Contents

UBI EasyCoder 301
Direct Protocol
Programmer's Guide
Edition 1, July 1997
Part No. 1-960419-00

Contents, cont'd.

Contents, cont'd.

Information in this manual is subject to change without prior notice and does not represent a commitment on the part of UBI Printer AB.

Introduction

UBI Direct Protocol is an easy-to-use printer programming language that has been developed for use with the computer-controlled direct thermal and thermal transfer printers manufactured by United Barcode Industries (UBI).

UBI Direct Protocol can be used in two ways:

- To create label layouts consisting of fields with fixed or variable information. A layout can then be selected and provided with variable input from the host computer in the form of a simple string of data.
- To send input data and formatting commands as a continuous string of data directly from the host computer.

In both cases, UBI Direct Protocol provides a flexible error handler, which allows you to compose your own error messages in any language.

UBI Direct Protocol has been created with two main types of application in mind:

- Applications where the end-user requires a simple printer control program, but needs a versatile error handler.
- Applications which have a comprehensive printer control program in the host computer.

Refer to chapter 13 in this manual for lists of the various commands in UBI Direct Protocol.

We recommend that you have the following manuals accessible:

- EasyCoder 301 Installation and Operation Manual
- This Programming Manual.

Getting Started

1. Computer Connection

UBI Direct Protocol is stored in two Flash EPROM packages fitted as standard on the printer's CPU board. No floppy disks or operating system, e.g. MS-DOS, are required. The printer only needs to be connected to a mains supply and to a device capable of transmitting characters in ASCII format, which can be anything from a non-intelligent terminal to a mainframe computer system.

For running the printer, we recommend a computer with a screen, an alphanumeric keyboard, and a communication program which provides two-way communication via an RS 232C serial port. A Centronics parallel port can also be used, but it will provide one-way communication only: no data or messages can be returned to the screen of the host.

Connect the printer and host as described in the EasyCoder 301 Installation and Operation Manual.

It is possible to set up the printer's communication protocol to fit the host computer, as described in the EasyCoder 301 Installation and Operation Manual. However, until you have become familiar with UBI Direct Protocol, it may be easier to adapt the host to the printer's default setup parameters. There is no communication setup for the Centronics parallel interface.

Default serial communication setup on "uart1:"

- Baud rate ..... : 9600
- Parity ..... : None
- Character length ..... : 8
- No. of stop bits ..... : 1
- Flow control ..... : Disabled
- New line ..... : CR/LF (*Carriage Return + Line Feed*)

2. Check Paper Supply

Check that the printer has an ample supply of paper or other receiving material and, where applicable, of thermal transfer ribbon. Refer to the EasyCoder 301 User's Guide or Installation and Operation Manual for loading commands.

3. Turn on the Printer

Check that the printhead is lowered. Turn on the mains switch, which is fitted on the printer's rear plate, and check that the **Power** LED comes on.

4. Serial Communications Test

Check that you have a working two-way serial communication by sending a simple command from the host to the printer. On the host, type:

```
? VERSION$ ↵
```

(↵ = Carriage Return)

Provided you have serial two-way communication, the printer should respond by immediately returning the version of the installed UBI Direct Protocol software to the screen of the host, e.g.:

```
D6.1
Ok
```

This indicates that the communication is working both ways.

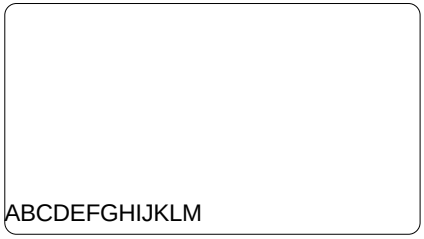
If the communication does not work, turn off the printer and check the connection cable. Also check if the communication setup in the host corresponds to the printer's setup and if the connection is made between the correct ports. Then try the communication test again.

Once you know that the communication is working, you may go on and send a line of text to make sure that characters transmitted from the terminal are interpreted as expected by the printer's software:

```
PP 10, 10 ↵
FT "Swiss 721 BT" ↵
PT "ABCDEFGHIJKLM" ↵
PF ↵
```

4. Serial Communications Test, cont'd.

Each line will be acknowledged by "**Ok**" on the screen, provided it has been entered correctly. When you press ↵ (*Carriage Return*) the third time, the printer will feed out a label, ticket, tag or piece of strip with the text printed in the lower left corner of the printable area.



You can try using other characters between the quotation marks in the third line, especially typical national characters like ÅÄÖÜ¿¢¥ç, etc. If any unexpected characters are printed, you may need to select another character set (see chapter 6.11), or change the communication setup from 7-bit to 8-bit character length.

Principles of Operation

1. UBI Direct Protocol Special Features

There is a built-in error handler, that can indicate selected error conditions and produce error-messages of your own creation in any language you like.

Variable input data (in a special format) can be inserted into fields in a predefined layout.

You can create counters without extensive programming.

You can easily enable the **Feed** key to produce printouts.

By default, verbosity is turned on in UBI Direct Protocol[1].

UBI Direct Protocol allows you to send commands to the printer in two ways:

- Immediate Mode
- Layout Mode

2. Sending Commands

- **Immediate Mode**
  You send commands that the printer will act upon immediately. This method is used for setting up the printer, for returning information from the printer back to the host and for managing files, fonts and images.

  Examples:

  `PRINT KEY ON` ↵ *(enable **Feed** key)*
  `? DATE$` ↵ *(read printer's calendar)*
  `KILL "LAYOUT1"` ↵ *(delete a layout or file)*

  Immediate Mode can also be used for creating label layouts including all the printable data as illustrated below.

- **Layout Mode**
  Layout Mode is only intended for creating labels. Firstly, create a layout containing formatting commands for a number of fields and such printable data that you do not need to change. Secondly, send a string of printable data to the empty fields plus a print command. You can create a number of different layouts, select the one you need, add the variable data and print out the labels.

[1]/. While you are experimenting with UBI Direct Protocol, we recommend that you retain the default verbosity setting (ON). Later you should change it to OFF, since this will produce optimum printer performance.

2. Sending Commands, cont'd.

Let us create the simple label shown on the left using both methods:

Immediate Mode:

```
BF ON:BF "Swiss 721 BT",10,0:PP 10,20:
PX 400,300,10:PP 25,25:PM "ROM:UBI.1":
PP 75,250:BT "CODE39":PB "UBI":PP 75,200:
FT "Swiss 721 BT",10,0:PT "My FIRST label!":
PF ↵
```

You can also send the same string line by line:

```
BF ON ↵                          (enable bar code interpretation)
BF "Swiss 721 BT",10,0 ↵         (select bar code interpr. font)
PP 10, 20 ↵                      (insertion point for box field)
PX 400,300,10 ↵                  (create a box)
PP 25,25 ↵                       (insertion point for image field)
PM "ROM:UBI.1" ↵                 (select image)
PP 75,250 ↵                      (insertion point for bar code field)
BT "CODE39" ↵                    (select bar code type)
PB "UBI" ↵                       (input data to bar code field)
PP 75,200 ↵                      (insertion point for text field)
BF "Swiss 721 BT",10,0 ↵         (select font for text field)
PT "My FIRST label!" ↵           (input data to text field)
PF ↵                             (print one label)
```

Layout Mode:

First, create a layout:

```
LAYOUT INPUT "LABEL1" ↵          (start layout recorder)
BF ON ↵                          (enable bar code interpretation)
BF "Swiss 721 BT",10,0 ↵         (select bar code interpr. font)
PP 10,20 ↵                       (insertion point for box field)
PX 400,300,10 ↵                  (create a box)
PP 25,25 ↵                       (insertion point for image field)
PM "ROM:UBI.1" ↵                 (select image)
PP 75,250 ↵                      (insertion point for bar code field)
BT "CODE39" ↵                    (select bar code type)
PB VAR1$ ↵                       (variable input data to bar code field)
PP 75,200 ↵                      (insertion point for text field)
BF "Swiss 721 BT",10,0 ↵         (select font for text field)
PT VAR2$ ↵                       (variable input data to text field)
LAYOUT END ↵                     (save layout)
```

Note:
If a label has been printed using a predefined layout and you want to return to Immediate Mode, the predefined layout must first be cleared from the printer's working memory using the following command:
LAYOUT RUN ""

2. Sending Commands, cont'd.

Then add the variable input data and a print command:

```
LAYOUT RUN "LABEL1" ↵          (select layout)
<STX>                          (start of input data, ASCII 02 dec)
UBI ↵                          (variable input data to VAR1$)
My FIRST label! ↵              (variable input data to VAR2$)
<EOT>                          (end of input data, ASCII 04 dec)
PF ↵                           (print one label)
```

3. Fields

The printable information on a label, ticket, tag or piece of strip consists of various types of field. A field can consist of:

- A single line of text
- A bar code with or without human readable interpretation
- An image, i.e. a picture, logo or sign in .PCX format
- A box, i.e. a hollow square or rectangle
- A line

4. General Formatting Commands

Any type of field should be specified with regard to:

- Position
- Alignment
- Direction

Refer to chapter 4.2 for more information.

5. Field-Related Formatting Commands

Depending on the type of field, additional formatting commands can be used:

- ***Text Field:***
  - Font (typeface)
  - Font Size (in points)
  - Font Slant (in degrees)
  - Magnification
  - Normal Image/Inverse Image
- ***Bar Code Field[1]:***
  - Bar Code Type
  - Height (height of bar pattern)
  - Ratio (wide bars/narrow bars)
  - Magnification (bar pattern)
  - Bar Code Interpretation On/Off
  - Bar Code Interpretation Font
  - Bar Code Interpretation Font Size
  - Bar Code Interpretation Font Slant

[1]/. This refers to one-dimensional bar codes only. Complex two-dimensional bar or dot codes may have other formatting parameters.

5. Field-Related Formatting Commands, cont'd.

- ***Image Field:***
  - Magnification
  - Normal Image/Inverse Image
- ***Box Field:***
  - Size (height, width, line thickness)
- ***Line Field:***
  - Size (length, line thickness)

Text, bar code and image fields also require some input data:

- Text Field: Alphanumeric text
- Bar Code Field: Alphanumeric or numeric value (depending on type of code)
- Image Field: Name of the image

The input data to text and bar code fields may either be provided by the host or read from the printer's software, e.g. counter values or various data related to the printer's clock/calendar.

In UBI Direct Protocol, you can either specify the fields both with regard to formatting parameters **and** input data in the same string, **or** you can create a layout to which you can send variable data later.

The layout should contain formatting parameters for all fields and input data to such fields that will always contain the same information. When the variable input data are added, they will be inserted into their respective fields (like filling in a pre-printed form).

6. Layout Commands

When creating predefined layouts, special commands must be used for:

- Starting the layout recorder
- Saving the layout

7.Printable Data Commands

Depending on the type of field and the type of bar code, printable data to text and bar code fields may consist of:

- Alphanumeric data (i.e. text)
- Numeric data
- Counter values
- Current date
- Current time
- Current date +/- nn days
- Current time +/- nn seconds
- Current week number
- Current weekday

8. Feeding and Printing Commands

Some commands control printing and paper feed, e.g.:
- Changing paper feed speed
- Rotating the print roller during cleaning
- Feeding out an empty label, ticket, tag or piece of strip
- Adjusting the label stop/black mark sensor.
- Printing one label or a batch of labels (or similar).

9. Setting Up the Printer

You can control how the printer will work, e.g.:
- Select standard IN and OUT channels
- Set the printer's clock/calendar
- Set formats for the printing of date and time
- Set separators for input data strings to predefined layouts
- Create label counters
- Enable/disable the optional label-taken sensor
- Enable/disable **Feed** key
- Format the printer's RAM memory or a memory card
- Remap certain incoming characters
- Select character set(s)
- Restart the printer
- Select verbosity level
- Select type of error message
- Enable error handling and create customized error messages
- Select method for breaking the printing of a batch of labels
- Change the printer's setup

10. Reading Printer's Status

Provided you have two-way serial communication between printer and host, you can return the printer's status with regard to a number of functions to the host, e.g.:
- Current date and time
- Memory status
- Printhead status and characteristics
- Value of various system counters
- Status of various sensors
- Software and hardware version

11. File-Handling Commands

A number of commands are used to control the printer's memory, for example:

- Read the names of files, fonts or images stored in the printer's memory
- Download binary files
- Download .PCX files
- Remove files
- Remove downloaded images
- Copy files

12. Syntax Descriptions

Many commonly used commands have a shorthand version to minimise the transfer of data. In the explanations of the various command that follow, both the full name and the shorthand version will be shown, separated by a thin vertical line, e.g.:

PRPOS | PP

Upper- and lower-case characters can be used at will in commands. Parameters for commands are shown like this:

<parameter>	*numeric value*
"<parameter>"	*alphanumeric text (enclosed by double quotation marks)*

- Compulsory space characters are indicated by a double-headed arrow (↔).
- Square brackets [ ] indicate optional parameters.
- Thin vertical bars (|) indicate alternatives.
- Always enter parentheses, commas, colons, semicolons, minus signs, quotation marks and period characters exactly as shown.
- Negative values are indicated by leading minus signs (-).

13. File Storage Devices and File Names

Files can be stored in the EasyCoder 301 on one of four devices:

- ***"ROM:"***
  Contains the built-in files necessary for using the printer.
- ***"RAM:"***
  Contains files created by the user.
- ***"CARD1:"***
  A PCMCIA card in the slot nearest the ON/OFF switch, which can be a DOS-formatted card or a specially formatted font cartridge.
- ***"CARD2:"***
  A PCMCIA card in the slot farthest from the ON/OFF switch, which can be a DOS-formatted card or a specially formatted font cartridge.

The files in each device can be listed using the **FILES** command, e.g. **FILES "RAM:"**

In general, ROM: and font cartridges in "CARD1:" or "CARD2:" are read-only devices: you cannot create, copy or delete files on these devices, e.g. the following commands **would cause errors**:

```
FILE&LOAD "ROM:NEWFILE",123
COPY "RAM:OLDFILE","CARD1:NEWFILE"
      (if CARD1: contains a font cartridge)
KILL "CARD2:OLDFILE"
      (if CARD2: contains a font cartridge)
```

In contrast, "RAM:" and DOS-formatted cards in "CARD1:" or "CARD2:" are read/write devices which support all file operations.

When using a Direct Protocol command which has the name of an existing file as a parameter, you can:

- specify the device name before the filename,
  e.g. **KILL "RAM:MYFILE.TXT"**
- omit the device name,
  e.g. **KILL "MYFILE.TXT"**
  in which case, the printer will search each device in turn until it finds a file with this name.

If you use a Direct Protocol command which creates a file, you can:

- specify the device on which the file is to be stored,
  e.g. **FILE&LOAD "CARD1:MYFILE.TXT", 123**
- omit the device name,
  e.g. **FILE&LOAD "MYFILE.TXT", 123**
  in which case the file will be stored on "RAM:".

Label Design

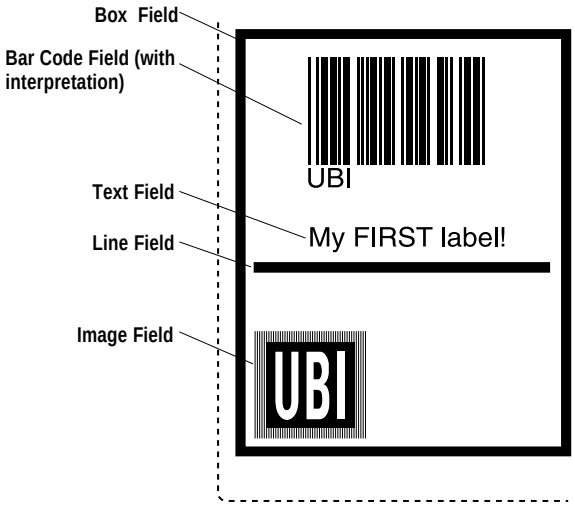
1. Introduction

Field Types

A label layout is made up of a number of fields. There are five different types of field:

- ***Text Field*** A text field consists of a single line of text.
- ***Bar Code Field*** A bar code field consists of a single bar code, with or without a bar code interpretation in human readable characters.
- ***Image Field*** An image field is a picture, drawing, logo or other type of illustration in bitmap format.
- ***Box Field*** A box field is a square or rectangular paper-coloured area surrounded by a black border line. If the border is sufficiently thick, the whole area may appear black.
- ***Line Field*** A line field is a black line that goes either along or across the paper web. A short but thick line can look like a black box.

There are no restrictions, other than the size of the memory, on the number of fields on a single label.



2. General Formatting Commands

Origin

The positioning of all printable objects on the label, i.e. text fields, bar code fields, images, boxes, and lines, uses a common system. The starting point (called the origin) is the point on the paper that corresponds to the innermost active dot on the printhead at the moment when printing is started.

The location of the origin is affected by the following factors:

- Position across the paper web (X-axis):
  The position of the origin in the X-axis is determined by the **XSTART** value in Setup Mode.
- Position along the paper web (Y-axis):
  The position of the origin in the Y-axis is determined by the **STARTADJ** and **STOPADJ** values in Setup Mode and by any **FORMFEED** command executed before the current **PRINTFEED** command or after the preceding **PRINTFEED**.

Coordinates

Starting from the origin, the X-axis runs across the paper web from left to right (as seen when facing the printer) and the Y-axis runs along the paper web from the printhead towards the back of the printer.

Units of Measurement

The unit of measurement is always “dots”, i.e. all measures depend on the density of the printhead. The EasyCoder 301 has an 8 dots/mm printhead, i.e. a dot is $^1/_8$ mm = 0.125 mm = 0.00492" or 4.92 mils.

Insertion Point

The insertion point of any printable object is specified within the coordinate system by means of **PRPOS|PP**. The coordinates must be selected so the field fits completely inside the printable area.

PRPOS | PP <x-coordinate>,<y-coordinate>	
<x-coordinate>	*the distance in dots along the x-axis from the origin to the insertion point*
<y-coordinate>	*the distance in dots along the y-axis from the origin to the insertion point*
Default value:	*PRPOS 0,0*
Reset to default by:	*PRINTFEED|PF*

Example:

```
PP 100, 200 ↵
```

2. General Formatting Commands, cont'd.

Alignment

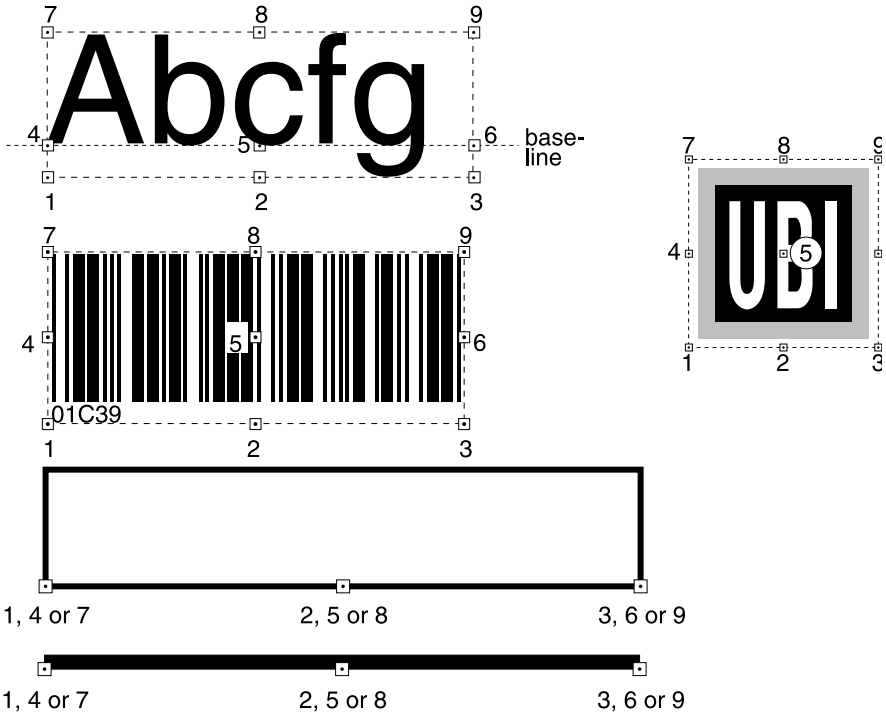
Once the insertion point is specified, you must also decide which part of the object should match the insertion point. For example, a text field forms a rectangle. There are 8 anchor points along the borders and one in the centre, numbered 1–9 as shown in this illustration. The desired anchor point is specified by means of the **ALIGN** command.

ALIGN | AN <anchor point>

<anchor point>	*is a number from 1 – 9*
Default value:	*1*
Reset to default by:	*PRINTFEED|PF*

Example:
AN 1 ↵

ALIGN|AN will place the lower left corner of the text field at the insertion point specified by PRPOS. Refer to the illustration below for detailed information on the anchor points for various types of printable objects.



2. General Formatting Commands, cont'd.

Direction

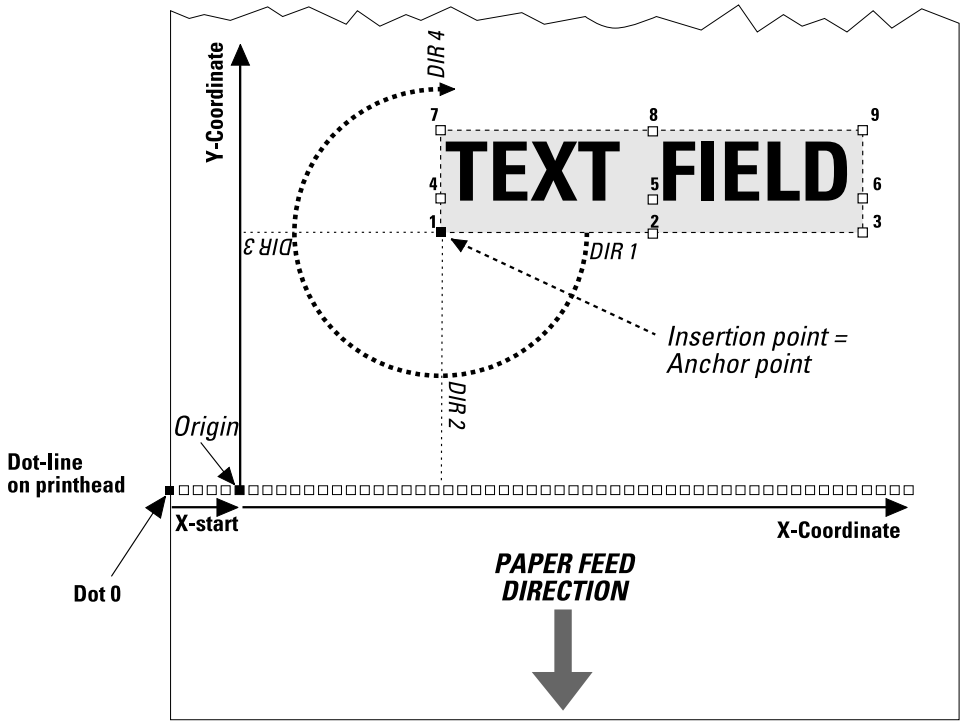
UBI Direct Protocol allows printing in four directions. You can rotate the printable object clockwise around the anchor point/insertion point with a 90° increment (0°, 90°, 180°, or 270°) by means of a **DIR** command:

DIR <direction>	
<direction>	*is a number from 1 – 4 (DIR 1 = 0°; DIR 2 = 90°; DIR 3 = 180°; DIR 4 = 270°)*
Default:	*1*
Reset to default by:	*PRINTFEED|PF*

Example:
DIR 3 ↵

DIR rotates all following printable fields so that they are printed across the paper web and upside down with respect to the way the paper is fed out of the printer.

This illustration summarizes the three general formatting commands **PRPOS|PP, ALIGN|AN** and **DIR**:



3. Text Field

A text field consists of one or more alphanumeric characters on the same line (max 300 characters). UBI Direct Protocol cannot wrap text to a new line, but each line must be specified as a separate text field.

UBI Direct Protocol has been extended to allow characters from any national character set, such as Chinese, Cyrillic and Hebrew, to be included in a text field. In this section we will discuss the commands required to control the format of text fields containing characters from the default character set, Roman 8 (see chapter 11.1). The Direct Protocol commands related to printing characters from other character sets are discussed in chapters 6.11 and 9.2.

In addition to the general formatting commands, **PRPOS, ALIGN** and **DIR** (see chapter 4.2), a text field can contain the following commands:

- Font (typeface)
- Font Size (in points)
- Font Slant (in degrees)
- Magnification
- Normal Image/Inverse Image

Select a Font

The **FONT** command specifies the file name of the scalable font to use and the height and the italic angle with which the characters will be printed.

FONT | FT "<font file name>" [,<font size>,<font slant>]	
<font file name>	*the name of a Speedo or TrueType font file, and must be enclosed by double quotation marks.*
<font size>	*the height of the characters in points (a point is a standard typographic unit, equal to 1/72 inches/0.352 mm).*
<font slant>	*the italic angle of the characters in degrees; a positive value slants the characters clockwise away from the vertical.*
Default:	*"Swiss 721 BT", size 12p, slant 0°*
Reset to default by:	*PRINTFEED|PF*

Before using any of the standard fonts OCR-A BT, OCR-B 10 Pitch BT or Zapf Dingbats BT, you must select a special character set for that font by means of a **NASC** command (see chapters 6.11, 9,2 and 11.1). Before switching back to any other font than those mentioned above, you must specify the original character set again by means of a new **NASC** command.

3. Text Field, cont'd.

The three font parameters can be specified separately using the following commands:

FONT | FT "<font file name>"

FONTSIZE | FS <font size>

FONTSLANT | FL <font slant>

For each parameter, the value specified by these commands will be used in all text fields on the current label, until a new value is specified.

Standard EasyCoder 301 printers contain the following 15 font files:

- Century Schoolbook BT
- Dutch 801 Roman BT
- Dutch 801 Bold BT
- Futura Light BT
- Letter Gothic 12 Pitch BT
- Monospace 821 BT
- Monospace 821 Bold BT
- OCR-A BT *(use NASC "OCR-A.NSC)*
- OCR-B 10 Pitch BT *(use NASC "OCR-B.NSC)*
- Prestige 12 Pitch Bold BT
- Swiss 721 BT (default font)
- Swiss 721 Bold BT
- Swiss 721 Bold Condensed BT
- Zapf Dingbats BT *(use NASC "ZAPF.NSC)*
- Zurich Extra Condensed BT

For examples of the above fonts, see chapter 11.2.

Previous EasyCoder printers contained bitmapped fonts, i.e. fonts scaled with a fixed size and slant[1]. To provide backward compatibility, the EasyCoder 301 provides **FONT** commands to select the equivalent combinations of font file name, size and slant:

"Old" EasyCoders	EasyCoder 301
"SW020BSN"	"Swiss 721 Bold BT",6,0
"SW030RSN"	"Swiss 721 BT",9,0
"SW050RSN"	"Swiss 721 BT",14,0
"SW060BSN"	"Swiss 721 Bold BT",17,0
"SW080BSN"	"Swiss 721 Bold BT",23,0
"SW120BSN"	"Swiss 721 Bold BT",34,0
"MS030RMN"	"Monospace 821 BT",9,0
"MS050RMN"	"Monospace 821 BT",14,0
"MS060BMN"	"Monospace 821 Bold BT",17,0
"OB035RM1"	"OCR-A BT",8,0

[1]/. Since the height of these fonts where specified in dots, not points, the point sizes in the table indicate the nearest value expressed as a whole number (integer) of points and is thus not 100% correct.

3. Text Field, cont'd.

Set Font Magnification

Fonts can be magnified 1 - 4 times independently with regard to height and width by means of a **MAG** command.

MAG <height mag>,<width mag>	
<height mag>	*the magnification factor 1,2,3 or 4 times with regard to height.*
<width mag>	*the magnification factor 1,2,3 or 4 times with regard to width.*
Default:	*1,1*
Reset to default by:	*PRINTFEED|PF*

Specifying both a font height and two magnification parameters allows the user to control both the height and the aspect ratio of printed characters. For example, to specify Swiss 721 Roman at 12 points with double width, use:

```
FONT "Swiss 721 BT",12,0
MAG 1,2
```

To specify Swiss 721 Roman at 12 points with a height:width ratio of 2:3, use:

```
FONT "Swiss 721 BT",6,0
MAG 2,3
```

Select Normal/Inverse Printing

Normally, text is printed in black without any background (**NORIMAGE**). Using **INVIMAGE,** the printing can be inversed so that the paper gives the colour of the characters, whereas the background will be black. The size of the background is decided by the font size and slant. A **NORIMAGE** statement is only needed when changing back from **INVIMAGE** printing.

NORIMAGE | NI	
INVIMAGE | II	
Default:	*NORIMAGE*
Reset to default by:	*PRINTFEED|PF*

3. Text Field, cont'd.

Summary

To create a text field, the following formatting commands must be given (in most cases default values may substitute missing parameters). Input data to the field is explained chapter 4.9, printing commands are explained in chapter 5.2, and the use of national character sets is explained in chapters 6.11 and 9.2.

Purpose	Command	Default	Remarks
X/Y Position	PRPOS **|** PP	0,0	Number of dots
Alignment	ALIGN **|** AN	1	Select AN 1 – 9
Direction	DIR	1	Select DIR 1 – 4
Font	FONT **|** FT	"Swiss 721 BT", 12, 0	
Font Size	FONTSIZE **|** FS	12	Can be added to FONT command
Font Slant	FONTSLANT **|** FL	0	Can be added to FONT command
Magnification	MAG	1,1	Height 1–4, Width 1–4
Appearance	INVIMAGE **|** II	off	White-on-black print
	NORIMAGE **|** NI	on	Black-on-white print

Examples:

```
PP 200,500:AN 7:DIR 2:
FT "Futura Light BT",15,10:MAG 2,2:II ↵

PP 50,300:NASC "OCR-A.NSC":
FT "OCR-A BT":PT "This is OCR-A":
PP 50,50:NASC 1:
FT "Futura Light BT",15,10:
PT "This is Futura Light" ↵
```

4. Bar Code Field

As standard, UBI Direct Protocol supports 37 of the most common bar code symbologies. Each bar code (optionally including its human readable interpretation) makes up a bar code field.

In addition to the general formatting commands, **PRPOS**, **ALIGN** and **DIR** (see chapter 4.2), a bar code field can specify the following properties:

- Bar Code Type
- Height (height of bar pattern)
- Ratio (wide bars/narrow bars)
- Magnification (bar pattern)
- Bar Code Interpretation On/Off
- Bar Code Interpretation Font
- Bar Code Interpretation Font Size
- Bar Code Interpretation Font Slant

Select a Bar Code Type

The type of bar code is specified by a **BARTYPE** command containing an abbreviation of the bar code name. The abbreviation must be entered exactly as listed below.

BARTYPE | BT "<bar code name abbreviation>"

<bar code name...>	*the designation of the bar code type according to the list below; it must be enclosed by double quotation marks.*
Default:	*"INT2OF5"*
Reset to default by:	*PRINTFEED|PF*

Bar Code	**Abbreviation**
Codabar	"CODABAR"
Code 11	"CODE11"
Code 16K	"CODE16K"
Code 39	"CODE39"
Code 39 full ASCII	"CODE39A"
Code 39 w. checksum	"CODE39C"
Code 49	"CODE49"
Code 93	"CODE93"
Code 128	"CODE128"
DUN-14/16	"DUN"
EAN-8	"EAN8"
EAN-13	"EAN13"
EAN-128	"EAN128"
Five-Character Supplemental Code	"ADDON5"
Industrial 2 of 5	"C2OF5IND"
Industrial 2 of 5 w. checksum	"C2OF5INDC"

Note:
Complex bar codes, such as PDF 417 and MaxiCode, must be specified by means of an extended version of the **BARSET** *command, see chapter 9.1.*

4. Bar Code Field, cont'd.

Bar Code	Abbreviation
Interleaved 2 of 5	"INT2OF5"
Interleaved 2 of 5 w. checksum	"INT2OF5C"
Interleaved 2 of 5 A	"I2OF5A"
Matrix 2 of 5	"C2OF5MAT"
Maxicode	"MAXICODE"
MSI (modified Plessey)	"MSI"
PDF 417	"PDF417"
Plessey	"PLESSEY"
Straight 2 of 5	"C2OF5"
Two-Character Supplemental Code	"ADDON2"
UCC-128 Serial Shipping Container Code	"UCC128"
UPC-5 digits Add-On Code	"SCCADDON"
UPC-B	"UPCB"
UPC-A	"UPCA"
UPC-D1	"UPCD1"
UPC-D2	"UPCD2"
UPC-D3	"UPCD3"
UPC-D4	"UPCD4"
UPC-D5	"UPCD5"
UPC-E	"UPCE"
UPC Shipping Container Code	"UPCSCC"

See also chapter 9.1.

Set Bar Code Height

The height of the bars that make up the bar code can be specified by means of the **BARHEIGHT** command.

BARHEIGHT | BH <height>

<height>	*the height of the bars in dots*
Default:	*100*
Reset to default by:	*PRINTFEED|PF*

Set Bar Code Ratio

The ratio between the wide and the narrow bars can be set using a **BARRATIO** command.

BARRATIO | BR <wide bars>,<narrow bars>

<wide bars>	*width of wide bars in dots*
<narrow bars>	*width of narrow bars in dots*
Default:	*3:1*
Reset to default by:	*PRINTFEED|PF*

4. Bar Code Field, cont'd.

Set Bar Code Magnification

The magnification of the bar code and the bar code ratio decides the actual thickness of the bars. For example, if **BARRATIO** is 3:1 and **BARMAG** is 2, then the wide bars will be 6 dots thick (3×2) and the narrow bars will be 2 dots (1×2). Magnification also affects interpretations that are integrated in the code, e.g. EAN and UPC codes.

BARMAG | BM <magnification>	
<magnification>	*the enlargement factor of the bar code pattern.*
Default:	*2*
Reset to default by:	*PRINTFEED|PF*

Bar Codes (combined command)

The **BARSET** command is primarily intended for complex 2-dimensional codes (see chapter 9.1), but can also be used to specify more simple bar codes by means of a single command:

BARSET"<code name>",<ratio wide bars>,<ratio narrow bars>,<magn.>,<height>	
<code name>	*specifies bar code name (default "INT2OF5")*
<ratio wide bars>	*specifies ratio wide/narrow bars (default 3)*
<ratio narrow bars>	*specifies ratio wide/narrow bars (default 1)*
<magnification>	*specifies magnification (default 2)*
<height>	*specifies height of bars in dots (default 100)*
Reset to default by:	*PRINTFEED|PF*

4. Bar Code Field, cont'd.

Select Bar Code Interpretation Font

Most bar codes do not automatically include any bar code interpretation in human readable characters. If you wish to include a bar code interpretation, you can specify the font to be used for the bar code interpretation (human readable).

BARFONT | BF[#<start parameter>,]"<font filename>"[, "<fontsize>"
[,"<fontslant>"[,<vert. offset>[,<height mag> [,<width mag>]]]][ON]

<start parameter>	*specifies the first parameter of the command (default 1)*
<font filename>	*the file name of the font to be used (default "Swiss 721 BT")*
<font size>	*the height of the font in points (default 12 pt)*
<font slant>	*the slant of the font in degrees clockwise (default 0°)*
<vert. offset>	*distance between bar code and interpretation in dots (default 6 dots)*
<height mag>	*the magnification with regard to height (default 1)*
<width mag>	*the magnification with regard to width (default 1)*
<ON>	*enables the printing of bar code interpretation (default OFF)*
Reset to default by:	*PRINTFEED|PF*

The bar code font size and slant can be specified separately using the following commands:

BARFONTSIZE | BFS <font size>

BARFONTSLANT | BFL <font slant>

As with text fields, human readable bar code interpretation fonts can contain characters from any national character set.

However, before using any of the standard fonts OCR-A BT, OCR-B 10 Pitch BT or Zapf Dingbats BT, you must select a special character set for that font by means of a **NASC** command (see chapters 6.11, 9,2 and 11.1). Before switching back to any other font than those mentioned above, you must specify the original character set again by means of a new **NASC** command.

4. Bar Code Field, cont'd.

Bar code interpretation printing can be enabled or disabled by means of these commands:

BARFONT | BF ON

BARFONT | BF OFF

By default, bar code interpretation is disabled.

Summary

To create a bar code field, the following formatting commands must be given (in most cases default values may substitute missing parameters). Input data and printing commands are explained in chapters 4.9 and 5.2 respectively.

Purpose	**Command**	**Default**	**Remarks**
X/Y Position	PRPOS | PP)	0,0	Number of dots
Alignment	ALIGN | AN	1	Select ALIGN 1 – 9
Direction	DIR	1	Select DIR 1 – 4
Bar Code Select	BARSET	see above	Can be replaced by BT, BH, BR, BM
Human Readable	BF ON	Off	Can be omitted

Examples:

```
PP 100,100:AN 7:DIR 4:BARSET "CODE39",2,1,3,120:
BF "Swiss 721 Bold BT"20,0,5,1,1 ON ↵

PP 100,100:AN 7:DIR 4:BARSET "CODE39",2,1,3,120:
NASC "OCR-A.NSC":
BF "OCR-A BT"20,0,5,1,1 ON:NASC 1 ↵
```

5. Image Field

An image field contains a picture or logo in .PCX format. The image can either be stored in the printer's RAM or in a memory card. It can also be downloaded from the host computer.

You cannot store images in the printer's ROM, but it does contain two standard images: "ROM:UBI.1", which is the UBI logo for DIR 1 and 3, and "ROM:UBI.2", which is the UBI logo for DIR 2 and 4.

In addition to the general formatting commands, **PRPOS**, **ALIGN** and **DIR**, an image field can contain the following commands:

- Name
- Magnification
- Normal/Inverse printing

Select an Image:

An image is selected by the full name under which it is stored in the printer's memory, either as plain text or as a variable (also see chapter 4.9).

PRIMAGE | PM "<image name>"

<image name>	*the full name and storage location of the image including extension, enclosed by double quotation marks*
Default:	*None*

The UBI image name convention includes an extension (.1 or .2), where .1 indicates that the image is intended for print directions 1 and 3, whereas .2 indicates print directions 2 and 4 (see chapter 4.2). For example, the image "ROM:UBI.1" is suited for directions 1 and 3.

Set Image Magnification:

Images can be magnified 1 – 4 times independently with regard to height and width by means of a **MAG** command.

MAG <height mag>,<width mag>

<height mag>	*the magnification factor 1, 2, 3 , or 4 times with regard to height.*
<width mag>	*the magnification factor 1, 2, 3 , or 4 times with regard to width.*
Default:	*1,1*

Reset to default by: *PRINTFEED|PF*

To obtain the best printout quality, select a larger image rather than magnifying a smaller one.

5. Image Field, cont'd.

Select Normal/Inverse Printing

Normally, an image is printed in black and white, just as it was created. Using **INVIMAGE**, the black and white can be inverted. The size of the background is decided by the actual size of the image including "invisible" background. A **NORIMAGE** statement is only needed when changing back from **INVIMAGE** printing.

NORIMAGE | NI

INVIMAGE | II

Default: *NORIMAGE*
Reset to default by: *PRINTFEED|PF*

Summary

To create an image field, the following formatting commands must be given (in most cases default values may substitute missing parameters). Image selection by means of variable input, and printing commands are explained in chapters 4.9 and 5.2 respectively.

Purpose	Command	Default	Remarks
X/Y Position	PRPOS **|** PP	0,0	Number of dots
Alignment	ALIGN **|** AN	1	Select ALIGN 1 – 9
Direction	DIR	1	Select DIR 1 – 4
Magnification	MAG	1,1	Height 1 – 4, Width 1– 4
Appearance	INVIMAGE **|** II	off	Black and white parts inverted
	NORIMAGE **|** NI	on	Normal (revokes INVIMAGE)
Image name	PRIMAGE **|** PM	n.a.	Full name including extension

Example:

```
PP 200,500:AN 3:DIR 3:MAG 2,2:II:PM "ROM:UBI.1"↵
```

6. Box Field

A box is a hollow square or rectangle that can be rotated with an increment of 90° according to the print direction. If the line thickness is sufficiently large, the box will appear to be filled (another method is to print an extremely thick short line).

In addition to the general formatting commands, **PRPOS**, **ALIGN** and **DIR**, a box field is specified by a single command.

Set Size of the Box

The size of the box is specified with regard to height, width and line weight (thickness).

PRBOX | PX <height>,<width>,<line thickness>

<height>	*the height of the box in dots perpendicular to the selected direction.*
<width>	*the length of the box in dots along the selected direction.*
<line thickness>	*the line weight in dots.*
Default:	*None*

Summary

To create a box field, the following formatting commands must be given (in most cases, default values may substitute missing parameters). Printing commands are explained in chapter 5.2.

Purpose	Command	Default	Remarks
X/Y Position	PRPOS **|** PP	0,0	Number of dots
Alignment	ALIGN **|** AN	1	Select ALIGN 1 – 9
Direction	DIR	1	Select DIR 1 – 4
Box size	PRBOX **|** PX	n.a.	Height, width and line thickness in dots

Example:

```
PP 250,250:AN 1:DIR 3:PX 200,200,10 ↵
```

7. Line Field

A line can be printed in right angles along or across the paper according to the print direction.

In addition to the general formatting commands, **PRPOS**, **ALIGN** and **DIR**, a line field is specified by a single command.

Set Size of the Line

The size of the line is specified with regard to length and line weight (thickness).

PRLINE | PL <length>,<line thickness>

<length>	*the length of the line in dots along the selected direction.*
<line thickness>	*the line weight in dots.*
Default:	*No*

Summary

To create a line field, the following formatting commands must be given (in most cases default values may substitute missing parameters). Printing commands are explained in chapter 5.2.

Purpose	Command	Default	Remarks
X/Y Position	PRPOS **|** PP	0,0	Number of dots
Alignment	ALIGN **|** AN	1	Select ALIGN 1 – 9
Direction	DIR	1	Select DIR 1 – 4
Line size	PRLINE **|** PL	n.a.	Length and line thickness in dots

Example:

```
PP 100,100:AN 1:DIR 4:PL 200,10 ↵
```

8. Layout Commands

Start Layout Recorder

The **LAYOUT INPUT** command clears the printer's working memory, starts the layout recorder and allows you to assign a name to the layout.

LAYOUT INPUT "<layout name>"

<layout name>	*up to 30 characters enclosed by double quotation marks.*

Example:
LAYOUT INPUT "Shipping Label" ↵

Assign Input Variables to Fields

The layout may consist of both fixed fields and fields for variable information. In the layout, you must give a reference to each text, bar code or image field intended to receive variable data, so that the input data can be inserted into the correct field. The variable **VAR<n>$** is used to indicate variable input, where **<n>** specifies the number of the field. The first variable input data block will be entered into the field containing **VAR1$**, the second block with **VAR2$,** etc. Fixed data and variable data can be combined in the same field. See also chapter 4.9.

VAR<n>**$**

<n>	*the number of the field. There is no practical limit to the number of fields.*

Examples:

PT VAR1$ ↵	*(text field with variable input)*
PT "Price: ";VAR2$ ↵	*(text field with fixed & variable input)*
PB VAR3$ ↵	*(bar code field with variable input)*
PM VAR4$ ↵	*(image field with variable input)*

Save the Layout

After completing the layout, save it in the printer's RAM memory, turn off the layout recorder and clear the printer's working memory by means of a **LAYOUT END** command. The layout can then be copied or killed like any other file, see chapter 3.1.

LAYOUT END

Example:
LAYOUT END ↵

9. Printable Data Commands

Select a Layout

Before any variable data can be transmitted to a preprogrammed layout, the layout must be selected by means of a **LAYOUT RUN** command.

LAYOUT RUN "<layout name>"

<layout name>	*the name given to the layout in the LAYOUT INPUT command and must be enclosed by double quotation marks.*

Example:

```
LAYOUT RUN "Shipping Label" ↵
```

Transmit Variable Data to a Layout

After having selecting a layout using a **LAYOUT RUN** command, you can transmit the variable data to their respective layout fields:

- The transmission starts with a start-of-text separator.
- Then comes a block of data to the field containing **VAR1$**.
- A field separator separates the blocks of data.
- Next block goes to the field containing **VAR2$**.
- A field separator separates the blocks of data and so on.
- The last block must also end with a field separator.
- The end of transmission is indicated by an End-of text separator.

By default, the following separators should be used:

- Start separator: **STX** (ASCII 02 dec)
- Field separator: **CR** (ASCII 13 dec)
- End separator: **EOT** (ASCII 04 dec)

All separators can be changed by means of **INPUT ON/OFF** and a **FORMAT INPUT** command, see chapter 6.5.

<STX> <Input data to VAR1$> **<CR>** <Input data to VAR2$> **<CR>.....<EOT>**

Input data **must not** be enclosed by double quotation marks.

Example:

```
LAYOUT RUN "Shipping Label" ↵
<STX> Abcdefg <CR>123456789 <CR> <EOT>
```

INPUT ON *is the default, so there is no need to issue the command before using* **LAYOUT RUN**.

9. Printable Data Commands, cont'd.

Input Data to Text Fields

Input data is inserted into a text field using a **PRTXT** command. You can add various types of data to a text field:

- Plain text, by typing e.g. "Abcdefgh" or "012345"
- Variable input data, using variables e.g.**VAR1$**
- Counter values, e.g. **CNT1$**
- Current date, using **DATE$** or **DATE$("F")**
- Current time, using **TIME$** or **TIME$("F")**
- The weekday of the current or specified date, using **WEEKDAY$**
- The number of the current or specified date, using **WEEKNUMBER**
- An offset date, using **DATEADD$** or **DATEADD$("F")**
- An offset time, using **TIMEADD$** or **TIMEADD$("F")**

You can combine different types of data in a single **PRTXT** command. The different parts are separated by semicolons (;). Note that plain text must be enclosed by double quotation marks.

PRTXT | PT "<input data>"[;"<input data>"...]

Examples:

```
PT "Price: $1.99" ↵
PT "Price: ";VAR1$;" per dozen" ↵
PT "Box No. ";CNT15$;" Packed: ";DATE$ ("F") ↵
PT WEEKDAY$(DATE$);", ";DATE$("F") ↵
PT "Week Number ";WEEKNUMBER(DATE$) ↵
PT "Expiry date: ";DATEADD$ (30,"F") ↵
```

Input Data to Bar Code Fields

Input data is inserted into a bar code field using a **PRBAR** command. You can add the same types of data to a bar code field as to a text field, so long as the type of data (e.g. numeric/alphanumeric) and the number of characters, etc. comply with the bar code specification.

You can combine different types of data in a single **PRBAR** command. The different parts are separated by semicolons (;). Note that alphanumeric input must be enclosed by double quotation marks.

PRBAR | PB <input data>[;<input data>...]

Examples:

```
PB "71543";VAR5$ ↵
PB "UBI" ↵
PB DATE$;TIME$ ↵
```

9. Printable Data Commands, cont'd.

Input Data to Image Fields

An image can be selected either by name in plain text (e.g. "ROM:UBI.1"), or in the form of a variable (e.g. **VAR1$**), (see also chapter 4.5). Note that plain text input must be enclosed by double quotation marks.

PRIMAGE | PM <image name>

Examples:
PM VAR5$ ↵
PM "LOGO.2" ↵

Input Data from Counters

By means of a **COUNT&** command, various counters can be created (see chapter 6.6). You can read the present value of a counter and use it as input data by including a reference to the counter in the **PRTXT** or **PRBAR** commands in the form of a variable.

CNT <Counter No.>**$**

<Counter No.> *the number assigned to the counter in the COUNT& command.*

Example:
PT "Label number: ";CNT1$ ↵
PB CNT2$ ↵

Input Data from the Printer's Clock/Calendar

The printer's clock/calendar can be used to provide input data for text and bar code fields by including any of the following in the **PRTXT** or **PRBAR** commands:

- **DATE$**
- **DATE$("F")**
- **TIME$**
- **TIME$("F")**
- **WEEKDAY$**
- **WEEKNUMBER**
- **DATEADD$**
- **TIMEADD$**

A real-time clock circuit is an optional extra on the EasyCoder 301.

If the optional RTC is not fitted, the date and time will be lost when the printer is switched off. Use the **DATE$** *and* **TIME$** *commands to reset the clock and calendar.*

9. Printable Data Commands, cont'd.

DATE$

*Returns the current date according to the printer's calendar in the standard format **YYMMDD**, where **YY** is the last two digits of the year, **MM** is the number of the month (01–12) and **DD** is the number of the day (01–31).*

Example:
PT DATE$ ↵

DATE$ ("F")

*Returns the current date according to the printer's calendar in the format specified by **FORMAT DATE$** (see chapter 6.4).*

Example:
PT DATE$("F") ↵

TIME$

*Returns the current time according to the printer's clock in the standard date format **HHMMSS**, where **HH** is the hour (00–24), **MM** is the minute (00–59) and **SS** is the second (00–59).*

Example:
PT TIME$ ↵

TIME$ ("F")

*Returns the current time according to the printer's clock in the format specified by **FORMAT TIME$** (see chapter 6.4).*

Example:
PT TIME$("F") ↵

WEEKDAY$ ("<date>")

*Returns the name of the weekday in plain text according to **NAME WEEKDAY$** (see chapter 6.4) from a given date or the current date.*
*<date> can be specified in the standard format **"YYMMDD"** or by a **DATE$** command.*

Examples:
PT WEEKDAY$("970601") ↵
PT WEEKDAY$(DATE$) ↵

9. Printable Data Commands, cont'd.

WEEKNUMBER ("<date>")

Returns the weeknumber from a given date or the current date.

<date> *can be specified in the standard format* ***"YYMMDD"*** *or by a* ***DATE$*** *command.*

Examples:

PT WEEKNUMBER("970601") ↵
PT WEEKNUMBER(DATE$) ↵

DATEADD$ (["<original date>",]<number of days>[,"F"])

Adds or subtracts a certain number of days to the current date or optionally to a specified date.

<original date> *optional; it is entered in the standard date format* ***"YYMMDD"****. Note that the original date must be enclosed by double quotation marks.*

<number of days> *the number of days to be added to or subtracted from the current date or, optionally, the date specified by <original date>. In case of subtraction, the <number of days> should be preceded by a minus sign (-).*

<"F"> *an optional flag specifying that the result should be returned in the format specified by* ***FORMAT DATE$*** *instead of the standard format* ***YYMMDD****.*

Example:

PT DATEADD$("970601",-15,"F") ↵

9. Printable Data Commands, cont'd.

TIMEADD$ (["<original time>",]<number of sec's>[,"F"]**)**

Adds or subtracts a certain number of seconds to the current time or optionally to a specified moment of time.

<original time>	*optional; it is entered in the standard date format* ***"HHMMSS"****. Note that the original time must be enclosed by double quotation marks.*
<number of secs>	*the number of seconds to be added to or subtracted from the current time or, optionally, the moment of time specified by <original time>. In case of subtraction, the <number of secs> should be preceded by a minus sign (-).*
<"F">	*an optional flag specifying that the result should be returned in the format specified by* ***FORMAT TIME$*** *instead of the standard format* ***HHMMSS****.*

Example:

```
PT TIMEADD$("123026",100,"F") ↵
```

Feeding and Printing Commands

1. Paper Feed

In order to provide maximum flexibility, there is a number of commands for controlling the paper feed:

- **CLEANFEED**
- **FORMFEED**
- **TESTFEED**

CLEANFEED <feed length in dots>

Runs the printer's paper feed mechanism in order to facilitate cleaning of the print roller.

FORMFEED | FF [<feed length in dots>]

Feeds out a blank label or optionally feeds out (+) or pulls back (-) a certain amount of paper without printing.

TESTFEED

Feeds out a blank label while adjusting the label stop/black mark sensor.

The paper is fed past the printhead by a rubber-coated roller driven by a stepper motor. The software can control the stepper motor with an accuracy of one dot in either direction. The movement of the paper is detected by the label stop sensor (**LSS**) or black mark sensor (**BMS**), except when various types of paper strip are used.

The printer's setup with regard to Detection, Media Size, Length and Media Type determines how the paper feed will work. There are five different types of Media Type options (also see the EasyCoder 301 Installation and Operation Manual):

- Label (w gaps)
- Ticket (w mark)
- Ticket (w gaps)
- Fix length strip
- Var length strip

When a **FORMFEED**, **TESTFEED** or **PRINTFEED** command is executed, the photoelectrical label stop sensor (LSS) detects the forward edge of each new label or the forward edge of each detection gap, and the black mark sensor (BMS) detects the forward edge of each black mark, as the paper is fed past the sensor.

1. Paper Feed, cont'd.

By performing a **TESTFEED** operation after loading a new supply of paper, the software is able to measure the distance between, for example, the forward edges of two consecutive labels, thereby determining the label length, so that it can adjust the paper feed accordingly. The same principle applies to tickets or tags with detection gaps and to tickets with black marks.

In the case of paper strip, the LSS will only detect an out-of-paper condition; the amount of paper feed is decided in two different ways:

- ***Fixed length strip***
  The amount of paper fed for each **FORMFEED**, **TESTFEED** and **PRINTFEED** operation is decided by the Media Size: Length setup.
- ***Variable length strip***
  After executing a **PRINTFEED**, the printer will stop feeding as soon as printing stops. Note that a blank space character or "white" part of an image is also regarded as a printable object. The amount of paper fed by **FORMFEED** is decided by the Media Size: Length setup.

The Detection setup allows you to perform two global adjustments to the paper feed described above:

- Start Adjust
- Stop Adjust

By default, both these parameters are set to 0, which allows for proper tear-off operation when there is no requirement of printing immediately at the forward edge of the label (equivalent).

- ***Start Adjust*** decides how much paper will be fed out or pulled back before the **FORMFEED**, **TESTFEED** or **PRINTFEED** is executed. There is a small distance between the dispenser shaft or tear off edge and the printhead. Thus, if you want to start printing directly at the forward edge of the label, you must pull back the paper before printing by means of a negative start adjust value.
- ***Stop Adjust*** decides how much more or less paper will be fed out after the **FORMFEED**, **TESTFEED** or **PRINTFEED** is executed.

Recommended Adjustments:

Dispensing:

Start Adjust :	*-88*
Stop Adjust:	*-48*

Tear Off:

Start Adjust:	*-136*
Stop Adjust:	*0*

Strip:

Start Adjust:	*-136*
Stop Adjust:	*-60*

1. Paper Feed, cont'd.

If the **FORMFEED** command is issued without any specification of the feed length, a complete blank label (or the equivalent) will be fed out. By specifying a positive or negative value in the **FORMFEED** command, the global Start Adjust and Stop Adjust setup can be substituted or supplemented for individual layouts or labels.

It is important whether the **FORMFEED** command is executed before or after the **PRINTFEED** command:

- **FORMFEED** before **PRINTFEED** corresponds to Start Adjust.
- **FORMFEED** after **PRINTFEED** corresponds to Stop Adjust.

The relationship between paper and printhead when printing starts decides all positioning along the Y-axis, i.e. along the paper web.

2. Label Printing

Printing Commands:

When a **PRINTFEED** command is issued, the software processes all previously entered text fields, bar code fields, image fields, box fields and line fields into a bitmap pattern. The bitmap pattern controls the heating of the printhead dots as the paper is fed past them. Each **PRINTFEED** command produces one single copy **or**, optionally, a batch of labels, tickets, tags or pieces of strip.

PRINTFEED | PF [<batch size>]

<batch size>	*specifies the number of copies to be printed.*
Default:	*1*

The execution of a **PRINTFEED** command resets the following commands to their respective default values:

ALIGN	**BARFONT**	**BARFONTSIZE**
BARFONTSLANT	**BARFONT ON/OFF**	**BARFONTD**
BARFONTDSIZE	**BARFONTDSLANT**	**BARHEIGHT**
BARMAG	**BARRATIO**	**BARSET**
BARTYPE	**DIR**	**FONT**
FONTSIZE	**FONTSLANT**	**FONTD**
FONTDSIZE	**FONTDSLANT**	**INVIMAGE**
MAG	**PRPOS**	

This reset only affects commands executed after **PRINTFEED**, not commands that have already been executed, which makes batch printing possible.

3. Batch Printing

General Information:

The term "Batch Printing" means the process of printing several labels. The labels may either be exact copies or different from each other.

For batch printing, the most critical factor is the time required to process the print image into a bitmap and store it in the image buffer. The image buffer compensates for differences between processing time and printing time. If the bitmap is too large to be stored in its entirety in the image buffer, it will be divided into a number of segments across the paper feed direction which will be processed one after the other and downloaded to the image buffer. As the buffer is emptied by printing, a new segment can be processed and downloaded.

When printing a batch of labels from a predefined layout, the layout is processed before each new copy is printed, in order to allow counter, date and time values to be updated. When a layout is not used, all copies will contain exactly the same information.

When using batch printing, take the following into consideration:

- Using command abbreviations, e.g. **PF** instead of **PRINTFEED** improves the performance.
- The more processing a label requires, the slower it will be printed.
- In case of small differences between labels, make use of **CLL** and **FIELDNO** commands (see below) and edit the layout so that variable data are processed last.
- If you are printing scaled TrueType fonts and the printer is working very slowly, use the **PRESCALE** command to create bitmapped versions of the scaled fonts for quicker access (see chapter 9.3).
- Switching verbosity off with **SYSVAR(18)** may improve printing speed.

If you have any problems, e.g. the printer stops between labels, lower the print speed, or make the print image easier to process.

You can interrupt batch printing by issuing a break command as described in chapter 6.16.

3. Batch Printing, cont'd.

Clearing the Image Buffer:

The image buffer stores the bitmap pattern of the label between processing and printing. The image buffer can be cleared partially or completely by means of a **CLL** command.

CLL [<field>%]	
<field>	*the same alphanumeric designator as in the corresponding **FIELDNO** command, followed by a mandatory **%** sign.*
CLL	*clears the image buffer completely*
CLL <field>%	*clears the image buffer from the corresponding **FIELDNO** command to the end of the label*

Complete clearing is useful to avoid printing a faulty label after certain errors have occurred.

Partial clearing is used in batch printing when only part of the label should be modified between the copies. In this case, the **CLL** statement must include a reference to a field, specified by a **FIELDNO** command. When a **CLL** command is issued, the image buffer will be cleared from the specified field to the end of the label.

<field>% = FIELDNO	
<field>	*the same alphanumeric designator as in the corresponding **CLL** command followed by a mandatory % sign.*

Example:

```
FT "Swiss 721 Bold BT" ↵
MAG 2,2 ↵
PP 100,300 ↵
PT "MONTH:" ↵              (kept in the image buffer for re-use)
PP 100,200 ↵
A%=FIELDNO ↵
PT "JANUARY":PF ↵          (cleared from the buffer after printing)
CLL A% ↵
PP 100,200 ↵
PT "FEBRUARY":PF ↵         (cleared from the buffer after printing)
CLL A% ↵
PP 100,200 ↵
PT "MARCH":PF ↵            (cleared from the buffer after printing)
CLL A% ↵
```

Setting Up the Printer

1. Layout Mode/ Immediate Mode

The EasyCoder 301 Printer can operate in Immediate Mode or Layout Mode. In Immediate Mode, layouts and variable data fields are disabled. In Layout Mode, layouts and variable data fields are enabled (see chapter 4.8). The printer default is Layout Mode.

INPUT ON

INPUT ON sets **SYSVAR(18)** to 0 and enables the use of layouts and variable data. (Default)

INPUT OFF

INPUT OFF restores **SYSVAR(18)** to its value before the **INPUT ON** command was received and disables the use of layouts and variable data.

2 IN/OUT Channel Setup

The printer will receive and transmit data on the standard serial communication channel "uart1:". If you use parallel communication, two-way communication will be lost and the printer will not be able to echo any data back to the host or return any other data. It is possible to select different channels manually as standard IN and standard OUT channels, using the **SETSTDIO** command.

SETSTDIO <IN channel>, <OUT channel>

<IN channel> and <OUT channel> can be individually specified as:

0	*automatic port switching (Default)*
1	*"uart1:" serial communication channel*
4	*"centronics:" parallel communication channel (IN channel only)*

Example:

```
SETSTDIO 0,1 ↵
```

3. Date and Time

The printer's CPU board is, as standard, provided with an internal clock/calendar without battery backup, i.e. the setting will be lost when the printer is turned off.

The CPU board may optionally be fitted with a real-time clock circuit (RTC). The RTC is battery backed-up and will keep running even when the printer is turned off.

If no RTC is installed and you try the read the date or time before the internal clock has been set, an error will occur (error 1010 “Hardware Error”). Once either time or date has been set, the internal clock will work until next power off or reboot. If only time has been set, by default the current date will be Jan 01 1980 and if only date has been set, by default the clock will start running at 00:00:00.

The following commands are used to set the clock/calendar:

DATE$ = "<YYMMDD>"

<YYMMDD>	*the current date in the standard format, where* *YY = Year, last two digits* *MM = Month, two digits* *DD = Day, two digits.*

Note that the input data must be enclosed by double quotation marks.

Example:
DATE$ = "970601" ↵

TIME$ = "<HHMMSS>"

<HHMMSS>	*the current time in the standard format, where* *HH = Hour (00–24)* *MM = Minute (00–59)* *SS = Second (00–59)*

Note that the input data must be enclosed by double quotation marks.

Example:
TIME$ = "131548" ↵

4. Date and Time Format

The formats for printing dates and time in connection with the commands **DATE$("F")**, **DATEADD$("F")**, **TIME$("F")** and **TIMEADD$("F")** (see chapter 6.5) can be specified by the commands **FORMAT DATE$** and **FORMAT TIME$**. With both these commands, you should enter characters representing the various types of information. The order and number of the characters decides the format. You can also include separating characters like periods, slashes, colons, etc. Note that the input string must be enclosed by double quotation marks.

FORMAT DATE$ "<string>"

Y	*Year*
M	*Month*
D	*Day*
Default:	*YYMMDD*

Examples:

FORMAT DATE$ "YYYY.MM.DD" ↵ *gives e.g.* ***1997.06.01***

FORMAT DATE$ "DD/MM/YY" ↵ *gives e.g.* ***01/06/95***

FORMAT TIME$ "<string>"

H	*Hour in 24-hour cycle (one digit per H; right-justified)*
h	*Hour in 12-hour cycle (one digit per h; right-justified)*
M	*Minute (one digit per M; right-justified)*
S	*Second (one digit per S; right-justified)*
P	*AM/PM (uppercase) in 12-hour cycle (one character per P; left-justified)*
p	*am/pm (lowercase) in 12-hour cycle (one character per p; left-justified)*
Default:	*HHMMSS*

Examples:

FORMAT TIME$ "HH:MM:SS" ↵ *gives e.g.* ***14:15:37***

FORMAT TIME$ "HH.MM" ↵ *gives e.g.* ***14.15***

FORMAT TIME$ "hh.MM.SS p" ↵ *gives e.g.* ***2.15.37 p***

FORMAT TIME$ "hh.MM PP" ↵ *gives e.g.* ***2.15 PM***

The date and time formats as well as the names of months and weekdays are not saved in the printer's battery backed-up memory, but must be transmitted to the printer after each power-up.

4. Date and Time Format, cont'd.

In many cases,you may prefer to have the names of months and weekdays printed in plain text rather than as a number. There are two commands which allow you to assign names in any language to months and weekdays:

NAME DATE$<No. of month>,"<name of month>"

<No. of month>	*1–12.*
<name of month>	*the desired name enclosed by double quotation marks.*

The name of the month will be printed according to the format specified by ***FORMAT DATE$*** *and will be left justified.*

NAME WEEKDAY$<No. of weekday>,"<name of weekday>"

<No. of weekday>	*1 (Monday) – 7 (Sunday).*
<name of weekday>	*the desired name enclosed by double quotation marks.*
Default:	*Full English names, e.g. Monday.*

5. Separators

When transmitting variable input data to a predefined layout, the string must contain certain separating characters. By default, you should use **<STX>** as start-of-text separator, **<CR>** as field separator and **<EOT>** as end-of-text separator(see chapter 6.5).

However, the **FORMAT INPUT** command allows you to select other characters as separators if the default separators for some reason cannot be produced or will interfere with the host computer system. Simply insert the desired separator characters (enclosed by double quotation marks) into the **FORMAT INPUT** command. Avoid using characters like **XON/XOFF** or **ENQ/ACK**, which may interfere with the communication between printer and host.

You must first leave Layout Mode by means of an **INPUT OFF** command, change the separators using a **FORMAT INPUT** command and then enter Layout Mode again by means of an **INPUT ON** command[1].

FORMAT INPUT "<start separator>","<end separator>","<field separator>"

Example:

```
INPUT OFF ↵
FORMAT INPUT "#","&",CHR$(13) ↵
INPUT ON ↵
```

[1]. If you transmit a **FORMAT INPUT** *command containing separators identical to any of the separators already stored in the printer's memory, an error will occur unless you first enter Immediate Mode.*

The separators are not saved in the printer's battery backed-up memory, but must be transmitted after each power-up.

6. Counters

You can create counters for use in text and bar code fields (see chapters 4.3 and 4.4). The counters are global, i.e. the same counter can be used in many different labels and layouts, but will be incremented/decremented on any **PRINTFEED** operation, regardless of label. Thus, if you want to use a counter for a specific layout only, you must not use it in any other layout.

Each counter is designated by means of a number. Alpha counters are restricted to A–Z; numeric counters have no practical limit. The type of counter is decided by the type of start, stop or reset value.

There are various commands for creating a counter and specifying its characteristics:

- Start Value
- Number of Digits
- Number of Copies Before Update
- Incrementation/Decrementation
- Stop Value
- Restart Value

Note that input data must be enclosed by double quotation marks.

Start Value

COUNT& "START", <counter number>,"<start value>"

<start value>	*is the first value to be printed (positive or negative). Negative values are indicated by a leading minus sign (-).*
Default:	*1 or A*

Number of Digits

COUNT& "WIDTH", <counter number>,"<number of digits>"

<number of digits>	*adds leading zero characters up to the specified number of digits. Must only be used in numeric counters.*
Default:	*1*

Number of Copies Before Update

COUNT& "COPY", <counter number>,"<number of copies>"

<number of copies>	*sets the quantity of copies to be printed before the counter is incremented or decremented.*
Default:	*1*

6. Counters, cont'd.

Note that counters are not saved in the printer's battery backed-up memory, but must be transmitted to the printer after each power-up.

Incrementation/Decrementation

COUNT& "INC", <counter number>,"<incr. value>|<decr. value>"	
<incr. value>	*sets the value by which the counter should be incremented.*
<decr. value>	*sets the value by which the counter should be decremented. Decrementation is indicated by a leading minus sign (-).*
Default:	*1*

Stop Value

COUNT& "STOP", <counter number>,"<stop value>"	
<stop value>	*sets the value after which the counter should start all over again at the RESTART value.*
Default:	*2, 147, 483, 647 or Z*

Restart Value

COUNT& "RESTART", <counter number>,"<restart value>"	
<restart value>	*sets the value at which the counter should start all over again after having exceeded the STOP value.*
Default:	*1 or A*

7. Label-Taken Sensor

All printers are fitted with a label-taken sensor (LTS) which detects if there is a label left in the printer's outfeed slot and holds up printing until the label has been removed. This facility is especially useful for batch printing.

You can enable or disable this function by means of the following command. By default, LTS is disabled.

LTS& ON | OFF

8. Feed Key

Feed on the EasyCoder 301 is used to initiate a **PRINTFEED** operation.

You can enable or disable this function by means of the following command. By default, the Print key is disabled.

PRINT KEY ON | OFF

9. Memory Formatting

By formatting the printer's memory, you will erase all files stored in the RAM memory. You can also format a RAM-type memory card inserted in the printer's PCMCIA slot to MS-DOS format. Be careful not to use this command unintentionally!

FORMAT "<device>"[,<No. of entries>[,<No. of bytes>]]

<device>	*"RAM:", "CARD1:" or "CARD2:"*
<No. of entries>	*used for "CARD1:" and "CARD2:"; it specifies the number of entries in the root directory (Default 208).*
<No. of bytes>	*used for "CARD1:" and "CARD2:"; it specifies the number of bytes per sector (Default 512 bps).*

10. Input Data Preprocess

All input data to the printer arrives in binary form via the standard IN channel (default "uart1:"). Characters are transmitted in ASCII format, which will be preprocessed by the printer's software according to any **MAP** commands.

The **MAP** command is used to modify a character set, or to filter out undesired characters on a specified communication channel by mapping them as Null (ASCII 0 dec).

If no character set meets your requirements completely (see **NASC** below), select the set that comes closest and modify it using **MAP** commands. Mapping will be saved even during power-up or reboot.

MAP [<device>],<original ASCII value>, <desired ASCII value>

<device>	*either 1 = Serial channel "uart1:" or 4 = Parallel channel "centronics:"*
<orig. ASCII value>	*the value of the character according to the selected character set (see NASC below).*
<des. ASCII value>	*the new ASCII value you want to assign to the character.*

Example:
You want to use the Roman 8 character set and 7-bit communication. However, you need to print £ characters which are not included in the 7-bit part of the Roman 8 character set (see chapter 3.2). However, you do not need the $ character. Then remap the £ character (ASCII 187 dec.) to the value of the $ character (ASCII 36 dec.).

MAP 36,187 ↵

11. Character Set Selection

The **NASC** command is used to select a character set that decides how the various characters will be printed. This command makes it possible to adapt the printer to various national standards. By default, characters will be printed according to the Roman 8 character set. Most resident character sets are illustrated in chapter 11.1. For information on how to create user-defined character sets, refer to chapter 9.2.

NASC <character set number> | <"file name">

<character set no.>	*one of the following numbers:*
	1: Roman 8 (default)
	33: French
	34: Spanish
	39: Italian
	44: English
	46: Swedish
	47: Norwegian
	49: German
	81: Japanese Latin (romají)
	240: Microsoft Hebrew [not documented]
	241: Microsoft Arabic [not documented]
	351: Portuguese
	850: MS-DOS Latin 1
	851: MS-DOS Greek 1
	852: MS-DOS Latin 2
	855: MS-DOS Cyrillic
	857: MS-DOS Turkish
	862: MS-DOS Hebrew [not documented]
	1250: Windows Latin 2 (Cental Europe)
	1251: Windows Cyrillic (Slavic)
	1252: Windows Latin 1 (ANSI)
	1253: Windows Greek
	1254: Windows Latin 5 (Turkish)
	1255: Windows Hebrew [not documented]
	1256: Windows Arabic [not documented]
	1257: Windows Baltic Rim
	-1: PCMAP (same as 1252)
	-2: ANSI
<"file name">	*one of the following names, or the name of a user-defined character set file*
	"OCR-A.NSC" [OCR-A BT]
	"OCR-B.NSC" [OCR-B 10 Pitch BT]
	"ZAPF.NSC" [Zapf Dingbats BT]

11. Character Set Selection, cont'd.

Suppose you order the printer to print the character ASCII 124 dec. ASCII 124 will generate the following:

- |, according to the Roman 8 character set,
- ù, according to the French character set
- ñ, according to the Spanish set, etc.

In general, the Western European (Latin) character sets (numbers - 2 to 81 and number 351) are the same, with the exception of the few national characters. Each of the remaining character sets are specific to a limited number of languages, and should be used in conjunction with fonts for those languages.

A special case is the standard fonts OCR-A BT, OCR-B 10 Pitch BT and Zapf Dingbats BT. Before using any of these fonts for the printing of text or bar code interpretations, you must switch to their respective character sets, and switch back again to the original character set before selecting another font.

Choose the character set which best matches your data equipment and printout requirements. If any of the characters in this set do not have the ASCII value that you require, the **MAP** command can be used to asign that character to a different ASCII value (see chapter 6.10).

The **NASC** setting is not reset by **PRINTFEED|PF**, nor by turning the printer off. Once a **NASC** command has been issued, it affects the printed appearance of characters in any subsequent text field, but not in those text fields which have already been processed and stored in the print buffer. The **NASC** setting can be changed several times in one label, which allows labels to be multilingual.

Similarly, the **NASC** setting determines the appearance of the human readable portion of bar codes. The pattern of the bars in a bar code is determined by the ASCII values of the input data, and so is not affected by the **NASC** setting.

User-defined character sets can be created and stored in the printer, and separate commands exist to control the appearance of Asian characters. These are discussed in chapter 9.2.

12. Printer Reboot

Note that counters and error messages will be lost and a number of commands will be reset to default.

As an alternative to turning the printer off and on with the main switch, you can issue a **REBOOT** command. Any data or layout in the working memory that has not been saved will be deleted and the buffers will be emptied.

REBOOT

13. Verbosity Level

The verbosity level controls the amount of information to be returned from print to host:

SYSVAR (18) = <value>	
<value> = -1	*All levels enabled (Default)*
<value> = 0	*No verbosity (Recommended for best performance)*
<value> = 1	*Echo received characters*
<value> = 2	*"Ok" after correct command lines*
<value> = 4	*Echo input characters from communication port*
<value> = 8	*Error after failed line*

Bits can be combined, so e.g. ***SYSVAR(18)=3*** *means both "Echo received characters" and "Ok after correct command lines".*

14. Error Message Types

Four types of error messages can be selected:

SYSVAR (19) = <value>	
<value> = 1	*<string>, e.g. Invalid font (Default)*
<value> = 2	*Error <number> <string>, e.g. Error 19 Invalid font*
<value> = 3	*E<number>, e.g." E19"*
<value> = 4	*Error <number>, e.g."Error 19*

15. Error Handling

By default, the UBI Direct Protocol error handler handles four error conditions:

- Out of paper
- Head lifted
- Out of transfer ribbon
- Next label not found

All other errors are ignored unless specified by an **ERROR** command.

When an error occurs, the standard IN channel is flagged as busy and the **Power** LED changes from Green to Red. The message specified for the error by the **ERROR** command is sent to the standard OUT channel. In the case of the four error conditions listed above, a standard message in English will be sent to the host computer if no other message is specified by an **ERROR** command.

When an error occurs, press **Feed** to acknowledge the error, then correct it, either in setup or physically, e.g. by changing the transfer ribbon.

15. Error Handling, cont'd.

The **ERROR** command activates error-handling for the specified error type and allows you to write an error message, which will be transmitted back to the host computer (according to selected verbosity and type of error message, see chapters 6.13 and 6.14):

ERROR <error number>, "<error message>"	
<error number>	*error number (see chapter 12.1)*
<error message>	*text string enclosed by double quotation marks. Maximum 33 characters*

Example:

```
ERROR 43,"MEMORY OVERFLOW" ↵
```

Four error conditions are always handled without having to be activated by an **ERROR** command:

Out of paper

The **Power** LED turns Red. The printer sends a message back to the host computer. The printer waits for the printhead to be lifted and lowered, then the error is cancelled.

Head lifted

The **Power** LED turns Red. The printer sends a message back to the host computer. The printer waits for the printhead to be lowered, then a formfeed is performed. If the error stopped a print operation, the operation will be restarted automatically.

Out of transfer ribbon

The **Power** LED turns Red. The printer sends a message back to the host computer. The printer waits for a ribbon to be loaded. If the error stopped a print operation, the operation must be restarted.

Next label not found

The **Power** LED turns Red. The printer sends a message back to the host computer. The printer performs a formfeed and then stops and waits for the operator to press **Feed**.

Note that error handling and error messages specified by **ERROR** *commands are not saved in the printer's battery backed-up memory, but must be transmitted to the printer after each power-up.*

16. Break in Batch Printing

When printing large batches of labels, it is useful to be able to stop printing, for example, if an error is detected. You can stop printing via any serial communication channel.

Two commands allow you to issue a **BREAK** command:

- **BREAK** specifies an individual break interrupt character for each serial communication channel
- **BREAK ON/OFF** enables/disables break interrupt and deletes the break character for the corresponding devices.

BREAK <device>,<break char.>

<device> = 1	*"uart1:" serial communication channel (default)*
<device> = 4	*"centronics:" parallel communication channel (IN channel only)*
<break char.>	*is the ASCII decimal value of the desired break character. (default on both channels: ASCII 03 dec.)*

BREAK <device> **ON | OFF**

<device>	*the same parameter as in the BREAK command.*
Default:	*Communication channels disabled*

17. Printer Setup

The parameters in **SETUP** mode determine how the printer works. There are three ways to change the printer's setup:

- Setup strings
- Setup files
- Bar Code Wand

The various setup parameters are described in the EasyCoder 301 Installation and Operation manual.

Setup Strings

Setup strings allow you to change individual setup parameters directly from the host:

SETUP "<setup string>"

<setup string>	*a string of data according to the special syntax listed below and enclosed by double quotation marks.*

17. Printer Setup, cont'd.

In the syntax description below, characters separated by vertical bars indicate alternatives, n–nnnnn indicate variable numeric input and underscored space characters indicate compulsory space characters:

```
SETUP "CONTRAST,5"
SETUP "SER-COM,UART1,BAUDRATE,300|600|1200|2400|4800|9600|19200|38400"
SETUP "SER-COM,UART1,PARITY,NONE|EVEN|ODD|MARK|SPACE"
SETUP "SER-COM,UART1,CHAR_LENGTH,7|8"
SETUP "SER-COM,UART1,STOPBITS,1|2"
SETUP "SER-COM,UART1,FLOWCONTROL,RTS/CTS,ENABLE|DISABLE"
SETUP "SER-COM,UART1,FLOWCONTROL,ENQ/ACK,ENABLE|DISABLE"
SETUP "SER-COM,UART1,FLOWCONTROL,XON/XOFF,DATA_TO_HOST,ENABLE|DISABLE"
SETUP "SER-COM,UART1,FLOWCONTROL,XON/XOFF,DATA_FROM_HOST,ENABLE|DISABLE"
SETUP "SER-COM,UART1,NEW_LINE,CR/LF|LF|CR"
SETUP "DETECTION,LSS_ADJUST,n|nnn"
SETUP "DETECTION,FEEDADJ,STARTADJ,nnnn"
SETUP "DETECTION,FEEDADJ,STOPADJ,nnnn"
SETUP "SERVICE,MEDIA_SIZE,XSTART,nnn"
SETUP "SERVICE,MEDIA_SIZE,WIDTH,nnnn"
SETUP "SERVICE,MEDIA_SIZE,LENGTH,nnnn"
SETUP "SERVICE,MEDIA_TYPE,LABEL_(w_GAPS)|TICKET_(w_MARK)|TICKET_(w_GAPS)|
      FIX_LENGTH_STRIP|VAR_LENGTH_STRIP"
SETUP "SERVICE,PRINT_DEFS,HEAD_RESIST,nnn"
SETUP "SERVICE,PRINT_DEFS,PAPER_TYPE,<name of paper or transfer ribbon>"
SETUP "SERVICE,PRINT_DEFS,NEW_SUPPLIES,<encoded value>"
SETUP "SERVICE,PERFORMANCE,NORMAL|HIGH"
```

17. Printer Setup, cont'd.

Setup files

Two commands are used to set a printer up from a setup file: **SETUP WRITE** and **SETUP**.

Use a setup string (as described above) to set up the printer. Then use the **SETUP WRITE** command to save the current printer setup to a file.

SETUP WRITE "<filename>"

<filename>	*the full name of the setup file, including storage area, enclosed by double quotation marks.*

You can then select the stored setup file, using the **SETUP** command.

SETUP "<filename>"

<filename>	*the full name of the setup file, including storage area, enclosed by double quotation marks.*

Examples:

```
SETUP WRITE "setup1" ↵
SETUP "setup3" ↵
```

Bar CodeWand

Setting up the printer with the Bar Code Wand does not involve using UBI Direct Protocol.

See the EasyCoder 301 Installation and Operation manual for details on using the Bar Code Wand to set up the printer.

Reading the Printer's Status

1. Introduction

Provided that two-way communication is working between the printer and the host computer, the printer's status with regard to various functions can be returned to the host. This implies that the serial channel "uart1:" must be selected standard OUT channel (Default, see **SETSTDIO** command in chapter 6.2).

The **PRINT** command returns information about the printer's status via the standard OUT channel to the host, where it will usually be printed on the screen. A complementary command specifies the type of information to be returned. The shorthand version of **PRINT** is a question mark (**?**).

PRINT | ?

2. Date and Time

The current date and time according to the printer's clock/calendar can be read as follows:

? DATE$	*or*	**? DATE$("F")**
? TIME$	*or*	**? TIME$("F")**

3. Memory

Various parts of the printer's memory can be tested by reading the result of a **FUNCTEST$** command:

? FUNCTEST$ ("<RAM>|<ROM1>|<ROM2>|<CARD1>|<CARD2>"**)**

<RAM>	*returns the result of the power-up RAM test.*
<ROM1>|*<ROM2>*	*tests the ROM package and returns "ROM OK" or "ROM FAIL".*
<CARD1>|*<CARD2>*	*checks an inserted memory card, performs either a RAM or a ROM test depending on type of card and returns "CARD NOT PRESENT", "CARD OK", or "CARD FAIL".*

4. Printhead

The thermal printhead can be tested in three different ways:

- **FUNCTEST$ ("HEAD")**
- **? HEAD (type of check)**
- **? SYSVAR (<parameter>)**

? FUNCTEST$ ("HEAD")

The returns from the FUNCTEST$ ("HEAD") command are:
HEAD OK, SIZE:nnn DOTS (nnn is the number of dots)
HEAD LIFTED (lower the printhead and try again)
FAULTY PRINTHEAD (an error is detected)

? HEAD(<type of check>**)**

<type of check> = -1 Printhead. Returns -1 if OK, else 0
<type of check> = -7 Mean resistance of printhead in ohms

? SYSVAR (<parameter>**)**

<parameter> = 21	*returns printhead density in dots per mm*
<parameter> = 22	*returns the number of dots in the printhead*

5. System Counters

There are a number of counters in the printer's system, that can be read and returned to the host:

? SYSVAR (<parameter>**)**

<parameter> = 14	*number of errors detected since last startup*
<parameter> = 15	*number of errors detected since last SYSVAR (15)*
<parameter> = 24	*power-up status since last SYSVAR(24) (0 = No; 1=Yes)*

Note!
***SYSVAR (24)** is important, since essential functions, like counters and error messages will be lost at power up and other functions will be reset to their default values. Save all such data in the host and retransmit them to the printer as soon as a power-up has been detected by a polling program taking advantage of **SYSVAR(24)**.*

6. Sensors

The printer has a number of sensors and setup parameters that can be read and their status or value returned to the host:

? SYSVAR (<parameter>**)**	
<parameter> = 18	*returns selected verbosity level (see 6.13)*
<parameter> = 19	*returns selected type of error message (see 6.14)*
<parameter> = 20	*returns printer's paper type setup 0= Direct thermal printing; 1= Thermal transfer printing*
<parameter> = 23	*returns status of the ribbon end sensor (0=No ribbon; 1=Ribbon)*

7. Printer's Status

The printer's status with regard to various errors and other conditions can be read and returned to the host using the **PRSTAT** command:

? (PRSTAT AND <parameter>**)**	
<parameter> = 1	*printhead lifted*
<parameter> = 2	*label not removed*
<parameter> = 4	*printer out of paper*
<parameter> = 8	*printer out of transfer ribbon*

Parameters can be combined, e.g. ***<3>*** *checks for both "printhead lifted" and "label not removed" conditions. The printer will return 1 (= yes) or 0 (= no).*

8. Software and Hardware Versions

The version of EasyCoder 301 software, the type of printer family and the type of CPU board can be read using a **VERSION$** command:

? VERSION$[(type of info)**]**	
<type of info> = 0	*the version of EasyCoder 301 software*
<type of info> = 1	*301 = EasyCoder 301*
<type of info> = 2	*1-301700 = CPU board 700*

File Handling

1. Reading the Printer Memory

There are a number of commands for reading the printer's memory and returning the information to the host. This requires a working two-way serial communication (see **SETSTDIO** in chapter 6.1):

? FRE (1)

Returns the number of free bytes in the printer's RAM memory.

FONTS

Returns the names and size of all fonts files stored in the printer's memory, followed by a list of the bitmapped fonts retained for compatibilty with other EasyCoder printers (see chapter 4.3). It also returns information on the total memory used.

IMAGES

Returns the names of all image files (in .PCX format) stored in the printer's memory. It also returns information on the total memory used.

FILES ["<RAM:>|<ROM:>|<CARD1:>|<CARD2:>"]

Returns the names of all the files in the printer's RAM memory by default, if no storage device is specified, or optionally in the ROM memory (**"ROM:"**) or in an inserted DOS-formatted memory card (**"CARD1:|CARD2:"**).It will also return information on the size of each file and the total number of used bytes in the RAM memory.

2. Removing Images, Files and Fonts

When images are loaded using **IMAGE LOAD**, they can either be loaded into the printer's volatile memory or as a file into the RAM memory. Images loaded into the volatile memory are removed using **REMOVE IMAGE**. Images loaded into RAM are removed using **KILL**.

REMOVE IMAGE "<name>"

The name of the image must correspond exactly to the name returned when using the **IMAGES** command and be enclosed by double quotation marks.

2. Removing Images, Files and Fonts, cont'd.

Files, including font files and images files, can be removed from the RAM memory or from an inserted DOS-formatted memory card:

KILL "<filename>|CARD1:<filename>|CARD2:<filename>"

The name of the file must correspond exactly to the name returned when using a **FILES**, **IMAGES** or **FONTS** command and be enclosed by double quotation marks. Since files will be removed from the RAM memory by default, it is not necessary to specify **"RAM:"**. If the file name is preceded by **"CARD1:"** or **"CARD2:"**, the file will be removed from the inserted memory card.

KILL cannot be used to remove images from the printer's volatile memory.

3. Copying Files

You can copy a file to the RAM memory (**"RAM:"**) or to a DOS-formatted memory card (**"CARD1:"|"CARD2:"**). You cannot copy to or from the ROM memory (**"ROM:"**). You can also use the **COPY** command to give the copy a new name.

COPY "[CARD1:|CARD2:]<original name>" [,"[CARD1:|CARD2:]<new name>"]

The filename can be up to 30 characters. Since files will be copied to the RAM memory by default, it is not necessary to specify **"RAM:".**

Example:
COPY "CARD1:Logo.1","LOGO.1" ↵

4. Downloading Binary Files

Binary files, e.g. font files, can be downloaded to the printer's RAM memory using the **FILE& LOAD** command. Before the transfer can be performed, the printer must be set up for two way communication[1].

FILE& LOAD"<file name>",<file size>

<file name>	*the name (including the drive) you want to assign to the file, e.g. "CARD1:MYFILE.TXT".*
<file size>	*the size in bytes of the original file in the host.*

Upon receiving this command, the printer waits for the specified number of bytes to be received with a 25 second timeout between characters.

[1] ***Suggested serial port setup:***
Baudrate: 9600
No parity
8 bits per character
1 stop bit
RTS/CTS enabled
XON/XOFF:
Data to Host disabled.
XON/XOFF:
Data from Host disabled.

5. Downloading Image Files

Image files in .PCX format can be downloaded to "**RAM:**", **"CARD1:"** or **"CARD2:"** using the **IMAGE LOAD** command. Before the transfer can be performed, the printer must be set up for two way communication[1].

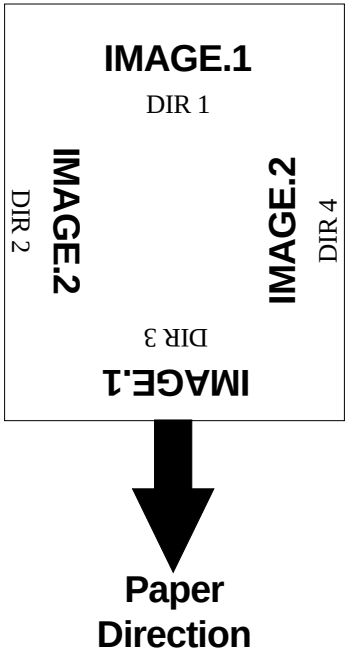
IMAGE LOAD"<image name>",<file size>,"<flag>"

<file name>	*the name (including the drive) you want to assign to the image, e.g. "CARD2:LOGO.1".*
<file size>	*the size in bytes of the original .PCX file in the host.*
<flag>	*either "S" or an empty string " ":* *S = the image will be saved in the battery backed-up part of the RAM memory;* *empty string ("") = the image will be saved in the volatile memory and will be deleted at next power-up.*

Upon receiving this command, the printer waits for the specified number of bytes to be received with a 25 second timeout between characters.

*[1]. **Suggested serial port setup:***
Baudrate: 9600
No parity
8 bits per character
1 stop bit
RTS/CTS enabled
XON/XOFF:
Data to Host disabled.
XON/XOFF:
Data from Host disabled.

We recommend that you add the extension .1 or .2 to the image name, so that you can tell which directions the image is suited for before you print it out, i.e.:
IMAGE.1 is suited for directions **across** the paper web, i.e. DIR 1 & DIR3.
IMAGE.2 is suited for directions **along** the paper web, i.e. DIR 2 & DIR4.



Advanced Features

1. Specifying Complex Bar Codes

Complex 2-dimensional bar or dot codes, e.g. PDF417, require many specifying parameters. Therefore, there is a complex command that allows all bar code parameters to be specified within a single command. This command can also be used for one-dimensional bar codes (see chapter 4.4).

BARSET[#<start parameter>,]["<bar code name>"[,<ratio wide bars>
[,<ratio narrow bars>[,<magnification>[,<height> [,<security level>
[,<aspect ratio height>[,<aspect ratio width>[,<no. of rows>[,<no. of columns>
[,<truncate flag>]]]]]]]]]]

<start parameter>	*specifies first parameter in the command (default 1)*
<bar code name>	*specifies bar code (default "INT2OF5")*
<ratio wide bars>	*specifies ratio wide/narrow bars (default 3)*
<ratio narrow bars>	*specifies ratio wide/narrow bars (default 1)*
<magnification>	*specifies magnification (default 2)*
<height>	*specifies height of bars in dots (default 100)*
<security level>	*only used in PDF417 and Maxicode (default 2)*
<aspect height ratio>	*only used in PDF417 and Maxicode (default 1)*
<aspect height width>	*only used in PDF417 and Maxicode (default 2)*
<no. of rows>	*only used in PDF417 and Maxicode (default 0)*
<no. of columns>	*only used in PDF417 and Maxicode (default 0)*
<truncate flag>	*only used in PDF417 and Maxicode (default 0)*

2. Using International Character Sets

The data input to text fields and bar codes takes the form of an ASCII string. Even compound data, such as...

```
PRTXT "Label No.";CNT1$;" Date ";DATE$("F")
```

is expanded by the printer into a single ASCII string before any other processing occurs. This ASCII string is then converted to a string of international character codes according to the Unicode standard. Every character, whether it be a Latin "A", Greek "alpha" or Chinese "ren", has a unique code in this standard, called its "unicode".

Unicoded fonts

The EasyCoder 301 works with Unicoded fonts in either Bitstream Speedo or TrueType format: its font scaler accesses images of characters within these fonts according to the character's unicode. Some care is required when using fonts not built into the printer:

- If a font is not Unicoded, the font scaler will access the wrong characters.
- A font may not include all the characters in the current character set, as specified by the **NASC** command. If a text field contains a character which is not in the current font, then a replacement character will usually be printed instead. The replacement character for Speedo fonts is a space, while for TrueType fonts it is usually an unfilled rectangle.

NASC and NASCD tables

There are many national and international standards for mapping ASCII strings to strings of unicode. The EasyCoder 301 provides support for virtually all of these. There are two types of mappings:

- Single-byte mappings, which map one ASCII character to a unicode. The EasyCoder 301 supports these using "NASC tables". For example, for each **NASC** setting listed in chapter 6.11, the EasyCoder 301 has an internally stored NASC table. User defined NASC tables can also be stored in the printer as a file (see later in this chapter).
- Double-byte mappings, which map pairs of ASCII characters to unicodes. The EasyCoder 301 supports these using "NASCD tables". Currently, there are NASCD tables for the most common double-byte mappings used for the large Asian character sets: BIG5, GB, JIS and Shift-JIS.

2. Using International Character Sets, cont'd.

To be exact, the double-byte mappings allow an ASCII string to contain a mixture of single-byte and double-bytes codes. For example, in the BIG5 mapping, any ASCII character with value 160 or greater is the first byte of a double-byte code, while the remaining characters form single-byte codes. So, in BIG5, the ASCII string "**<160><64><65>**" splits into the double-byte code "**<160><64>**" and the single-byte code "**<65>**". In the EasyCoder 301, the current NASCD table specifies both how to divide an ASCII string into single-byte and double-byte codes and also how the double-byte codes are mapped to unicodes. The current NASC table specifies how the single-byte codes are mapped to unicodes. The relevant commands are:

NASC <character set no.> | <"file name">

NASCD "<NASCD table file>"

<character set no.>	*one of the values listed in chapter 6.11*
<"file name">	*one of the character set names listed in chapter 6.11 or the name of a user-defined NASC table (see later in this chapter)*
<NASCD table file>	*the file name of a **NASCD** table.*
Default:	***NASCD** "" (empty string; disables double-byte interpretation of ASCII strings)*

Double-byte fonts

As discussed above, the ASCII data input to text fields and human readable parts of bar codes can contain a mixture of single-byte and double-byte codes, which are mapped to unicodes by the **NASC** and **NASCD** settings respectively. The characters for the single-byte codes are printed using the current single-byte font, as specified by the **FONT** command (see chapter 4.3). The characters for the double-byte codes are printed using the current double-byte font, as specified by the following commands:

FONTD | FD "<font file name>" [,<font size>,<font slant>]

<font file name>	*the name of a Speedo or TrueType font file; it must be enclosed by double quotation marks.*
<font size>	*the height of the characters in points (a point is a standard typographic unit, equal to 1/72 inches).*
<font slant>	*the italic angle of the characters in degrees; a positive value slants the characters clockwise away from the vertical. Default: 0.*
Reset to default by:	*PRINTFEED|PF*

2. Using International Character Sets, cont'd.

Label produced by the program example to the right.

The three double-byte font parameters can be specified separately using the following commands:

FONTD | FD "<font file name>"

FONTDSIZE | FSD <font size>

FONTDSLANT | FLD <font slant>

For each parameter, the value specified by these commands will be used in all text fields on the current label, until a new value is specified. These parameters need only be specified if a **NASCD** file is specified, as otherwise double-byte interpretation is disabled.

Example:

```
NASC 1 ↵
NASCD "CARD1:BIG5.NCD" ↵
FONT "Swiss 721 BT",20,0 ↵
FONTD "CARD1:CHINESE.TTF",20,0 ↵
PP 100,100 ↵
PT "NASCD ON: ";CHR$(185);CHR$(235) ↵
NASCD "" ↵
PP 100,200 ↵
PT "NASCD OFF: ";CHR$(185);CHR$(235) ↵
PF ↵
```

Double-byte fonts can also be used in the human readable parts of barcodes. The corresponding commands are:

BARFONTD | BFD "<font file name>"[,<font size>,<font slant>]

<font file name>	*the file name of the double-byte font to be use*
<font size>	*the height of the font in points. Default 12.*
<font slant>	*the italic angle of the font in degrees clockwise Default: 0.*
Reset to default by:	*PRINTFEED|PF*

The bar code font size and slant can be specified separately using the following commands:

BARFONTDSIZE | BFSD <font size>

BARFONTDSLANT | BLFD <font slant>

Double-byte interpretation of human readables is disabled unless a NASCD file is specified. Also, note that many double-byte mappings use ASCII characters greater than 127, although these characters are not allowed as input data to many bar code types. PDF 417 is one bar code type that does allow ASCII characters greater than 127 as input.

2. Using International Character Sets, cont'd.

User-defined NASC tables

A user-defined NASC table can be used in place of one of the built-in NASC settings. It specifies a unicode for each of the 256 ASCII characters, and determines the mapping of single-byte codes in text fields and human readable parts of bar codes. A user-defined NASC table is specified using the command:

NASC "<file name>"

<file name>	*the name of a file containing a user-defined NASC table.*
Default:	*NASC 1 (the built-in Roman 8 NASC table is used)*

Such a file can be created using some text editors, or by a user-written piece of software. It can be sent to the printer using the **FILE& LOAD** command (see page 65) or copied onto a MSDOS formatted memory card which can be inserted in one of the printer's PCMCIA slots. It is a binary file 516 bytes long, with the following format:

Byte 0:	N
Byte 1:	S
Byte 2:	C
Byte 3:	1
Byte 4:	high byte of unicode for ASCII 0
Byte 5:	low byte of unicode for ASCII 0
Byte 6:	high byte of unicode for ASCII 1
Byte 7:	low byte of unicode for ASCII 1
↓	↓ ↓
Byte 514:	high byte of unicode for ASCII 255
Byte 515:	low byte of unicode for ASCII 255.

3. Font Scaling

In general, the EasyCoder 301 scales characters "on the fly" into bitmaps ready for printing. It stores as many character bitmaps as possible in a memory cache, to avoid having to rescale characters which are used more than once. Character bitmaps are deleted from the cache automatically by the printer, and so characters which are used often usually need to be rescaled frequently.

In some instances it can take a long time for the printer to scale characters, such as when using double-byte TrueType fonts on a card. The user can force the printer to hold character bitmaps in the cache using the following command:

PRESCALE | PS "<string of characters>"

<string of char.>	*an ASCII string specifying all the characters that are to be scaled from the current single- and double-byte fonts. It is interpreted according to the current NASC and NASCD settings.*

Example:
The 10 digits from a single-byte font and the ren character from a double-byte font are prescaled.

```
NASCD "CARD1:BIG5.NCD" ↵
FONT "Swiss 721 BT",50,10 ↵
FONTD "CARD1:HOMINCHO.TTF" ↵
DIR 2 ↵
MAG 2,1 ↵
PS "0123456789";CHR$(164);CHR$(56)↵
```

The contents of a character bitmap depend on all of the following settings:

- Single-byte character:
  DIR, MAG, NASC, FONT, FONTSIZE, FONTSLANT
- Double-byte character:
  DIR, MAG, NASCD, FONTD, FONTDSIZE, FONTDSLANT

The printer holds prescaled characters for the last 15 combinations of these settings. When a new combination is used, (for example, when the specification of **DIR** is changed), the prescaled characters for the 15th previous combination are lost. All prescaled characters are lost when the printer is turned off.

Firmware Upgrade

1. Software and Fonts

Firmware (software and resident fonts) can be upgraded using 1 or 2 PCMCIA cards, which may contain either printer software updates or fonts[1]. In either case, use the following procedure:

1 Print out a test label, which will contain all your printer settings as these will be lost during the upgrade[2].

2 Switch off the printer.

3 Insert the card in either slot with its top facing the ON/OFF switch.

4 Switch on the printer. The data on the card is copied to the printer's flash memory.

The Power LED should go through the following sequence, which takes about 30 seconds..
- Flash amber
- Continuous amber
- Flash amber
- Continuous amber
- Green

WARNING!
Do not switch off the printer until the upgrading is completed and the Power LED shines green!

5 Switch off the printer.

6 Remove the card.

7 Insert the next card, if necessary, and repeat steps 4 – 6.

8 Switch on the printer.

9 Enter the settings shown on the test label you printed earlier (see chapter 6.9).

10 Reset the printhead resistance setting using the command:
`?Head (-1)`

[1]/. This chapter refers to cards containing fonts which will be loaded into the printer's flash memory. Font cartridges which are permanently fitted in the printer are not covered.

[2]/. You cannot save your settings to a text file and use it later when you upgrade firmware or fonts, since all your files are lost and the printer settings are reset to factory defaults during the upgrade,

Character Set, Fonts and Bar Codes

1. Character Sets

This chapter contains the various single-byte character sets, that can be selected using the **NASC** statement. Double-byte character sets are not included, but are available separately on special request.

The following information applies to all single-byte character sets:

- Characters between ASCII 0 decimal and ASCII 31 decimal are unprintable control characters as listed below.
- Characters between ASCII 32 decimal and ASCII 127 decimal can always be printed, regardless of 7-bit or 8-bit communication protocol, provided that the selected font contains the characters in question.
- Characters between ASCII 127 decimal and ASCII 255 decimal can only be printed if the selected font contains the characters in question and an 8-bit communication protocol is used. If you use 7-bit communication, select another national character set (see **NASC** statement in chapter 6.11) or use a **MAP** statement to remap a character set (see chapter 6.10).
- If a character, which does not exist in the selected font, is used, a space will be substituted in a Speedo font, and an empty rectangle in a TrueType font.

Non-printable control characters (ASCII 00 – 31 dec):

00	**NUL**	**16**	**DLE**
01	**SOH**	**17**	**DC1**
02	**STX**	**18**	**DC2**
03	**ETX**	**19**	**DC3**
04	**EOT**	**20**	**DC4**
05	**ENQ**	**21**	**NAK**
06	**ACK**	**22**	**SYN**
07	**BEL**	**23**	**ETB**
08	**BS**	**24**	**CAN**
09	**HT**	**25**	**EM**
10	**LF**	**26**	**SUB**
11	**VT**	**27**	**ESC**
12	**FF**	**28**	**FS**
13	**CR**	**29**	**GS**
14	**SO**	**30**	**RS**
15	**SI**	**31**	**US**

1. Character Sets, cont'd.

Roman 8 (NASC 1)

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	[	\	]	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	{	|	}	~		□	□
130	□	□	□	□	□	□	□	□	□	□
140	□	□	□	□	□	□	□	□	□	□
150	□	□	□	□	□	□	□	□	□	□
160		À	Â	È	Ê	Ë	Î	Ï	´	`
170	^	¨	~	Ù	Û		₋	Ý	ý	°
180	Ç	ç	Ñ	ñ	¡	¿	¤	£	¥	§
190	ƒ	¢	â	ê	ô	û	á	é	ó	ú
200	à	è	ò	ù	ä	ë	ö	ü	Å	î
210	Ø	Æ	å	í	ø	æ	Ä	ì	Ö	Ü
220	É	ï	ß	Ô	Á	Ã	ã	Đ	ð	Í
230	Ì	Ó	Ò	Õ	õ	Š	š	Ú	Ÿ	ÿ
240	Þ	þ	·	µ	¶	¾	—	¼	½	ª
250	º	«		»	±	□				

1. Character Sets, cont'd.

French (NASC 33)

	0	1	2	3	4	5	6	7	8	9
30				!	"	£	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	à	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	°	ç	§	^	_	µ	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	é	ù	è	¨		□	□
130	□	□	□	□	□	□	□	□	□	□
140	□	□	□	□	□	□	□	□	□	□
150	□	□	□	□	□	□	□	□	□	□
160		À	Â	È	Ê	Ë	Î	Ï	´	`
170	^	¨	~	Ù	Û		-	Ý	ý	°
180	Ç	ç	Ñ	ñ	¡	¿	¤	£	¥	§
190	ƒ	¢	â	ê	ô	û	á	é	ó	ú
200	à	è	ò	ù	ä	ë	ö	ü	Å	î
210	Ø	Æ	å	í	ø	æ	Ä	ì	Ö	Ü
220	É	ï	ß	Ô	Á	Ã	ã	Ð	ð	Í
230	Ì	Ó	Ò	Õ	õ	Š	š	Ú	Ÿ	ÿ
240	Þ	þ	·	µ	¶	¾	—	¼	½	ª
250	º	«		»	±	□				

1. Character Sets, cont'd.

Spanish (NASC 34)

	0	1	2	3	4	5	6	7	8	9
30				!	"	£	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	§	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	i	Ñ	¿	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	°	ñ	ç	~		□	□
130	□	□	□	□	□	□	□	□	□	□
140	□	□	□	□	□	□	□	□	□	□
150	□	□	□	□	□	□	□	□	□	□
160		À	Â	È	Ê	Ë	Î	Ï	´	`
170	^	¨	~	Ù	Û		¯	Ý	ý	°
180	Ç	ç	Ñ	ñ	¡	¿	¤	£	¥	§
190	ƒ	¢	â	ê	ô	û	á	é	ó	ú
200	à	è	ò	ù	ä	ë	ö	ü	Å	î
210	Ø	Æ	å	í	ø	æ	Ä	ì	Ö	Ü
220	É	ï	ß	Ô	Á	Ã	ã	Ð	ð	Í
230	Ì	Ó	Ò	Õ	õ	Š	š	Ú	Ÿ	ÿ
240	Þ	þ	·	µ	¶	¾	—	¼	½	ª
250	º	«		»	±	□				

1. Character Sets, cont'd.

Italian
(NASC 39)

	0	1	2	3	4	5	6	7	8	9
30				!	"	£	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	§	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	°	ç	é	^	_	ù	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	à	ò	è	ì		□	□
130	□	□	□	□	□	□	□	□	□	□
140	□	□	□	□	□	□	□	□	□	□
150	□	□	□	□	□	□	□	□	□	□
160		À	Â	È	Ê	Ë	Î	Ï	´	`
170	^	¨	~	Ù	Û		¯	Ý	ý	°
180	Ç	ç	Ñ	ñ	¡	¿	¤	£	¥	§
190	ƒ	¢	â	ê	ô	û	á	é	ó	ú
200	à	è	ò	ù	ä	ë	ö	ü	Å	î
210	Ø	Æ	å	í	ø	æ	Ä	ì	Ö	Ü
220	É	ï	ß	Ô	Á	Ã	ã	Ð	ð	Í
230	Ì	Ó	Ò	Õ	õ	Š	š	Ú	Ÿ	ÿ
240	Þ	þ	·	µ	¶	¾	—	¼	½	ª
250	º	«		»	±	□				

1. Character Sets, cont'd.

English (UK)
(NASC 44)

	0	1	2	3	4	5	6	7	8	9
30				!	"	£	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	[	\	]	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	{	|	}	~		□	□
130	□	□	□	□	□	□	□	□	□	□
140	□	□	□	□	□	□	□	□	□	□
150	□	□	□	□	□	□	□	□	□	□
160		À	Â	È	Ê	Ë	Î	Ï	´	`
170	^	¨	~	Ù	Û		-	Ý	ý	°
180	Ç	ç	Ñ	ñ	¡	¿	¤	£	¥	§
190	ƒ	¢	â	ê	ô	û	á	é	ó	ú
200	à	è	ò	ù	ä	ë	ö	ü	Å	î
210	Ø	Æ	å	í	ø	æ	Ä	ì	Ö	Ü
220	É	ï	ß	Ô	Á	Ã	ã	Đ	ð	Í
230	Ì	Ó	Ò	Õ	õ	Š	š	Ú	Ÿ	ÿ
240	Þ	þ	·	µ	¶	¾	—	¼	½	ª
250	º	«		»	±	□				

1. Character Sets, cont'd.

Swedish (NASC 46)

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	¤	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	É	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	Ä	Ö	Å	Ü	_	é	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	ä	ö	å	ü		□	□
130	□	□	□	□	□	□	□	□	□	□
140	□	□	□	□	□	□	□	□	□	□
150	□	□	□	□	□	□	□	□	□	□
160		À	Â	È	Ê	Ë	Î	Ï	´	`
170	^	¨	~	Ù	Û		-	Ý	ý	◦
180	Ç	ç	Ñ	ñ	¡	¿	¤	£	¥	§
190	ƒ	¢	â	ê	ô	û	á	é	ó	ú
200	à	è	ò	ù	ä	ë	ö	ü	Å	î
210	Ø	Æ	å	í	ø	æ	Ä	ì	Ö	Ü
220	É	ï	ß	Ô	Á	Ã	ã	Ð	ð	Í
230	Ì	Ó	Ò	Õ	õ	Š	š	Ú	Ÿ	ÿ
240	Þ	þ	·	µ	¶	¾	—	¼	½	ª
250	º	«		»	±	□				

1. Character Sets, cont'd.

Norwegian (NASC 47)

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	Æ	Ø	Å	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	æ	ø	å	¯		□	□
130	□	□	□	□	□	□	□	□	□	□
140	□	□	□	□	□	□	□	□	□	□
150	□	□	□	□	□	□	□	□	□	□
160		À	Â	È	Ê	Ë	Î	Ï	´	`
170	^	¨	~	Ù	Û		¯	Ý	ý	°
180	Ç	ç	Ñ	ñ	¡	¿	¤	£	¥	§
190	ƒ	¢	â	ê	ô	û	á	é	ó	ú
200	à	è	ò	ù	ä	ë	ö	ü	Å	î
210	Ø	Æ	å	í	ø	æ	Ä	ì	Ö	Ü
220	É	ï	ß	Ô	Á	Ã	ã	Ð	ð	Í
230	Ì	Ó	Ò	Õ	õ	Š	š	Ú	Ÿ	ÿ
240	Þ	þ	·	µ	¶	¾	—	¼	½	ª
250	º	«		»	±	□				

1. Character Sets, cont'd.

German (NASC 49)

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	§	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	Ä	Ö	Ü	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	ä	ö	ü	ß		□	□
130	□	□	□	□	□	□	□	□	□	□
140	□	□	□	□	□	□	□	□	□	□
150	□	□	□	□	□	□	□	□	□	□
160		À	Â	È	Ê	Ë	Î	Ï	´	`
170	^	¨	~	Ù	Û		¯	Ý	ý	°
180	Ç	ç	Ñ	ñ	¡	¿	¤	£	¥	§
190	ƒ	¢	â	ê	ô	û	á	é	ó	ú
200	à	è	ò	ù	ä	ë	ö	ü	Å	î
210	Ø	Æ	å	í	ø	æ	Ä	ì	Ö	Ü
220	É	ï	ß	Ô	Á	Ã	ã	Đ	ð	Í
230	Ì	Ó	Ò	Õ	õ	Š	š	Ú	Ÿ	ÿ
240	Þ	þ	·	µ	¶	¾	—	¼	½	ª
250	º	«		»	±	□				

1. Character Sets, cont'd.

Japanese Latin (NASC 81)

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	[	¥	]	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	{	|	}	~		□	□
130	□	□	□	□	□	□	□	□	□	□
140	□	□	□	□	□	□	□	□	□	□
150	□	□	□	□	□	□	□	□	□	□
160		Á	Â	È	Ê	Ë	Î	Ï	´	`
170	^	¨	~	Ù	Û		¯	Ý	ý	°
180	Ç	ç	Ñ	ñ	¡	¿	¤	£	¥	§
190	ƒ	¢	â	ê	ô	û	á	é	ó	ú
200	à	è	ò	ù	ä	ë	ö	ü	Å	î
210	Ø	Æ	å	í	ø	æ	Ä	ì	Ö	Ü
220	É	ï	ß	Ô	Á	Ã	ã	Ð	ð	Í
230	Ì	Ó	Ò	Õ	õ	Š	š	Ú	Ÿ	ÿ
240	Þ	þ	·	µ	¶	¾	—	¼	½	ª
250	º	«		»	±	□				

1. Character Sets, cont'd.

Portuguese (NASC 351)

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	§	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	Ã	Ç	Õ	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	ã	ç	õ	°		□	□
130	□	□	□	□	□	□	□	□	□	□
140	□	□	□	□	□	□	□	□	□	□
150	□	□	□	□	□	□	□	□	□	□
160		À	Â	È	Ê	Ë	Î	Ï	´	`
170	^	¨	~	Ù	Û		-	Ý	ý	°
180	Ç	ç	Ñ	ñ	¡	¿	¤	£	¥	§
190	ƒ	¢	â	ê	ô	û	á	é	ó	ú
200	à	è	ò	ù	ä	ë	ö	ü	Å	î
210	Ø	Æ	å	í	ø	æ	Ä	ì	Ö	Ü
220	É	ï	ß	Ô	Á	Ã	ã	Ð	ð	Í
230	Ì	Ó	Ò	Õ	õ	Š	š	Ú	Ÿ	ÿ
240	Þ	þ	·	µ	¶	¾	—	¼	½	ª
250	º	«		»	±	□				

1. Character Sets, cont'd.

MS-DOS Latin 1 (NASC 850)

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	[	\	]	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	{	|	}	~		Ç	ü
130	é	â	ä	à	å	ç	ê	ë	è	ï
140	î	ì	Ä	Å	É	æ	Æ	ô	ö	ò
150	û	ù	ÿ	Ö	Ü	ø	£	Ø	×	ƒ
160	á	í	ó	ú	ñ	Ñ	ª	º	¿	®
170	¬	½	¼	¡	«	»				
180		Á	Â	À	©					¢
190	¥								ã	Ã
200								¤	ð	Ð
210	Ê	Ë	È	ı	Í	Î	Ï			
220		¦	Ì		Ó	ß	Ô	Ò	õ	Õ
230	µ	þ	Þ	Ú	Û	Ù	ý	Ý	¯	´
240	-	±		¾	¶	§	÷	¸	°	¨
250	·	1	3	2						

1. Character Sets, cont'd.

MS-DOS Greek 1 (NASC 851)

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	[	\	]	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	{	|	}	~		Ç	ü
130	é	â	ä	à	Ά	ç	ê	ë	è	ï
140	î	Έ	Ä	Ή	Ί	□	Ό	ô	ö	Ύ
150	û	ù	Ώ	Ö	Ü	ά	£	έ	ή	ί
160	ϊ	ΐ	ό	ύ	A	B	Γ	Δ	E	Z
170	H	½	Θ	I	«	»		▒		
180		K	Λ	M	N					Ξ
190	O								Π	P
200								Σ	T	Y
210	Φ	X	Ψ	Ω	α	β	γ			
220		δ	ε		ζ	η	θ	ι	κ	λ
230	μ	ν	ξ	ο	π	ρ	σ	ς	τ	΄
240	-	±	υ	φ	χ	§	ψ	¸	°	¨
250	ω	ϋ	ΰ	ώ						

1. Character Sets, cont'd.

MS-DOS Latin 2 (NASC 852)

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	[	\	]	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	{	|	}	~		Ç	ü
130	é	â	ä	ů	ć	ç	ł	ë	Ő	ő
140	î	Ź	Ä	Ć	É	Ĺ	Í	ô	ö	Ľ
150	ľ	Ś	ś	Ö	Ü	Ť	ť	Ł	×	č
160	á	í	ó	ú	Ą	ą	Ž	ž	Ę	ę
170	¬	ź	Č	ş	«	»		▒		
180		Á	Â	Ę	Ş					Ż
190	ż								Ă	ă
200								¤	đ	Đ
210	Ď	Ë	ď	Ň	Í	Î	ě			
220		Ţ	Ů		Ó	ß	Ô	Ń	ń	ň
230	Š	š	Ŕ	Ú	ŕ	Ű	ý	Ý	ţ	´
240	-	˝		˘	˘	§	÷	¸	°	¨
250	˙	ű	Ř	ř						

1. Character Sets, cont'd.

MS-DOS Cyrillic (NASC 855)

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	[	\	]	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	{	|	}	~		ђ	Ђ
130	ѓ	Ѓ	ё	Ё	є	Є	s	S	i	I
140	ї	Ї	j	J	љ	Љ	њ	Њ	ћ	Ћ
150	ќ	Ќ	ў	Ў	џ	Џ	ю	Ю	ъ	Ъ
160	а	А	б	Б	ц	Ц	д	Д	е	Е
170	ф	Ф	г	Г	«	»		▒		
180		х	Х	и	И					й
190	Й								к	К
200								¤	л	Л
210	м	М	н	Н	о	О	п			
220		П	я		Я	р	Р	с	С	т
230	Т	у	У	ж	Ж	в	В	ь	Ь	№
240	-	ы	Ы	з	З	ш	Ш	э	Э	щ
250	Щ	ч	Ч	§						

1. Character Sets, cont'd.

MS-DOS Turkish (NASC 857)

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	[	\	]	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	{	|	}	~		Ç	ü
130	é	â	ä	à	å	ç	ê	ë	è	ï
140	î	ı	Ä	Å	É	æ	Æ	ô	ö	ò
150	û	ù	İ	Ö	Ü	ø	£	Ø	Ş	ş
160	á	í	ó	ú	ñ	Ñ	Ğ	ğ	¿	®
170	¬	½	¼	¡	«	»		▒		
180		Á	Â	À	©					¢
190	¥								ã	Ã
200								¤	º	ª
210	Ê	Ë	È	□	Í	Î	Ï			
220		¦	Ì		Ó	ß	Ô	Ò	õ	Õ
230	µ	□	×	Ú	Û	Ù	ì	ÿ	¯	´
240	-	±	□	¾	¶	§	÷	¸	°	¨
250	·	1	3	2						

1. Character Sets, cont'd.

Windows Latin 2 (NASC 1250)

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	[	\	]	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	{	|	}	~		□	□
130	˛	□	„	…	†	‡	□	‰	Š	‹
140	Ś	Ť	Ž	Ź	□	‘	’	“	”	•
150	–	—	□	™	š	›	ś	ť	ž	ź
160		ˇ	˘	Ł	¤	Ą	¦	§	¨	©
170	Ş	«	¬	-	®	Ż	°	±	˛	ł
180	´	µ	¶	·	¸	ą	ş	»	Ľ	˝
190	ľ	ż	Ŕ	Á	Â	Ă	Ä	Ĺ	Ć	Ç
200	Č	É	Ę	Ë	Ě	Í	Î	Ď	Đ	Ń
210	Ň	Ó	Ô	Ő	Ö	×	Ř	Ů	Ú	Ü
220	Ü	Ý	Ţ	ß	ŕ	á	â	ă	ä	Í
230	ć	ç	č	é	ę	ë	ě	í	î	ď
240	đ	ń	ň	ó	ô	ő	ö	÷	ř	ů
250	ú	ű	ü	ý	ţ	˙				

1. Character Sets, cont'd.

Windows Cyrillic (NASC 1251)

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	[	\	]	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	{	|	}	~		Ђ	Ѓ
130	‚	ѓ	„	…	†	‡	□	‰	Љ	‹
140	Њ	Ќ	Ћ	Џ	ђ	‘	’	“	”	•
150	–	—	□	™	љ	›	њ	ќ	ћ	џ
160		Ў	ў	Ј	¤	Ґ	¦	§	Ё	©
170	Є	«	¬	-	®	Ї	°	±	І	і
180	ґ	µ	¶	·	ё	№	є	»	ј	S
190	s	ї	А	Б	В	Г	Д	Е	Ж	З
200	И	Й	К	Л	М	Н	О	П	Р	С
210	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы
220	Ь	Э	Ю	Я	а	б	в	г	д	е
230	ж	з	и	й	к	л	м	н	о	п
240	р	с	т	у	ф	х	ц	ч	ш	щ
250	ъ	ы	ь	э	ю	я				

1. Character Sets, cont'd.

Windows Latin 1 (NASC 1252)

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	[	\	]	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	{	|	}	~		□	□
130	‚	ƒ	„	…	†	‡	ˆ	‰	Š	‹
140	Œ	□	□	□	□	‘	’	“	”	•
150	–	—	˜	™	š	›	œ	□	□	Ÿ
160		¡	¢	£	¤	¥	¦	§	¨	©
170	ª	«	¬	-	®	¯	°	±	2	3
180	´	µ	¶	·	¸	1	º	»	¼	½
190	¾	¿	À	Á	Â	Ã	Ä	Å	Æ	Ç
200	È	É	Ê	Ë	Ì	Í	Î	Ï	Ð	Ñ
210	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û
220	Ü	Ý	Þ	ß	à	á	â	ã	ä	å
230	æ	ç	è	é	ê	ë	ì	í	î	ï
240	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù
250	ú	û	ü	ý	þ	ÿ				

1. Character Sets, cont'd.

Windows Greek (NASC 1253)

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	[	\	]	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	{	|	}	~		□	□
130	‚	ƒ	„	…	†	‡	□	‰	□	‹
140	□	□	□	□	□	‘	’	“	”	•
150	–	—	□	™	□	›	□	□	□	□
160			Ά	£	¤	¥	¦	§	¨	©
170	□	«	¬	-	®		°	±	2	3
180	΄	μ	¶	·	Έ	Ή	Ί	»	Ό	½
190	Ύ	Ώ	ΐ	Α	Β	Γ	Δ	Ε	Ζ	Η
200	Θ	Ι	Κ	Λ	Μ	Ν	Ξ	Ο	Π	Ρ
210	□	Σ	Τ	Υ	Φ	Χ	Ψ	Ω	Ϊ	Ϋ
220	ά	έ	ή	ί	ΰ	α	β	γ	δ	ε
230	ζ	η	θ	ι	κ	λ	μ	ν	ξ	ο
240	π	ρ	ς	σ	τ	υ	φ	χ	ψ	ω
250	ϊ	ϋ	ό	ύ	ώ	□				

1. Character Sets, cont'd.

Windows Latin 5 (NASC 1254)

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	[	\	]	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	{	|	}	~		□	□
130	‚	ƒ	„	…	†	‡	ˆ	‰	Š	‹
140	Œ	□	□	□	□	‘	’	“	”	•
150	–	—	˜	™	š	›	œ	□	□	Ÿ
160		¡	¢	£	¤	¥	¦	§	¨	©
170	ª	«	¬	-	®	¯	°	±	2	3
180	´	µ	¶	·	¸	1	º	»	¼	½
190	¾	¿	À	Á	Â	Ã	Ä	Å	Æ	Ç
200	È	É	Ê	Ë	Ì	Í	Î	Ï	Ğ	Ñ
210	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û
220	Ü	İ	Ş	ß	à	á	â	ã	ä	å
230	æ	ç	è	é	ê	ë	ì	í	î	ï
240	ğ	ñ	ò	ó	ô	õ	ö	÷	ø	ù
250	ú	û	ü	ı	ş	ÿ				

1. Character Sets, cont'd.

Windows Baltic Rim (NASC 1257)

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	[	\	]	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	{	|	}	~		□	□
130	‚	□	„	…	†	‡	□	‰	□	‹
140	□	¨	ˇ	¸	□	‘	’	“	”	•
150	–	—	□	™	□	›	□	¯	˛	□
160		□	¢	£	¤	□	¦	§	Ø	©
170	Ŗ	«	¬	-	®	Æ	°	±	2	3
180	´	µ	¶	·	ø	1	ŗ	»	¼	½
190	¾	æ	Ą	Į	Ā	Ć	Ä	Å	Ę	Ē
200	Č	É	Ź	Ė	Ģ	Ķ	Ī	Ļ	Š	Ń
210	Ņ	Ó	Ō	Õ	Ö	×	Ų	Ł	Ś	Ū
220	Ü	Ż	Ž	ß	ą	į	ā	ć	ä	å
230	ę	ē	č	é	ź	ė	ģ	ķ	ī	ļ
240	š	ń	ņ	ó	ō	õ	ö	÷	ų	ł
250	ś	ū	ü	ż	ž	˙				

1. Character Sets, cont'd.

PCMAP
(NASC -1)

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	[	\	]	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	{	|	}	~		Ç	ü
130	é	â	ä	à	å	ç	ê	ë	è	ï
140	î	ì	Ä	Å	É	æ	Æ	ô	ö	ò
150	û	ù	ÿ	Ö	Ü	¢	£	¥	§	ƒ
160	á	í	ó	ú	ñ	Ñ	ª	º	¿	`
170	^	½	¼	¡	«	»	¯	Ý	ý	°
180	Ç	ç	Ñ	ñ	¡	¿	¤	£	¥	§
190	ƒ	¢	â	ê	ô	û	á	é	ó	ú
200	à	è	ò	ù	ä	ë	ö	ü	Å	î
210	Ø	Æ	å	í	ø	æ	Ä	ì	Ö	Ü
220	É	ï	ß	Ô	Á	Ã	ã	Đ	ð	Í
230	Ì	Ó	Ò	Õ	õ	Š	š	Ú	Ÿ	ÿ
240	Þ	þ	·	µ	¶	¾	—	¼	½	ª
250	º	«		»	±	□				

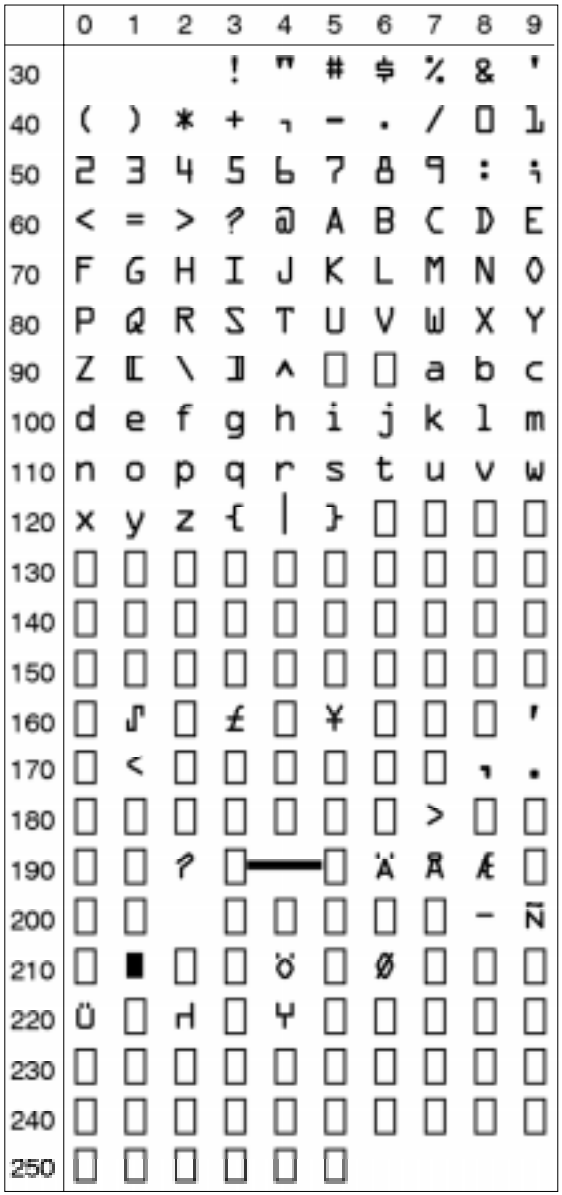
1. Character Sets, cont'd.

ANSI (NASC -2)

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	[	\	]	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	{	|	}	~		□	□
130	‚	*f*	„	…	†	‡	ˆ	‰	Š	‹
140	Œ	□	□	□	□	‘	’	“	”	•
150	–	—	˜	™	š	›	œ	□	□	Ÿ
160		¡	¢	£	¤	¥	¦	§	¨	©
170	ª	«	¬	-	®	¯	°	±	2	3
180	´	µ	¶	·	¸	1	º	»	¼	½
190	¾	¿	À	Á	Â	Ã	Ä	Å	Æ	Ç
200	È	É	Ê	Ë	Ì	Í	Î	Ï	Ð	Ñ
210	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û
220	Ü	Ý	Þ	ß	à	á	â	ã	ä	å
230	æ	ç	è	é	ê	ë	ì	í	î	ï
240	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù
250	ú	û	ü	ý	þ	ÿ				

1. Character Sets, cont'd.

OCR-A BT
(NASC "OCR-A.NSC")



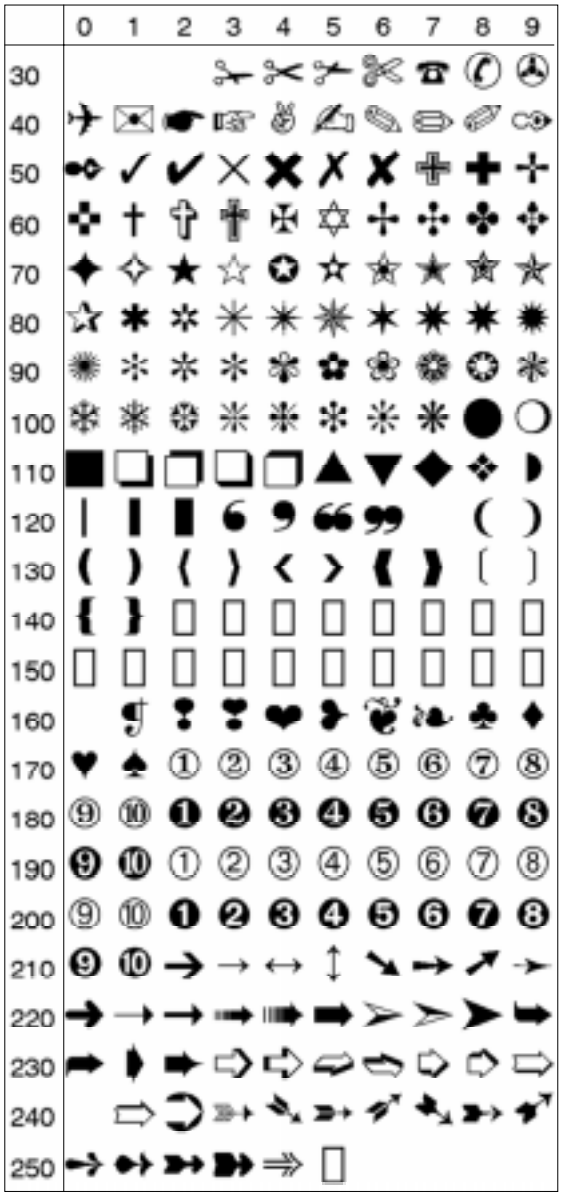
1. Character Sets, cont'd.

OCR-B 10 Pitch BT
(NASC "OCR-B.NSC")

	0	1	2	3	4	5	6	7	8	9
30				!	"	#	$	%	&	'
40	(	)	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	[	\	]	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	{	|	}	~	□	□	□
130	□	□	□	□	□	□	□	□	□	□
140	□	□	□	□	□	□	□	□	□	□
150	□	□	□	□	□	□	□	□	□	□
160		□	□	£	¤	¥	□	§	¨	'
170	"	□	□	□	□	□	□	□	†	□
180	´	m	□	·	¸	□	"	□	□	□
190	□	□	□	'	—	^	Ä	Å	Æ	□
200	□	□		□	□	□	IJ	ij	□	Ñ
210	|	■	_	□	Ö	□	ø	□	□	□
220	Ü	□	□	ß	□	□	□	□	□	å
230	æ	□	□	□	□	□	□	□	□	□
240	□	□	□	□	□	□	□	□	ø	□
250	□	□	□	□	□	□				

1. Character Sets, cont'd.

Zapf Dingbats BT
(NASC "ZAPF.NSC")



2. Resident Fonts

Century Schoolbook BT	THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG the quick brown fox jumps over the lazy dog
Dutch 801 Roman BT	THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG the quick brown fox jumps over the lazy dog
Dutch 801 Bold BT	**THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG** **the quick brown fox jumps over the lazy dog**
Futura Light BT	THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG the quick brown fox jumps over the lazy dog
Letter Gothic 12 Pitch BT	THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG the quick brown fox jumps over the lazy dog
Monospace 821 BT	THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG the quick brown fox jumps over the lazy dog
Monospace 821 Bold BT	**THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG** **the quick brown fox jumps over the lazy dog**
OCR-A BT	THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG the quick brown fox jumps over the lazy dog
OCR-B 10 Pitch BT	THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG the quick brown fox jumps over the lazy dog
Prestige 12 Pitch Bold BT	**THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG** **the quick brown fox jumps over the lazy dog**
Swiss 721 BT (default font)	THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG the quick brown fox jumps over the lazy dog
Swiss 721 Bold BT	**THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG** **the quick brown fox jumps over the lazy dog**
Swiss 721 Bold Condensed BT	**THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG** **the quick brown fox jumps over the lazy dog**
Zapf Dingbats BT	✶★✣ ✱✳☆✣☆ ✢✲✭✷✵ ✦✭✸ ✪✳✵✭✳ ✭✷✣✲ ✶★✣ ✵✡✺✹ ▼✻❋ ❑◆❉✻❋ ❂❒❑◗■ ❄❒❘ ❊◆❍❒▲ ❑❖✻❒ ▼✻❋ ●❁▮▮ ❄❑✻
Zurich Extra Condensed BT	THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG the quick brown fox jumps over the lazy dog

3. Resident Bar Code Fonts

Bar Code	Abbreviation
Codabar	"CODABAR"
Code 11	"CODE11"
Code 16K	"CODE16K"
Code 39	"CODE39"
Code 39 full ASCII	"CODE39A"
Code 39 w. checksum	"CODE39C"
Code 49	"CODE49"
Code 93	"CODE93"
Code 128	"CODE128"
DUN-14/16	"DUN"
EAN-8	"EAN8"
EAN-13	"EAN13"
EAN-128	"EAN128"
Five-Character Supplemental Code	"ADDON5"
Industrial 2 of 5	"C2OF5IND"
Industrial 2 of 5 w. checksum	"C2OF5INDC"
Interleaved 2 of 5	"INT2OF5"
Interleaved 2 of 5 w. checksum	"INT2OF5C"
Interleaved 2 of 5 A	"I2OF5A"
Matrix 2 of 5	"C2OF5MAT"
Maxicode	"MAXICODE"
MSI (modified Plessey)	"MSI"
PDF 417	"PDF417"
Plessey	"PLESSEY"
Straight 2 of 5	"C2OF5"
Two-Character Supplemental Code	"ADDON2"
UCC-128 Serial Shipping Container Code	"UCC128"
UPC-5 digits Add-On Code	"SCCADDON"
UPC-B	"UPCB"
UPC-A	"UPCA"
UPC-D1	"UPCD1"
UPC-D2	"UPCD2"
UPC-D3	"UPCD3"
UPC-D4	"UPCD4"
UPC-D5	"UPCD5"
UPC-E	"UPCE"
UPC Shipping Container Code	"UPCSCC"

Error Messages

1. Error Messages in Numerical Order

Code	Description
0	No error
1	Syntax error.
2	Unbalanced parenthesis.
3	Feature not implemented.
5	Unrecognized token.
9	Undefined token.
12	Type mismatch.
15	Font not found.
17	Bar code type not implemented.
18	Disk full.
19	Error in file name.
24	Overflow in temporary string buffer.
25	Wrong number of parameters.
26	Parameter too large.
27	Parameter too small.
30	Assign to a read-only variable.
34	File is not open.
37	Cutter device not found.
38	User break.
41	Parameter out of range.
43	Memory overflow.
58	Field overflow.
67	Error from communication channel.
78	Not allowed in execution mode.
79	Not allowed in a layout.
1001	Not implemented.
1003	Field out of label.
1005	Out of paper.
1009	Invalid parameter.
1010	Hardware error.
1011	I/O error.
1013	Device not found.
1015	File is read-only.
1018	Bad file descriptor.
1019	Invalid font.
1022	Head lifted.
1023	Incomplete label.
1025	File does not exist.

1. Error Messages in Numerical Order, cont'd.

Code	Description
1027	Out of transfer ribbon.
1035	File pointer is not inside the file.
1037	No acknowledge received within specified timeout.
1041	Error in fos structure.
1042	Internal error in mcs.
1054	Error when trying to write to device.
1055	Error when trying to read from device.
1057	File exists.
1058	Transfer ribbon fitted.
1061	Wrong type of media.
1101	Illegal character in bar code.
1102	Illegal bar code font.
1103	Too many characters in bar code.
1104	Bar code too large.
1105	Bar code parameter error.
1106	Wrong number of characters.
1107	Illegal bar code size.

Commands and Functions

1. Commands (Alphabetical)

Instruction	Chapter	Purpose
ALIGN (AN)	4.2	Specifies which part of a text, bar code field, image field, line or box will be positioned at the insertion point.
BARFONT (BF)	4.4	Specifies a single-byte character set font for printing human readable bar code interpretation. The settings are: start parameter; font name; font size in points; character slant in degrees, with positive values slanting the characters to the right; vertical offset between the barcode and the interpretation; width magnification; height magnification; enable/disable the printing of barcode interpretations.
BARFONT (BF) ON/OFF	4.4	Enables/disables the printing of bar code interpretations.
BARFONTD (BFD)	9.2	Specifies a double-byte character set font for printing human readable bar code interpretation. The settings are: font name; font size in points; character slant in degrees.
BARFONTDSIZE (BFSD)	9.2	Specifies the font size of a double-byte character set in points for printing human readable bar code interpretation.
BARFONTDSLANT (BFLD)	9.2	Specifies the slant of a double-byte character set font in degrees for printing human readable bar code interpretation.
BARFONTSIZE (BFS)	9.2	Specifies the font size for a single-byte character set font in points for printing human readable bar code interpretation.
BARFONTSLANT (BFL)	9.2	Specifies the slant of a single-byte character set font in degrees for printing human readable bar code interpretation.
BARHEIGHT (BH)	4.4	Specifies the height of a bar code in dots.
BARMAG (BM)	4.4	Specifies the magnification with regard to the width of the bars in a bar code.
BARRATIO (BR)	4.4	Specifies the width ratio between wide and narrow bars in a bar code.
BARSET	4.4, 9.1	Specifies a bar code type and sets parameters for complex bar codes
BARTYPE (BT)	4.4	Specifies the type of bar code.
BREAK	6.16	Specifies a break interrupt character separately for the parallel and serial communication channels.
BREAK ON/OFF	6.16	Enables/disables break interrupt separately for the parallel and serial communication channels.
CHR$ (n)	9.2	Returns the ASCII character whose number is given in the brackets.

1. Commands (Alphabetical), cont'd.

Instruction	Chapter	Purpose
CLEANFEED	5.1	Runs the printer's feed mechanism, for the distance specified in dots to allow cleaning.
CLL	5.3	Completely clears the print image buffer, if no starting field is specified. If a field is specified, the print image buffer is cleared from the field to the end of the label.
COPY	8.3	Copies files.
COUNT&	6.6	Creates a counter. Alpha counters range from A to Z, but numeric counters have no practical limit.
DATE$	4.9, 6.3, 7.2	Sets or returns the current date, optionally in the format specified by the FORMAT DATE$ command.
DATEADD$	4.9	Returns a new date after a number of days has been added to, or subtracted from, the current date or optionally a specified date. The current date is not changed.
DIR	4.2	Specifies the print direction.
ERROR	6.15	Defines error messages and enables the error handler for specified error conditions.
FIELDNO	5.3	Gets the current field number for partial clearing of the print buffer by a **CLL** command.
FILE& LOAD	8.4	Receives and stores binary files in the printer's RAM memory.
FILES	8.1	Lists the files stored in one of the printer's directories to the standard OUT channel. RAM: is the default if no other drive is specified.
FONTD (FD)	9.2	Selects a double-byte character set font for the printing of subsequent **PRTXT** commands, and optionally sets the font size and slant.
FONTDSIZE (FSD)	9.2	Sets the size in points of the current double-byte character set font.
FONTDSLANT (FLD)	9.2	Sets the slant in degrees of the current double-byte character set font.
FONT (FT)	4.3	Selects a single-byte character set font for the printing of subsequent **PRTXT** commands, and optionally sets the font size and slant.
FONTSIZE (FS)	4.3	Sets the size in points of the current single-byte character set font.
FONTSLANT (FL)	4.3	Sets the slant in degrees of the current single-byte character set font.
FONTS	8.1	Returns the names of all fonts stored in the printer's memory to the standard OUT channel.
FORMAT	6.9	Formats the printer's RAM memory, or formats a RAM-type memory card to MS-DOS format.

1. Commands (Alphabetical), cont'd.

Instruction	Chapter	Purpose
FORMAT DATE$	6.4	Specifies the format of the string returned by **DATE$("F")** and **DATEADD$(...,"F")** commands.
FORMAT INPUT	6.5	Specifies separators for layout variable data entry.
FORMAT TIME$	6.4	Specifies the format of the string returned by **TIME$("F")** and **TIMEADD$(...,"F")** commands.
FORMFEED (FF)	5.1	Activates the paper feed mechanism in order to feed out or pull back a certain length of the paper web. If the feed length is not specified, the printer feeds a blank label.
FRE	8.1	Returns the number of free bytes in the printer's RAM memory.
FUNCTEST$	7.3, 7.4	Performs various hardware tests on the ROM, RAM, any cards fitted and on the printhead.
HEAD	7.4	Returns the result of a thermal printhead check.
IMAGE LOAD	8.5	Receives and stores image files in .PCX format either in the file system or in volatile memory.
IMAGES	8.1	Returns the names of all images stored in the printer's memory to the standard OUT channel.
INPUT ON	6.1	Sets **SYSVAR(18)** to 0 and enables layouts and use of variable data fields.
INPUT OFF	6.1	Restores **SYSVAR(18)** and disables layouts and use of variable data fields.
INVIMAGE (II)	4.3, 4.5	Inverts the printing of text and images from "black-on-white" to "white-on-black.
KILL	8.2	Deletes a file from the printer's RAM memory or from a DOS-formatted memory card.
LAYOUT END	4.8	Stops the recording of a layout description and saves the layout.
LAYOUT INPUT	4.8	Starts the recording of a layout description.
LAYOUT RUN	4.9	Enables a pre-defined layout.
LTS& ON/OFF	6.7	Enables or disables the label taken sensor.
MAG	4.3, 4.5	Magnifies a font, barfont or image up to four times separately with regard to height and width.
MAP	6.10	Changes the ASCII value of a character when received on the standard IN channel, or optionally on another specified communication channel.
NAME DATE$	6.4	Enters the preferred month name in return strings of **DATE$("F")** and **DATEADD$(...,"F")**.
NAME WEEKDAY$	6.4	Enters the preferred weekday name in return strings of **WEEKDAY$**.
NASC	6.11, 9.2	Selects a national single-byte character set.

1. Commands (Alphabetical), cont'd.

Instruction	Chapter	Purpose
NASCD	9.2	Selects a mapping method for large character set fonts.
NORIMAGE (NI)	4.3, 4.5	Returns to normal printing after an **INVIMAGE** statement has been issued.
PRBAR (PB)	4.9	Prints a bar code.
PRBOX (PX)	4.6	Prints a box.
PRESCALE (PS)	9.3	Scales the characters listed in the accompanying text string using the current font settings.
PRIMAGE (PM)	4.5, 4.9	Prints an image stored in the printer's memory.
PRINT (?)	7.1	Prints data to the standard OUT channel.
PRINT KEY ON	6.8	Enables printing of the current label when the **Feed** key is pressed.
PRINT KEY OFF	6.8	Enables feeding of a blank label when the **Feed** key is pressed.
PRINTFEED (PF)	5.2	Prints and feeds out one or a specified number of labels, tickets, tags or portions of strip, according to the printer's setup.
PRLINE (PL)	4.7	Draws a solid rectangle.
PRPOS (PP)	4.2	Specifes the insertion point for a line of text, a bar code, an image, a box or a line.
PRSTAT	7.7	Returns the printer's current status.
PRTXT (PT)	4.9	Provides the input data for a text field, i.e. a line of text.
REBOOT	6.12	Restarts the printer.
REMOVE IMAGE	8.2	Removes a specified image from the printer's memory.
SETSTDIO	6.2	Forces selection of standard IN and OUT communication channels (normally automatic).
SETUP	6.17	Changes the setup by means of a setup string or setup file.
SETUP WRITE	6.17	Creates a setup file containing the printer's current setup values.
SYSVAR	6.13-14, 7.4-6	Reads or sets various system variables.
TESTFEED	5.1	Slowly feeds paper to allow the label stop sensor to adjust itself according to the presently loaded paper web.
TIME$	4.9, 6.3, 7.2	Sets or returns the current time.
TIMEADD$	4.9	Returns a new time after a number of seconds have been added to, or subtracted from, the current time or optionally a specified time.
VERSION$	7.8	Returns the version of the software, printer family, or type of CPU board.
WEEKDAY$	4.9	Returns the name of the weekday for a specified date.
WEEKNUMBER	4.9	Returns the number of the week for a specified date.

2. Command Syntax

```
ALIGN <anchor point>
BARFONT [#<starting parameter number>,]"fontname"[,size [,slant[,offset[,hmag[,vmag] ] ] ] ]
[ON|OFF]
BARFONT ON | OFF
BARFONTD "fontname"[,size[,slant]]
BARFONTDSIZE <size>
BARFONTDSLANT <slant>
BARFONTSIZE <size>
BARFONTSLANT <slant>
BARHEIGHT <height>
BARMAG <magnification>
BARRATIO <ratio>
BARSET [#<starting parameter number>"codename",<ratio wide bars>,<ratio narrow
       bars>,<mag>,<height>
BARTYPE "<bar code name>"
BREAK <device>, <break character>
BREAK <device> ON | OFF
CLEANFEED <length in dots>
CLL
CLL [<variable name>%]
COPY "[RAM:|CARD1:|CARD2:]<original filename>", "[RAM:|CARD1:|CARD2:]<new filename>"
COUNT& "START", <counter number>, "<start value>"
COUNT& "WIDTH", <counter number>, "<number of digits>"
COUNT& "COPY", <counter number>, "<number of copies>"
COUNT& "INC", <counter number>, "<incr. value>|<decr. value>"
COUNT& "STOP", <counter number>, "<stop value>"
COUNT& "RESTART", <counter number>, "<restart value>"
CUT
DATE$ = "<YYMMDD>"
DIR <direction>
ERROR <error number>, "<error message>"
<variable name>%=FIELDNO
FILE& LOAD "<filename>", <file size>
FILES ["ROM:|RAM:|CARD1:|CARD2:"]
FONTD "<fontname>"[, <size> [, <slant>]]
FONTDSIZE <size>
FONTDSLANT <slant>
FONT "<fontname>"[, <size> [, <slant>]]
FONTSIZE <size>
FONTSLANT <slant>
FONTS
FORMAT "<device>"[,<no of entries>[.<no of bytes>]]
```

2. Command Syntax, cont'd.

```
FORMAT DATE$ <string>
FORMAT INPUT "<start separator>","<end separator>","<field separator>","<start of message
             character>","<end of message character>"
FORMAT TIME$ "<string>"
FORMFEED [<feed length in dots>]
IMAGE LOAD "<imagename>",<filesize>,"<flag>"
IMAGES
INPUT ON | OFF
INVIMAGE
KILL "<filename>|CARD1:<filename>|CARD2:<filename>"
LAYOUT END
LAYOUT INPUT "<layout name>"
LAYOUT RUN "<layout name>"
LTS& ON | OFF
MAG <height mag>, <width mag>
MAP [<device>], <original ASCII value>, <desired ASCII value>
NAME DATE$ <no of month>, "<name of month>"
NAME WEEKDAY$ <no of weekday>, "<name of weekday>"
NASC <character set number>|"<filename of character set>"
NASCD "<filename of character set>"
NORIMAGE
PRBAR <input data>[;<input data>]
PRBOX <height>, <width>, <line thickness>
PRESCALE "<string>"
PRIMAGE "<image name>"
PRINT "<input data>"[;"<input data>"]
PRINT KEY ON
PRINT KEY OFF
PRINTFEED [<BATCH SIZE>]
PRLINE <length>, <line thickness>
PRPOS <x coordinate>, <y coordinate>
PRTXT "<input data>"[;"<input data>"]
REBOOT
REMOVE IMAGE "<name>"
SETSTDIO <In channel>, <OUT channel>
SETUP "<setup string>"|"<setup filename>"
SETUP WRITE "<setup filename>"
SYSVAR (<parameter>)= <number>
TESTFEED
TIME$ = "<HHMMSS>"
```

3. Function Syntax

Funtions cannot stand alone: they must be used in conjunction with a command, e.g. **PRINT**, **PRTXT**.

```
CHR$ (<ASCII character number>)
DATE$ ("F")
DATEADD$ (["<original date>",] <number of days>[,"F"])
FRE(1)
FUNCTEST$ ("ROM1:|ROM2:|RAM:|CARD1:|CARD2:|head")
HEAD (<type of check>)
PRSTAT
PRSTAT AND <parameter>
SYSVAR <parameter>
TIME$ [("F")]
TIMEADD$ (["<original time>",]<no. of seconds>[,"F"])
VERSION$ [(type of info)]
WEEKDAY$ ("<date>")
WEEKNUMBER ("<date>")
```