NCT
NORD COLOUR TERMINAL

User's Guide

NCT
NORD COLOUR TERMINAL
User's Guide

REVISION RECORD

Revision	Notes
10/77	Original Printing
02/78	Second Edition – supersedes Original Printing

NCT – NORD Colour Terminal User's Guide
Publication No. ND.60.094.02

NORSK DATA A.S.

Lørenveien 57, Postboks 163 Økern, Oslo 5, Norway

TABLE OF CONTENTS

+ + +

1 THE NORD COLOUR TERMINAL SYSTEM

To run the NCT system you need the following equipment:

- one TTY or asynchronous modem interface
- one red-green-blue monitor
- one NCT controller
- .one cable to connect the interface to the controller
- coax cables to connect the RGB monitor to the controller

In addition, as options, there is a special NCT keyboard connected directly to the controller and a joystick, also connected directly to the controller.

The NCT controller contains: a 3K-16 bits refresh memory, a 2K-8 bits symbol buffer and a 32 cell-12 bits colour buffer.

All buffers/memories can be written in from the computer, but cannot be read.

The possibilities are as follows:

- 512 x 384 programmable points on the monitor

- the 512 horizontal points are divided into 64 addressable columns and the 384 vertical points are divided into 48 addressable lines.

- each of the 3072 grids (64 x 48) may be displayed with:

    - one of 16 programmable foreground colours
    - one of 8 programmable background colours
    - blinking or not blinking (foreground and background blinks with different frequencies)
    - one of 16 height sizes on the grid
    - one of 256 programmable symbols within the grid
    - a cursor follow and a cursor hold possibility
    - one of the 8 programmable cursor colours
    - a window up/down possibility for looking at parts of the refresh memory not at present on the monitor screen

- a keyboard may be delivered (as an option) with standard alphanumeric keys, a set of control keys and function keys with the text on keytops with built in light bulbs. These lamps are programmable to give light/no light.

- a joystick option for cursor control.

1.1 REFRESH MEMORY

The refresh buffer holds, as mentioned, 48 lines of information.

When an 8 x 6 symbol size is specified, all the 48 lines are shown on the monitor screen at the same time. If, however, a greater symbol size is specified, there will be more lines in the refresh buffer than are visible on the monitor screen. The windowing mechanism may be used in this case for showing the other part of the refresh buffer.

The content of a cell in the refresh buffer contains pointers to the foreground and background colours in the colour buffer, a pointer to one of four symbol tables and a pointer to one of the 64 symbol tables inside each symbol table.

And, finally, blink/no blink flag.

Figure 1.1 illustrates the layout of the refresh memory with its pointers.

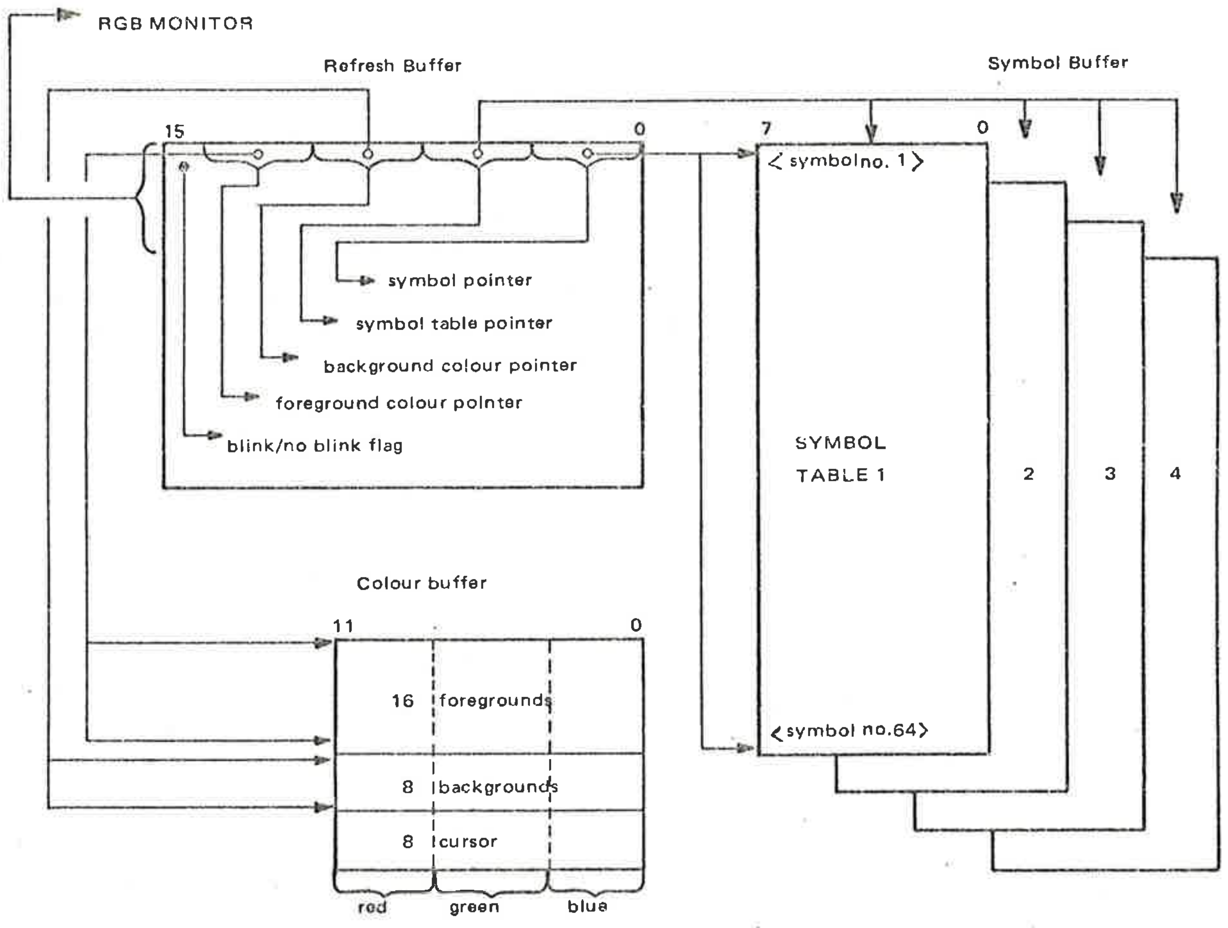


Figure 1.1.

1.2 SYMBOL BUFFER

The symbol buffer is divided into four blocks (tables). Within each block it is possible to write 64 symbols. The symbol generator utilizes a writable storage to specify each symbol. This store contains one bit for each point in the symbol. The symbol size may be from an 8 x 1 grid and up to an 8 x 16 grid.

Using the 8 x 8 format or less, the symbol store can contain specifications of up to 256 different symbols.

For formats greater than 8 x 8, the symbol stores can contain up to 128 different symbols since two of the symbol blocks must be connected to obtain such grids.

```
.  .  .  .  .  .  .  .
.  .  x  x  x  x  .  .
.  x  .  .  .  .  x  .
.  x  .  .  .  .  x  .
.  x  x  x  x  x  x  .
.  x  .  .  .  .  x  .
.  x  .  .  .  .  x  .
.  .  .  .  .  .  .  .
```

Figure 1.2.

This figure shows the layout of an alphanumeric character, which has been written into a character position in one of the symbol tables in an 8 x 8 grid (. means 0, x means 1).

Initially, the ASCII set is loaded into symbol buffer number one using an 8 x 8 grid.

1.3 *COLOUR BUFFER*

The colour buffer contains:

- 16 cells for the foreground colours present at any one time
- 8 cells for the background colours present at any one time
- 8 cells for the cursor colours present at any one time

Each colour level contains information about how much red, green and blue colour must be mixed to give the actual colour on the specific level.

Each colour on one level has 16 mixing possibilities, and since each level has three main colour possibilities, this will give one of the foreground, background or cursor levels the ability to select one colour out of 4096 colour possibilities.

Initially, the colour buffer is loaded with a standard set of colours, so the user may start running the software using these.

Example:

The foreground colour number one has the following binary value in the colour buffer:

1 1 1 1	0 1 0 0	0 0 0 0
red	green	blue

This content gives 15 red parts (maximum), 4 green parts and no blue parts, which together gives a nice red colour.

The initial colours are: Standard farger

Foreground Number:

		R	G	B
0	black	0	0	0
1	red	15	4	0
2	green	0	14	0
3	yellow	14	14	0
4	blue	0	10	14
5	mangenta	15	5	14
6	cyan	0	14	13
7	orange	13	9	2
8	olive green	9	11	0
9	crimson lake	14	6	6
10	prussian blue	4	6	10
11	dark red	12	3	0
12	turquoise blue	0	10	9
13	brown	11	6	2
14	imperial blue	10	14	9
15	white	14	14	14

Background Number:

0	black	0	0	0
1	red	2	0	0
2	green	0	2	0
3	yellow	2	2	0
4	blue	0	[illegible]	2
5	mangenta	7	1	7
6	cyan	0	2	2
7	grey			

Cursor Number:

0	grey
1	cyan
2	mangenta
3	blue
4	yellow
5	green
6	red
7	black

The cursor colours are chosen to give a colour different from the background colour on the same level. The reason for this is that the cursor levels automatically follows the background levels when cursor is used.

This means that when cursor is shown on background level 0 (black), the cursor level used is level 0 (grey), and whenever the background colour changes the cursor colour changes automatically. Due to the black level setting of the monitor, the background number 0 always has to be defined as black.

2 NCT - SERVICE PROGRAM

The NCT Service Program is divided into two parts:

- one subsystem which makes it easy to update, generate and save different, or parts of pictures, symbols and colours
- one BRF library containing most of the subsystem facilities to be called and used by other subsystems.

The first part of this chapter describes the system commands and how to use them. The second part contains the parameters to be used when calling the same functions from other subsystems.

A reasonable way to use the NCT-SP is to generate the wanted symbols, colours, macros and main pictures by help of the subsystem, and then update, change and look at the results by calling the BRF library from programs written in BASIC, FORTRAN or NORD PL.

The commands or the possibilities of the NCT-SP may be divided into the following groups:

- The clear/initiate commands
- The colour commands
- The symbol commands
- The picture commands
- The macro commands
- The keyboard commands
- All other commands (auxilary)

2.1 *THE CLEAR/INITIATE COMMANDS*

- NCT-NAME

- CLEAR

2.1.1 *NCT-NAME <device name>*

The first thing to do when entering the NCT-SP is to specify the output device name using this command.

The command is necessary even when one is using an NCT keyboard as it initializes the refresh image and loads a standard set of colours and symbols.

Example:

```
*NCT-NAME     NCT
```

In this example the actual terminal is named NCT in SINTRAN.

Example:

```
*NCT-NAME ↵
```

Here the command is given from an NCT keyboard and therefore no name has to be specified, but the command is nevertheless necessary.

2.1.2 *CLEAR*

This command clears the screen, the internal refresh buffer, and an internal dynamic buffer (described later in connection with the picture commands).

2.2 *THE COLOUR COMMANDS*

The following colour commands are available:

- LOOK-AT-COLOUR
- WRITE-COLOUR
- CHECK-COLOUR
- SAVE-COLOUR
- FOREGROUND
- BACKGROUND

2.2.1 *LOOK-AT-COLOUR <FG/BG> <No.>*

This command gives the opportunity to look at the colours currently in the colour buffer. The colours will be shown from the present position on the monitor.

The command has two parameters, where the first determines whether foreground (FG) or background (BG) or both (default (CR)) shall be displayed.

The second parameter determines whether only one <no.> or all (default (CR)) colour levels shall be displayed.

Example:

```
*LOO-AT-COL ↵
(FG/BG): F ↵
NO.: ↵
```

This example will display all 15 foreground colours present from the current position on the monitor.

2.2.2 *WRITE-COLOUR <FG/BG/CU> <No.> <Input File>*

WRITE-COLOUR is used to change one or more of the present colours.

The first parameter determines whether it is the foreground, background or cursor colour which is to be changed. The second parameter determines the level number inside the described group.

The third parameter determines which file the new colour information is to be taken from (default file is the terminal (CR)).

The colour information is three decimal numbers ranging from 0 to 15 separated by spaces, which give information of how many red, green and blue parts the actual new colour level is to consist of.

Example:

```
*WRITE-COL
(FG/BG/CU): BG
NO.: 1
INPUT FILE
10  3  0 ↵
```

This command will change the background colour number 1 into a colour consisting of 10 red parts, 3 green parts and no blue parts.

NOTE: Do not try to change background colour number 0 (black). It will cause problems for the monitor and the NCT-SP.

2.2.3 *CHECK-COLOUR <FG/BG/CU> <No.>*

This command will print on the terminal, for the colour specified by the two parameters used in the same way as the commands above, 3 decimal numbers which show the relation of red, green and blue parts giving the actual colour.
Example:

```
*CH-COL
(FG/BG/CU): FG
NO.: 2
0 14 0
```

The present foreground colour for level 2 consists of no red parts, 14 green parts and no blue parts.

2.2.4 *SAVE-COLOURS <Output File>*

The present colour buffer may be saved on a file to be used as needed in connection with the WRITE-COLOUR command.

2.2.5 *FOREGROUND <No.>*

The parameter <no.> means a foreground level between 0 and 15. All further output on the monitor will be made with the foreground colour specified.

Example:

*FOREGR 1

2.2.6 *BACKGROUND <No.>*

The parameter <no.> means a background level between 0 and 7. All further output on the monitor will be made with the background colour specified.

Example:

*BACK 7

2.3 *THE SYMBOL COMMANDS*

The following symbol commands are available:

- LOOK-AT SYMBOL
- WRITE-SYMBOL
- CHECK-SYMBOL
- SAVE-SYMBOL
- SET-SYMBOL-BUFFER

2.3.1 *LOOK-AT-SYMBOL <SYMBOL-BUFFER (1/2/3/4)> <No.>*

It is possible to look at the content of the different symbol buffers by using this command.

The content is displayed from the current position on the screen with the current foreground and background colours.

The first parameter determines which one of the four symbol buffers is to be shown (default is all 64 (CR)).

Example:

```
*LOOK-AT-SYM
SYMBOL BUFFER (1/2/3/4): 1
NO.: ↵
```

This example will display all the 64 characters present in symbol buffer number one.

2.3.2 *WRITE-SYMBOL<SYMBOL BUFFER (1/2/3/4)><No><Input File>*

This command is used to change one or more of the characters in one or more of the symbol buffers.

The first parameter determines which of the symbol buffers is to be used (default is all four (CR)).

The second parameter determines which character number inside one symbol buffer one is to start with (default is character number 0 (CR)).

The third parameter determines the input file (default is the terminal (CR)). The character (symbol) is defined by a matrix where each element is an X or a period (.); where X means that the corresponding bit in symbol matrix is set, while a period means it is not. When defining a character, X or period is typed in the proper sequence.

In addition, two characters are used (: and ;) to easily set a whole line or the rest of the line equal to one or zero where : means periods on the rest of the line and ; means x's on the rest of the line.

A character consists of 8 points in horizontal direction, but the number of lines in the matrix (grid) depends on the actual grid size (default are 8 lines).

For grid sizes up to 8 x 8, the command continues automatically to the next character when one grid is finished.

For grid sizes greater than 8 x 8 (8 x 9 - 8 x 16), the command will continue to update the corresponding character using the next symbol buffer before the next character is begun.

The upper half of a symbol in such grids is in one symbol buffer, and the lower half of the symbol is in the corresponding position in the next buffer.

Symbol buffers 1 and 3 are used for the upper half of symbols greater than 8 x 8 grids, and symbol buffers 2 and 4 are used for the lower half of the symbols.

Any character different from the four described (. - x - ; - :) will terminate the command.

Example:

```
*W-SYMB
SYMBOL BUFFER (1/2/3/4): 1
NO.: 1
INPUT FILE:

                              :
                              . . . . . . . .
                              . . X X X X . .
                              . X . . . . X .
                              . X . . . . X .
                              . X X X X X X .
                              . X . . . . X .
                              . X . . . . X .
                              . . . . . . . .
                              :
                              E
```

In this example, the character number one in the symbol buffer one is written in from the terminal.

The grid size is 8 x 8 in this example, and after changing the specified character the command is terminated (by typing E).

2.3.3 *CHECK-SYMBOL <SYMBOL BUFFER (1/2/3/4)> <No.>*

This command prints on the terminal the character number specified by the two parameters and in the same way as for the previous command, the actual grid presented by the use of periods (.) and x's.

Example:

```
*CH-SYM
SYMBOL BUFFER (1/2/3/4): 1
NO.: 1
```

```
. . . . . . . .
. . X X X X . .
. X . . . . X .
. X . . . . X .
. X X X X X X .
. X . . . . X .
. X . . . . X .
. . . . . . . .
```

2.3.4 *SAVE-SYMBOLS <SYMBOL BUFFER (1/2/3/4)><Output File>*

One or all of the present symbol buffers are saved on a file to be used whenever required in connection with the WRITE-SYMBOL command.

2.3.5 *SET-SYMBOL-BUFFER <No.>*

All further output on the monitor will be done with the symbol buffer specified by the parameter <no.> .

2.4 *THE PICTURE COMMANDS*

The following picture commands are available:

- COPY-FILE
- READ-FILE
- SAVE-PICTURE
- WRITE-PICTURE

2.4.1 *COPY-FILE <Input File>*

COPY-FILE is used to copy a picture file to the NCT to enable one to look at and/or update the picture, or the command can be used to draw a new picture directly from the terminal.

While executing this command it is possible to change window, blink, direction mode, cursor follow/stop, select symbol buffer, foreground and background using the control characters described in Appendix A of this manual.

One or two digits specifying the relevant information must be typed in after the various control characters (which are engraved buttons on an NCT keyboard).

Example:

FG (control A) must be followed by two decimal digits (00 to 15) specifying the actual FG level.

BG (control C) must be followed by one decimal digit (0 to 7) specifying the actual BG level.

Block (control Y) must be followed by one decimal digit (1 to 4), specifying the actual symbol buffer.

Window (control X) must be followed by two decimal digits (00 to 15) specifying the actual down line on the windowing.

All information copied by this command is saved in two buffers.

One is the refresh image buffer used in connection with the "write-picture" command, and the other is the dynamic buffer used in connection with the "save-picture" command. These buffers are also described in corresponding commands.

Example:

*COPY-F PICT

The picture on the file PICT is copied to the NCT.

*COPY-F ↵

A^C 01 C^C3 DEMONSTRATION W^C

*

In this example, foreground number 1 and background number 3 are selected before the text "DEMONSTRATION" is written on the monitor. W^C (control w) will terminate the COPY-FILE command.

2.4.2 *SAVE-PICTURE <Output File>*

This command is used to save, on a file, all information written on the NCT since the last "CLEAR" command.

This information is stored sequentially in the so-called internal dynamic buffer, and the only character which clears this buffer is the clear character (14_8).

This information also includes everything written to the symbol and colour buffers. Thus, the desired symbol and colour set may be included in a picture file and/or a dynamic changing of symbols and colours is possible since this internal buffer has all information from the latest "WRITE-SYMBOL" and "WRITE COLOUR" commands. When such a file is copied to the monitor screen, the picture will be drawn in the same sequence as that in which the information was given to the NCT.

2.4.3 *WRITE-PICTURE <Output File>*

This command will save, on a file, the current contents of the refresh image buffer. This buffer is continually updated and will always contain an exact representation of the present monitor picture. It is only the refresh image which is saved on the file, not the symbol or the colour buffers.

A combination of "WRITE-PICTURE" and "SAVE-PICTURE" commands is the best way to store a complete picture.

Note that a "WRITE-PICTURE"command must be followed by the relevant "SAVE-PICTURE" command, otherwise conflicts will arise in the refresh image buffer.

When copying a "WRITE-PICTURE" file to the monitor screen again, the picture will be drawn in the same sequence as the refresh buffer (from the upper left part of the monitor and downwards).

2.5 *THE MACRO COMMANDS*

- CREATE-MACRO
- LOOK-AT-MACRO
- MOVE-MACRO
- CHECK-MACRO-FILE
- DELETE-MACRO-NAME

2.5.1 *CREATE-MACRO <Macro File> <Name> <No. of Lines> <No. of Col.>*

Here, a macro refers to part of a picture element built up in the way described in the "COPY-FILE" command.

This picture element may be inspected at any time, and at any position on the screen. One macro file may contain several macros where the <name> parameter identifies each individual macro.

The <name> parameter only takes account of the first 6 characters written.

The two last parameters (<no. of lines>, <no. of col.>) determine the size of the picture element. A macro can always be created from the terminal.

Example:

```
*CREATE-M
MACRO FILE: PICTURES
NAME: PART1
NO. OF LINES: 3
NO. OF COL.: 4
-READY-
Yᶜ2 ABCD
    EFGH
    IJKLWᶜ
*
```

This macro creation starts by selecting symbol buffer number 2 (control Y, 2).

Then, the first line contains the characters numbered 1, 2, 3 and 4 (A, B, C and D).

The user must then position the cursor correctly to the starting point of the next line by using the direction codes (the four arrows: → ↓ ← ↑), and the next four characters from symbol buffer number 2 are typed in. The same procedure is carried out for the third line after which the macro is terminated using W^C (control W).

All the control functions may be included in a macro. The symbol buffer, foreground and background information may either be included in the macro definition or be chosen when the macro is used (see the following two commands).

2.5.2 *LOOK-AT-MACRO <Macro File> <Name> <Line> <Col.> <FG/BG/SB>*

This command will fetch the actual macro specified by the two first parameters (<macro file> and <name>) and place it on the monitor screen at the position specified by the two following parameters (<line> and <col.>). These parameters refer to the upper left part of the macro picture.

The three last parameters determine with which foreground, background and symbol buffer the macro shall be shown. If this information is included in the macro, the answer for the relevant parameter should be CR. It is a prerequisite for the next command to be described ("MOVE-MACRO") that the relevant macro first be fetched using a "LOOK-AT-MACRO" command.

Example:

```
*LOO-AT-MA
MACRO-FILE: PICTURES
NAME: PART1
LINE: 2
COL.: 3
FG: 1
BG: 0
SB: ↵
*
```

This example will fetch the specified macro (PICTURES (PART1)) and display it on the monitor screen from the specified position (2, 3) with the foreground colour 1, background colour 0 and with the symbol buffer specified in the macro.

1. If the parameter has a value, the new value is used.
2. If the parameter has a default value, CR in the subsystem or 64_{10} in the BRF library, the value from the macro creation is used.
3. If the parameter is equal to 63_{10} the actual value from that special place in the refresh memory is used.

This concerns LOOK-AT-MACRO and MOVE-MACRO.

2.5.3 *MOVE-MACRO <Line> <Col.> <FG> <BG> <SB> <Keep Old (Y/N)>*

The macro last fetched using the "LOOK-AT-MACRO command may be moved using this command. The position the macro is moved to is determined by the first two parameters (line, column), and the three next parameters determine the new foreground and background colour and the symbol buffer to be used in the same way as described in the "LOOK-AT-MACRO" command. A macro may be moved as many times as desired and the cursor blinks at the macro which it is possible to move.

The last parameter <keep old (Y/N)> determines whether the old picture element is to remain on the picture (Y) or not (N).

Example:

```
*MOVE-M
LINE: 6
COL.: 7
FG: 2
BG: 1
SB: 3
KEEP OLD (Y/N) N
*
```

Here the relevant macro is moved into position 6, 7 with a new foreground colour 2, a new background colour 1 and a new symbol buffer 3.

The old picture element is removed.

2.5.4 *CHECK-MACRO-FILE <Macro File>*

The command will list, on the terminal, all the macro names on the specified macro file.

Example:

```
*C-M-F PICTURES
1   PART1
2   PART2
3   PART3
*
```

In this example, the macro file PICTURES consists of three entries named PART1, PART2 and PART3.

2.5.5 *DELETE-MACRO-NAME<Macro File><Name-Number . . (1-77)>*

This command will delete one entry in a macro file. The actual entry is specified by a number which refers to the list given by the check-macro-file command.

Example:

```
*DEL-MA-NA PICTURES  3
```

2.6 *THE KEYBOARD COMMAND*

One keyboard command is available:

- SET-COMMAND-FIELD

2.6.1 *SET-COMMAND-FIELD <Line> <Col.> <No. of Lines>*

This command must be given immediately after the "NCT-NAME" command if an NCT keyboard is used.

The command will define a picture command field (specified by the three parameters) which displays the NCT-SP commands.

The NCT is now being used as an ordinary terminal.

The command field may be moved whenever required by repeating the command.

This is done whenever one wishes to avoid the mixing on the screen of picture data and command output.

When using the NCT keyboard the symbol buffer number 1 should always contain the ASCII character set to make it easier to read the commands.

Example:

```
*SET-C-F
LINE:   33
COL.:   0
NO. OF LINES:   16
*
```

In this example, the command field is moved to line 33, column 0 and consists of 16 lines.

When the last line in a command field is reached, the next output will be printed on the first line of the command field.

2.7 *AUXILARY COMMANDS*

The other commands available are:

- LEFT/RIGHT/UP/DOWN
- GRID-SIZE
- CURSOR
- BLINK
- SET-ADDRESS
- WINDOW
- STATUS
- HELP
- EXIT

2.7.1 *LEFT/RIGHT/UP/DOWN*

When characters are transmitted sequentially to the refresh buffer, the normal text mode is to step the address automatically towards the right hand side of the line, but when working with semigraphic pictures it is just as normal to step in any direction: right, left, up or down. The above commands are used to specify the step direction (right direction is default). The same result can be achieved by using the special buttons on an NCT keyboard or by using the following control characters in a "COPY-FILE" command: Q^C = right, R^C = left, T^C = up and S^C = down.

2.7.2 *GRID-SIZE <Height 1 to 16>*

The default value of the grid height is 8 points.

This grid height can, whenever required, be decremented or incremented by using this command.

Besides the actual picture on the monitor screen, this command will also influence the "WRITE-SYMBOL" command; symbols written from now on will be terminated at the new grid height.

Example:

GRID <u>6</u>

The actual picture on the monitor will be shown in an 8 x 6 grid and characters written from now on will be terminated after 6 lines, and the next character will automatically be started.

2.7.3 *CURSOR <ON/OFF>*

This command enables one to either let the cursor follow the picture drawing all the time (ON (default)) or to hold the cursor in a fixed position (OFF). The same effect is achieved by using the control characters 36 (ON), 37 (OFF).

2.7.4 *BLINK <ON/OFF>*

Blink off is the default mode. After a blink on command is given further information output to the screen monitor will blink (foreground and background blink with different frequencies). The same effect can be obtained by using the following control character Z^C.

2.7.5 *SET-ADDRESS <Line><Col.>*

An absolute address determined by the two parameters can be specified whenever required using this command. All further output will continue from this address.

2.7.6 *WINDOW <address>*

The address parameter here is the number of lines the picture is to move upwards. If an 8 x 8 grid picture is drawn, only the first 36 lines in the refresh buffer are shown on the screen. The 12 last lines can be displayed by giving this command with the address parameter equal to 12.

Example:

```
*WIND 12
```

The picture is moved 12 lines upwards. The 12 last lines are now visible but the first 12 lines will now have moved off the screen.

Example:

```
*Wind 0
```

The picture is moved back to show the 36 first lines (its original position). The same possibility can be obtained by using the control character X^C followed by two decimal digits (00 to 12).

2.7.7 STATUS

The following information is printed on the terminal:

The current line
The current column
The current foreground
The current background
The current symbol buffer
The current blink mode
The current grid height
The current window address
The current move direction
The current cursor mode

2.7.8 HELP

Help lists on the terminal all available commands and the parameters needed for each of them.

```
HELP
EXIT
LOOK-AT-COLOUR <(FG/BG)> <NO>
FOREGROUND <NO>
BACKGROUND <NO>
WRITE-COLOUR <(FG/BG/CU)> <NO> <INPUT FILE>
SAVE-COLOUR <OUTPUT FILE>
CHECK-COLOUR <(FG/BG/CU)> <NO>
LOOK-AT-SYMBOL <SYMBOL BUFFER (1/2/3/4)> <NO>
SET-SYMBOL-BUFFER <SYMBOL BUFFER (1/2/3/4)>
WRITE-SYMBOL <SYMBOL BUFFER (1/2/3/4)> <NO> <INPUT FILE>
SAVE-SYMBOLS <SYMBOL BUFFER (1/2/3/4)> <OUTPUT-FILE>
CHECK-SYMBOL <SYMBOL BUFFER (1/2/3/4)> <NO>
LEFT
RIGHT
UP
DOWN
GRID-SIZE <HIGHT (1-16)>
CURSOR <ON/OFF>
BLINK <ON/OFF>
SET-ADDRESS <LINE> <COLOUMN>
WINDOW <ADDRESS>
SAVE-PICTURE <OUTPUT-FILE>
COPY-FILE <INPUT FILE>
WRITE-PICTURE <OUTPUT-FILE>
NCT-NAME <FILE NAME>
CLEAR
STATUS
CREATE-MACRO <MACRO FILE> <NAME> <NO OF LIN.> <NO OF COL.>
MOVE-MACRO <LINE> <COL> <FG> <BG> <SB> <KEEP OLD (Y/N)>
LOOK-AT-MACRO <MACRO FILE> <NAME> <LINE> <COL> <FG> <BG> <SB>
CHECK-MACRO-FILE <MACRO FILE>
DELETE-MACRO-NAME <MACRO FILE> <NAME-NUMBER (1-77)>
SET-COMMAND-FIELD <LINE> <COL.> <NO. OF LINES>
*
```

2.7.9 *Exit*

Returns from the NCT-SP back to SINTRAN III.

2.8 *NCT-SP BRF LIBRARY*

Most of the NCT-SP commands can be called from other subsystems. The actual calls with their parameters are:

NCT-NAME:	CALL NCTNA (<device name>)
Example:	
FORTRAN:	CALL NCTNA ('NCT')
BASIC:	CALL NCTNA ("NCT")
CLEAR:	CALL KLEAR
WRITE-COLOUR:	CALL WRICO (<FG, BG, CU>, <no.>, <input file>)
Example:	
FORTRAN:	CALL WRICO ('FG', 64, 'COLOURS')
BASIC:	WRICO ("FG", 64%, "COLOURS")
SAVE-COLOUR:	CALL SAVCO (<output file>)
Example:	
FORTRAN:	CALL SAVCO ('COLOURS')
BASIC:	CALL SAVCO ("COLOURS")
FOREGROUND:	CALL FORGR (<no.>)
Example:	
FORTRAN:	CALL FORGR (1)
BASIC:	CALL FORGR (1%)
BACKGROUND:	CALL BACGR (<no.>)
Example:	
FORTRAN:	CALL BACGR (7)
BASIC:	CALL BACGR (7%)
WRITE-SYMBOL:	CALL WRISY (<symbol buffer no.>, <char. no.>, <input file>)
Example:	
FORTRAN	CALL WRISY (2, 5, 'SYMBOLS')
BASIC:	CALL WRISY (2%, 5%, "SYMBOLS")
SAVE-SYMBOLS:	CALL SAVSY (<symbol buffer no.>, <output file>)
Example:	
FORTRAN:	CALL SAVSY (2, 'SYMBOLS')
BASIC:	CALL SAVSY (2%, "SYMBOLS")

SET-SYMBOL-BUFFER:	CALL SSYBU (<no.>)
Example:	
FORTRAN	CALL SSYBU (1)
BASIC:	CALL SSYBU (1%)
COPY-FILE:	CALL COPYF (<input file>)
Example:	
FORTRAN:	CALL COPYF ('PICTURE')
BASIC:	CALL COPYF ("PICTURE")
SAVE-PICTURE:	CALL SAVPI (<output file>)
Example:	
FORTRAN:	CALL SAVPI ('PICTURE')
BASIC:	CALL SAVPI ("PICTURE")
WRITE-PICTURE:	CALL WRIPI (<output file>)
Example:	
FORTRAN:	CALL WRIPI ('PICTURE')
BASIC:	CALL WRIPI ("PICTURE")
CREATE-MACRO:	CALL CREMA (<macro file>, <name>, <no. of lines>, <no. of col.>)
Example:	
FORTRAN:	CALL CREMA ('MACFI', 'PART1', 2, 3)
BASIC:	CALL CREMA ("MACFI", "PART1", 2%, 3%)
LOOK-AT-MACRO:	CALL LOAMA (<macro file>, <name>, <line>, <col,>,<fg>,<bg>,<sb>)
Example:	
FORTRAN:	CALL LOAMA ('MACFI', 'PART1', 10, 12, 64, 64, 64)
BASIC:	CALL LOAMA ("MACFI", "PART1", 10%, 12%, 64%, 64%, 64%)
MOVE-MACRO:	CALL MOVMA (<line>, <col>, <fg>, <bg>, <sb>, <keep old (Y/N)>)
Example:	
FORTRAN:	CALL MOVMA (22, 33, 3, 63, 64, 'Y')
BASIC	CALL MOVMA (22%, 33%, 3%, 63%, 64%, "Y")
LEFT/RIGHT/UP/DOWN:	CALL LEFT/CALL RIGHT/CALL UP/CALL DOWN
GRID-SIZE:	CALL GRISI (<height>)
Example:	
FORTRAN:	CALL GRISI (10)
BASIC:	CALL GRISI (10%)

CURSOR:	CALL CURSO (<ON/OFF>)
Example:	
FORTRAN:	CALL CURSO ('OFF')
BASIC:	CALL CURSO ("OFF")
BLINK:	CALL BLINK (<ON/OFF>)
Example:	
FORTRAN:	CALL BLINK 'ON')
BASIC:	CALL BLINK ("ON")
SET-ADDRESS:	CALL SETAD (<line>, <col.>)
Example:	
FORTRAN:	SETAD (20, 30)
BASIC:	CALL SETAD (20%, 30%)
WINDOW:	CALL WINDO (<lines up>)
Example:	
FORTRAN:	CALL WINDO (12)
BASIC:	CALL WINDO (12%)

Note:

When default values are wanted, code 64_{10} is used in the BRF part instead of CR.

3 PROGRAMMING SPECIFICATIONS

Seven bits coded characters are used to control NORDCOM TERMINAL. One additional bit used for parity-checking is supplied for serial transmission.

A standard TTY driver may be used for both input and output, but automatic echo on input must be omitted.

There are four main groups of characters. These are:

- Control Characters
- Control-Mode Characters
- Character-Mode Characters
- Address Mode Characters

3.1 *CONTROL CHARACTERS*

Control Characters have octal code from 0 to 37, and are used for functional control of transmission and the terminal. Some Control Characters are used to control the "mode" of the terminal.

The interpretation of the Control Characters is independent of which "mode" the terminal is set to.

3.1.1 *Transmission Control Characters*

STX (octal 2)

Set terminal open for transmission.

EOT (octal 4)

Set terminal closed for transmission. When the terminal is closed, the only characters to give a reaction are STX, RSTS, BELL and ENQ.

ENQ (octal5)

Request error status. The terminal checks for parity error and framing error. An error condition will affect the error status. ENQ resets the status.

ACK (octal 6)

Response character when no error has been detected since last ENQ.

NAK (octal 25)

Response character when an error has been detected since last ENQ.

RSTS (octal 6)

Read Status from NCT. The "Ignore Error" mode and the "lock switch" status may be read by this command. The returned character is as follows:

Bit (0)	= 1: IGNORE ERROR ON
Bit (0)	= 0: IGNORE ERROR OFF
Bit (1)	= 1: LOCK SWITCH OFF
Bit (1)	= 0: LOCK SWITCH ON
Bit (2-6)	= 1
Bit (7)	= Parity

BELL (octal 7)

Bell or alarm command. This command will start a one-shot (adjustable from 500 ms to 5 s), which will light the "alarm" lamp and/or make the alarm buzzer sound.

3.1.2 *Mode Control Characters*

The terminal may be in character mode or in control mode.

The interpretation of all characters except control characters will depend on which mode the terminal is set to.

SET-CHARACTER-MODE (octal 16)
SET-CONTROL-MODE (octal 17)

3.1.3 *Address Control Characters*

The address register contained in NCT is used in connection with the transfer of data byte to the Refresh Buffer and Symbol Buffer.

For the purpose of holding the address of the next position in the refresh buffer, the address register consists of Row Address and Column Address.

Each of these registers may be incremented or decremented automatically of under program control, or may be cleared or set to any absolute value.

3.1.4 *Set Address Absolute*

To transfer an absolute value to the Row and Column Address requires a sequence of three consecutive characters.

First Character:

ADDRESS HEADING (octal 20)
The following two characters are interpreted to be a special format.

Second Character:

ROW ADDRESS (octal value from 100 to 177)
Row Address greater than 157 means addressing of symbol buffer. Refer to Section 3.2.

Third Character:

COLUMN ADDRESS (octal value from 100 to 177)

3.1.5 *Automatic Address Step*

The NORDCOM TERMINAL will be in one of four modes for automatic step of address. Stepping of address takes place for each character transmitted to the refresh buffer.

The direction of address stepping is determined by the mode.

SET MODE RIGHT (octal 21)
SET MODE LEFT (octal 22)
SET MODE DOWN (octal 23)
SET MODE UP (octal 24)

3.1.6 *The Control Characters*

← (octal 10)
→ (octal 11)
↓ (octal 13)
↑ (octal 34)

These step the corresponding address independent of the automatic mode.

Clearing of the Row and Column Register is done by:

- Power-on clear
- Control character "ERASE PICTURE" (octal 14)
- Control character "HOME" (↖) (octal 35)

3.1.7 *Cursor Marker*

There is a visible cursor marker that by the control character "CURSOR LOCK" (octal 36) may be programmed to follow the current address position on the screen. The cursor may be presented in different colours (see colour control).

The control character "CURSOR UNLOCK" (octal 37) will hold the cursor marker on the specific position on the screen, whatever the contents of Row and Column registers may be thereafter.

3.1.8 *INIT (octal 33)*

Start hardware "ROM LOADER". This command is equal to power-up or manually clear, which starts the internal loading of the standard 64 symbol ASCII set, a complete standard colour set, a test picture in refresh buffer. In addition, the following initial state will be set:

> Right mode, character mode, 8 x 8 size, cursor follow, terminal open, down, up, all programmable lamps off, FG colour 15 (white), BG colour 0 (black), Block 0 and Blink off.
>
> The string is terminated with an ENQ which returns an ACK code (octal 6) on the line.
>
> The terminal is closed during the load sequence (about 1/8 s.) but the returned ACK may be used as a ready signal.

3.2 SYMBOL GENERATOR

The symbol generator utilizes a writeable store to specify each symbol. This store contains one bit for each point in the symbol.

When the terminal is in "Control Mode" the symbol size is specified by one of the following characters:

Octal 105 - 8 x 6 per symbol
Octal 107 - 8 x 8 per symbol
Octal 113 - 8 x 12 per symbol
Octal 117 - 8 x 16 per symbol

For the formats 8 x 6 and 8 x 8 the symbol store contains specifications of 256 different symbols.

For the formats 8 x 12 and 8 x 16 the symbol store specifies 128 different symbols.

Loading of symbol store is done in "Control Mode". The first step is to address the proper symbol position in the store, then follow the string of characters describing the symbol.

The Row and Column registers are used for addressing. The Row register shall hold the address of the symbol block. The first block of 64 symbols is addressed by entering oct. 160 to the Row register, the next blocks by entering oct. 164, 170 and 174.

The symbol address within a block is entered in the Column register by an octal value between 100 and 177. The string characters that follow shall be transmitted in the sequence indicated in the figure below. The binary value of the four least significant bits in each character defines the bit pattern for that part of the symbol. The character will then have a value between oct. 40 and oct. 57. The "ones" are presented on the screen in foreground colour. The "zeros" are presented in background colour.

An internal byte counter takes care of the addressing within one symbol. This byte counter is reset by any control character, but incremented by any other character. To reset the counter and to insure control mode the character "SET CONTROL MODE" (oct. 17) must be transmitted after the last address byte.

3 2 1 0	3 2 1 0
2.	1.
4.	3.
6.	5.
8.	7.
10.	9.
12.	11.
14.	13.
16.	15.

When the 16 characters have been received by the terminal, the Column register holding the symbol address is incremented automatically.

For symbol formats of 8 x 12 and 8 x 16 the symbol description is loaded into pairs of blocks. Block 0 and 1 define the first 64 symbols, and block 2 and 3 define the last 64 symbols. The loading of symbol store for these formats must be done with separate strings of 16 characters. Blocks 0 and 2 hold the upper half of the symbol.

3.3 COLOUR GENERATOR

The colour generator utilizes a writeable store to specify 16 foreground colours, 8 background colours and 8 cursor colours.

Each of these colours are described in a format of three codes of four bits each.

Each code is used to specify a certain level of colour intensity by giving input to the three primary colour guns of the RGB monitor. Code "0000" gives the highest intensity.

Loading of the colour store is done in "Control Mode" by transferring the following sequence of 96 characters, where the characters will have an octal value between 60 and 77.

1st to 16th RED Foreground level
1st to 8th RED Background level
1st to 8th RED Cursor level
1st to 16th GREEN Foreground level
1st to 8th GREEN Background level
1st to 8th GREEN Cursor level
1st to 16th BLUE Foreground level
1st to 8th BLUE Background level
1st to 8th BLUE Cursor level

Example:

A symbol at a specific position on the screen shall have Foreground colour number 5 and Background colour number 3 (the first colour has number 0).

Foreground symbol is then shown with the colour specified by the 6th code for RED, GREEN and BLUE foreground. The rest of the symbol matrix is shown with the colour specified by the 4th code for RED, GREEN and BLUE backgrounds.

If the cursor should appear in this position, it will be shown with the colour addressed by the background colour number, but shown with the colour specified for cursor in the colour store.

In the example, this will give the 4th code for RED, GREEN and BLUE cursor colours.

3.4 WINDOWING

The refresh buffer holds 48 rows (lines) of information. When an 8 x 6 symbol size is specified, all the 48 lines are shown at the same time, and then windowing has no significant meaning.

For the other symbol sizes, however, there will be a greater number of lines in the refresh buffer, than that shown on the screen at the same time.

The "WINDOW POINTER" character octal 140 to 177, entered in "Control Mode" will define the line number in the refresh buffer which appears as the topmost on the screen.

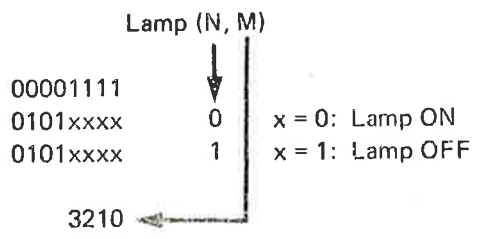
The table below shows the correspondance between the symbol format and the number of rows on the screen, and the window pointer that should be entered in order to get the 48th row on the lower line on the screen.

Symbol Format:	Rows on Screen:	Window Pointer:
8 x 6	48	octal 140
8 x 8	36	octal 154
8 x 12	24	octal 170
8 x 16	18	octal 176

3.5 LAMP FUNCTIONS

Eight predefined positions on the keyboard have keytops with built-in light bulbs. These lamps are controlled by two consecutive characters transmitted in "Control Mode". The four least significant bits in the character give information light/no-light. The characters are octal 120 to 137.

17 + 137 + 137:	All lamps OFF
17 + 120 + 120:	All lamps ON
17 + 137 + 127:	Lamp (1, 3) ON (F1), other OFF
17 + 137 + 133:	Lamp (1, 2) ON (F2), other OFF
17 + 137 + 135:	Lamp (1, 1) ON (SIZE), other OFF
17 + 137 + 136:	Lamp (1, 0) ON (BLOCK), other OFF
17 + 136 + 137:	Lamp (0, 0) ON (BLINK), other OFF
17 + 135 + 137:	Lamp (0, 1) ON (WINDOW), other OFF
17 + 133 + 137:	Lamp (0, 2) ON (FG COL), other OFF
17 + 127 + 137:	Lamp (0, 3) ON (BG COL), other OFF
17 + 136 + 127:	Lamp (0, 0) and (1, 3) ON, other OFF
	or any combination.



3.6 CHARACTER MODE

When in character mode all characters with octal value from 40 to 137 cause a 16 bit word to be written in the refresh memory to the position by Row and Column address.

The six least significant bits are taken directly from the character itself. Bits 6 and 7 are taken from the block flags, bits 8, 9, 10 and 11 from the FG colour flags, bits 12, 13 and 14 from the BG colour flags and bit 15 from the Blink flag.

All flags are set by characters greater than 137 as follows:

140 - 157:	Set FG colours (0 - 15)
160 - 167:	Set BG colours (0 - 7)
170 - 173:	Set BLOCK (0 - 3)
174 - 175:	Set BLINK (ON/OFF)

See Figure 3.1.

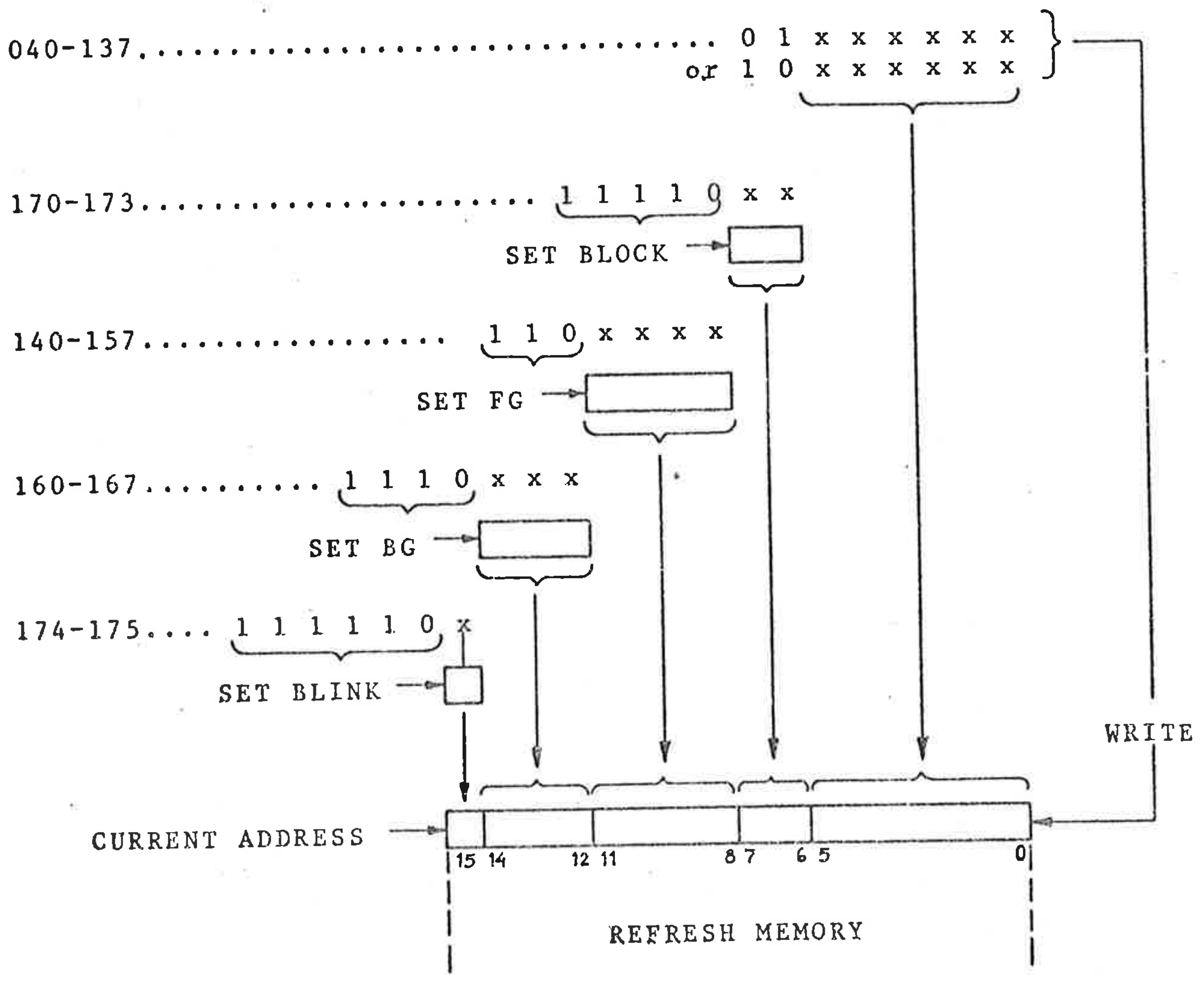
3.7 KEYBOARD

NORDCOM TERMINAL may be delivered with a keyboard which, in addition to the standard alphanumeric keys, has a set of control keys and 25 function keys.

The keyboard layout is specified in Appendix A. Keyboard control is designed for N-key roll-over. Key force is 80 - 100 grams.

Input codes from the keyboard are specified in Appendix B and C.

EFFECT OF CHARACTERS WHEN CHARACTER MODE



040-137 .... 0 1 x x x x x x
or 1 0 x x x x x x
170-173 .... 1 1 1 1 0 x x
SET BLOCK
140-157 .... 1 1 0 x x x x
SET FG
160-167 .... 1 1 1 0 x x x
SET BG
174-175 .... 1 1 1 1 1 0 x
SET BLINK
WRITE
CURRENT ADDRESS
15 14 12 11 8 7 6 5 0
REFRESH MEMORY

Figure 3.1

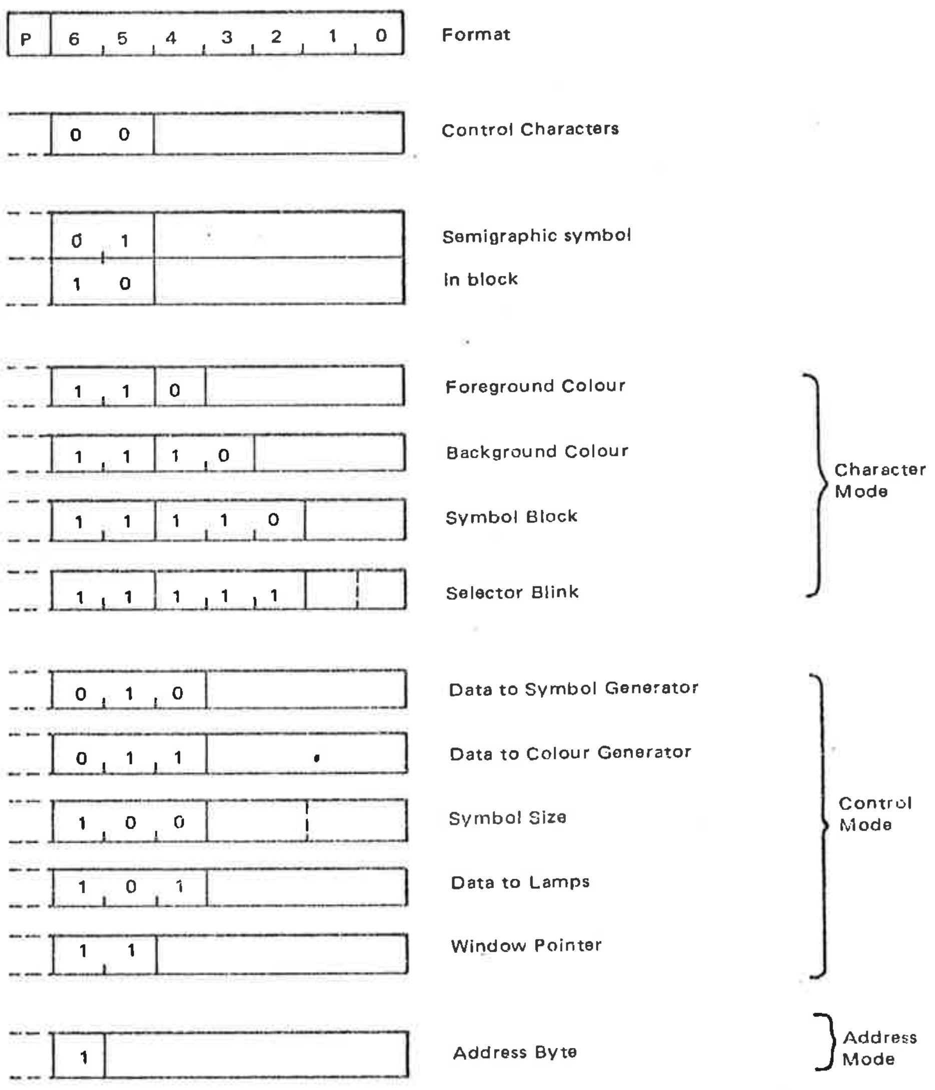
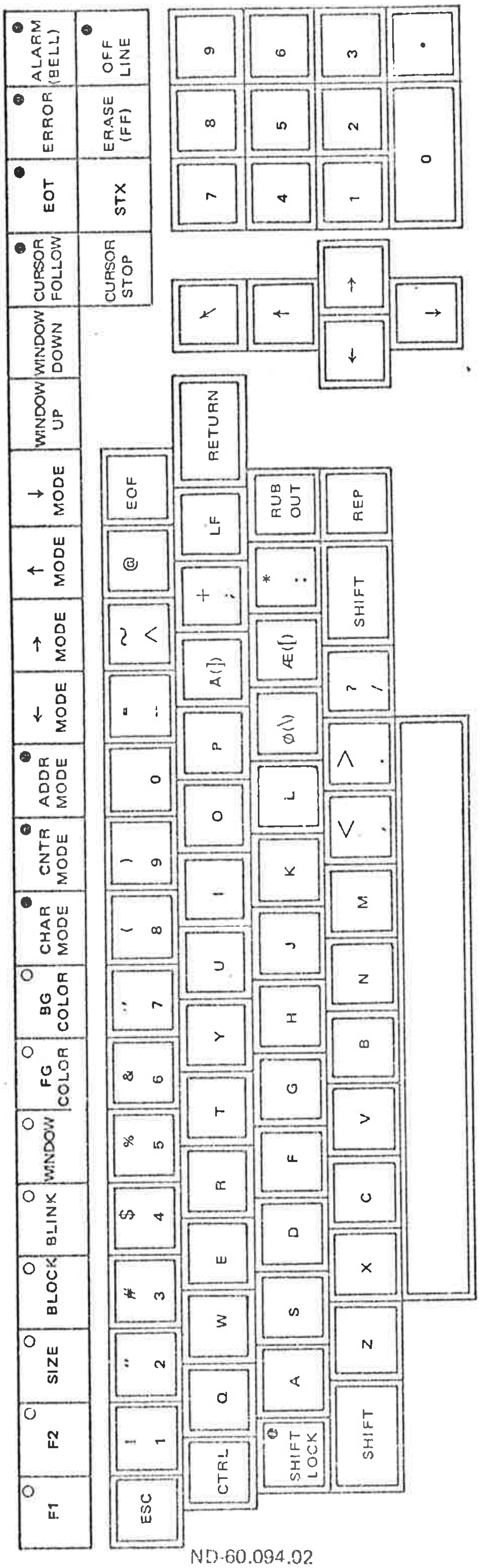

P 6 5 4 3 2 1 0
Format
0 0
Control Characters
0 1
1 0
Semigraphic symbol
In block
1 1 0
Foreground Colour
1 1 1 0
Background Colour
1 1 1 1 0
Symbol Block
1 1 1 1 1
Selector Blink
Character Mode
0 1 0
Data to Symbol Generator
0 1 1
Data to Colour Generator
1 0 0
Symbol Size
1 0 1
Data to Lamps
1 1
Window Pointer
Control Mode
1
Address Byte
Address Mode

Figure 3.2

APPENDIX A


F1
F2
SIZE
BLOCK
BLINK
WINDOW
FG COLOR
BG COLOR
CHAR MODE
CNTR MODE
ADDR MODE
← MODE
→ MODE
↑ MODE
↓ MODE
WINDOW UP
WINDOW DOWN
CURSOR FOLLOW
EOT
ERROR
ALARM (BELL)
CURSOR STOP
STX
ERASE (FF)
OFF LINE
ESC
! 1
" 2
# 3
$ 4
% 5
& 6
' 7
(8
) 9
0
= -
~ ^
@
EOF
CTRL
Q
W
E
R
T
Y
U
I
O
P
Å(])
+ ;
LF
RETURN
SHIFT LOCK
A
S
D
F
G
H
J
K
L
Ø(\)
Æ([)
* :
RUB OUT
SHIFT
Z
X
C
V
B
N
M
< ,
> .
? /
SHIFT
REP
7
8
9
4
5
6
1
2
3
0
.

APPENDIX B

CODES FROM KEYBOARD

KEY TEXT	Wir. x y	Enc. out (oct)	Octal value of character			
			Norm	Shift	CTRL	C+S
@	00	000	100	100	000	000
A	01	001	141	101	001	001
B	02	002	142	102	002	002
C	03	003	143	103	003	003
D	04	004	144	104	004	004
E	05	005	145	105	005	005
F	06	006	146	106	006	006
G	07	007	147	107	007	007
H	08	010	150	110	010	010
I	09	011	151	111	011	011
J	10	012	152	112	012	012
K	11	013	153	113	013	013
L	12	014	154	114	014	014
M	13	015	155	115	015	015
N	14	016	156	116	016	016
O	15	017	157	117	017	017
P	16	020	160	120	020	020
Q	17	021	161	121	021	021
R	18	022	162	122	022	022
S	19	023	163	123	023	023
T	20	024	164	124	024	024
U	21	025	165	125	025	025
V	22	026	166	126	026	026
W	23	027	167	127	027	027
X	24	030	170	130	030	030
Y	25	031	171	131	031	031
Z	26	032	172	132	032	032
Æ ({ [)	27	033	173	133	033	033
Ø (| \\)	28	034	174	134	034	034
Å (}])	29	035	175	135	035	035
^ ~	30	036	176	136	176	136
DEL –	31	037	177	137	177	137
0	32	040	060	060	060	060
1	33	041	061	061	061	061
2	34	042	062	062	062	062
3	35	043	063	063	063	063
4	36	044	064	064	064	064
5	37	045	065	065	065	065
6	38	046	066	066	066	066
7	39	047	067	067	067	067
8	40	050	070	070	070	070
9	41	051	071	071	071	071
	42	052	172	172	172	172
	43	053	173	173	173	173
	44	054	174	174	174	174
	45	055	175	175	175	175
.	46	056	056	056	056	056
WINDOW DOWN	47	057	177	177	177	177

KEY TEXT	Wir. x y	Enc. out (oct)	Octal value of character			
			Norm	Shift	CTRL	C+S
0	48	060	060	060	060	060
1 !	49	061	061	041	061	041
2 "	50	062	062	042	062	042
3 #	51	063	063	043	063	043
4 $	52	064	064	044	064	044
5 %	53	065	065	045	065	045
6 &	54	066	066	046	066	046
7 '	55	067	067	047	067	047
8 (	56	070	070	050	070	050
9)	57	071	071	051	071	051
: *	58	072	072	052	072	052
; +	59	073	073	053	073	053
, <	60	074	054	074	054	074
– =	61	075	055	075	055	075
. >	62	076	056	076	056	076
/ ?	63	077	057	077	057	077
SPACE	64	100	040	040	040	040
FG COLOR ○	65	101	001	001	001	001
STX	66	102	002	SAME AS NORMAL		
BG COLOR ○	67	103	003			
EOT ●	68	104	004			
ERROR ●	69	105	005			
ACK *)	70	106	006			
ALARM (BELL) ●	71	107	007			
←	72	110	010			
→	73	111	011			
LF	74	112	012			
↓	75	113	013			
ERASE	76	114	014			
CR	77	115	015			
CHAR. MODE ●	78	116	016			
CNTR MODE ●	79	117	017			
ADDR. MODE ●	80	120	020			
→ MODE	81	121	021			
← MODE	82	122	022			
↓ MODE	83	123	023			
↑ MODE	84	124	024			
NAK *)	85	125	025			
SIZE ○	86	126	026			
EOF	87	127	027			
WINDOW ○	88	130	030			
BLOCK ○	89	131	031			
BLINK ○	90	132	032			
ESC	91	133	033			
↑	92	134	034			
HOME	93	135	035			
CURSOR FOLLOW ●	94	136	036			
CURSOR STOP	95	137	037			
WINDOW UP	96	140	140			
F1 ○	97	141	041			
F2 ○	98	142	042			

● :	HARDWARE CONTROLLED LAMPS
○ :	PROGRAMMABLE LAMPS
*) :	HARDWARE GENERATED (INITIATED)

APPENDIX C

	0	1	2	3	4	5	6	7
0	$@^c$	FG COLOR A^c	STX B^c	BG COLOR C^c	EOT D^c	ERROR E^c	F^c	ALARM (BELL) G^c
10	← H^c	→ I^c	LF J^c	↓ K^c	ERASE (FF) L^c	CR M^c	CHAR MODE N^c	CNTR MODE O^c
20	ADDR MODE P^c	→ MODE Q^c	← MODE R^c	↓ MODE S^c	↑ MODE T^c	U^c	SIZE V^c	EOF W^c
30	WINDOW X^c	BLOCK Y^c	BLINK Z^c	ESC $Æ^c$	↑ $Ø^c$	HOME $Å^c$	CURSOR FOLLOW	CURSOR STOP
40	SPACE	F1 !	F2 "	#	$	%	&	'
50	(	)	*	+	,	—	.	/
60	0	1	2	3	4	5	6	7
70	8	9	:	;	<	=	>	?
100	@	A	B	C	D	E	F	G
110	H	I	J	K	L	M	N	O
120	P	Q	R	S	T	U	V	W
130	X	Y	Z	Æ	Ø	Å	∧	—
140	WINDOW UP	a	b	c	d	e	f	g
150	h	i	j	k	l	m	n	o
160	p	q	r	s	t	u	v	w
170	x	y	z	æ	ø	å	~	WINDOW DOWN DEL

NORSK DATA A.S.
Lørenveien 57 - Postboks 163, Økern
OSLO 1

COMMENT AND EVALUATION SHEET

NCT – NORD Colour Terminal User's Guide
February 1978

Publ. No. ND-60.094.02

In order for this manual to develop to the point where it best suits your needs, we must have your comments, corrections, suggestions for additions, etc. Please write down your comments on this pre-addressed form and post it. Please be specific wherever possible.

FROM ______________________________

____

____