depend on the interrupt mode selected within the
program and the interrupt handling routines which the program
provides. These modes of interrupt handling within the computer
itself are described in section 3 of this manual.

5.3.1 Interrupt enable
------------------------------

Interrupts will only be generated by the PIO if the interrupt
system of the PIO is enabled. This is done be sending a byte
containing 03 to the control port of the PIO port. Note that this
is in addition to the requirement that the computer's interrupt
system should be enabled.

The values of M1 and M2 have the following effect:

```
0 , 0      All the bits are used for output (mode 0)
0 , 1      All the bits are used for input (mode 1)
1 , 0      All the bits are bidirectional (mode 2)
1 , 1      All the bits are to be individually specified (mode 3)
```

In mode 3 a further byte must be output immediately to the control port to specify which bits are input and which are output. This byte will contain a 1 in each bit which is to be an input, and a 0 in each bit which is to be an output. In this mode the STB and RDY lines are not used.

In mode 0 (output) data appears on the outputs as soon as it is sent by the computer. The RDY line is simultaneously turned on to indicate to any external device connected to the port that data is ready. The external device could then acknowledge one it has completed action on this data by turning on the STB line. If the interrupt facility is use (see section 4.3 below) then this strobe can be used to cause an interrupt, which an interrupt handler can use as an indication that the next block of data should be output.

In mode 1 data is only read in to a port when the STB line is turned on. It is edge, rather than level, triggered. This strobe can also be used to cause an interrupt (if interrupt is enabled) so that the computer can be programmed to react to the availability of data. When the computer reads the data the RDY line is turned on to indicate to the external device that the data has been accepted. If you wish to define the port as being for input, but without a STB to cause data to be latched onto the port then the port should be defined as mode 0, all inputs, rather than as mode 1.

The bidirectional mode, mode 2, requires the use of all 4 handshake lines associated with the one port. It can only be used with port A, and port B must be in mode 3, so that it does not attempt to use handshake lines. In this mode the port A RDY and STB signals are used for output control and the port B handshakes are use for input control. It is effectively a combination of modes 0 and 1, except that data output from port A is only enabled when the A STB line is turned on (in order to prevent conflict with data input).

Note that a pulse on the STB line sent to a port is not remembered by the PIO. If the STB is to be used to notify the computer that input data is ready, or output has been completed, the PIO should be run under interrupt (see section 4.3).

5.1.2 Other control instructions
-----------------------------------------------

Three other optional forms of control byte can be sent to the control port of a PIO. These are all concerned with operation under interrupt, and are as follows:

## 5.3.2 Interrupt control word
-------------------------------

This indicates the logic to be used in interrupt generation. It is specified by a byte sent to the corresponding control port of the form

         X  Y  Z  F  O  1  1  1

If bit X is zero, interrupts from this PIO port are disabled. If bit X is set then interrupts are enabled. In mode 3, control mode, the bits Y, Z and F have the meaning shown below (in other modes they are not used):

Y = 1 means ALL monitored lines must go active to generate an interrupt.

Y = 0 means ANY monitored line going active generates an interrupt.

Z = 1 defines the active state as being high.

Z = 0 defines the active state as being low.

F = 1 means that a further byte will be transmitted to the cotrol port specifying those lines which are to be monitored - ie a mask word is to follow. Only lines for which the mask bits are 0 (ZERO) will be monitored.

F = 0 means no mask is specified (all monitored).

## 5.3.3 Interrupt mask word
-------------------------------

This determines which data lines on the port will be moitored to generate interrupts, as described in 5.3.2. Only the data bits corresponding to mask bits set to 0 (ZERO) are monitored.


## 5.3.4 Interrupt vector
-------------------------------

When the computer is running in interrupt mode 2, as described in section 3, the vector indicating the low address byte of the interrupt handler must be specified to the PIO. This is done by sending the control byte

         V7  V6  V5  V4  V3  V2  V1   0

to the control port, where V7 - V1 represent the required vector.

## 5.4 Priority of interrupts
-------------------------------

The Z80 microprocessor and its associated PIO's use a daisy chain interrupt system to allocate priority to different interrupts. This gives, on the Nascom, the on-board PIO the highest priority in the chain. The priority of other devices will depend entirely on their

distance (in logical terms of the wiring of the interrupts on the computer's backplane) from the processor. This is discussed in more detail in section 3.


IMPORTANT NOTE:-

When the power is turned on to the computer and interfaces they all perform a power on reset, which clears internal registers and leaves intrrupts turned off. However, a reset initiated by the front panel reset switch does NOT reset the PIO's - this is a design feature of the PIO. This means that after program crashes, particularly when the interrupt system is in use, the PIO may be left in a state where it is awaiting acknowledgement of an interrupt, thus disabling further interrupts. This must be cleared to resume normal operation. This can be achieved in several ways -

1. Power the complete system off and on again.

2. Execute an RTI instruction, follwed by a return to NAS-SYS in order to clear the interrupt in the normal way by software. This is done by executing
LD HL,0000 ;21 00 00
PUSH HL    ;E5
RETI       ;ED 4D
 3. Outputting the value 83 to the control ports of the PIO.

These precautions are necessary only when using interrupts, and when programs have failed leaving interrupts unacknowledged.