PCL Barcode Flash 3.0

User Manual

KYOCERA

This manual, revision 3.2 (03/2013) is based on PCL Barcode Flash version 3.02.17. The software version is published as the 'Application Version' on the PCL Barcode Flash status page as well.

User Manual

PCL Barcode Flash

Please read this manual before installing and using the software. Keep the manual somewhere close to the printer.

Contents

1 Introduction

1.1 About this manual

This manual provides information on how to install and use the PCL Barcode Flash software for Kyocera printers and multifunction devices.

Instructions on the installation and configuration of the *devices* can be found in the relevant installation and user manuals. You will also find here additional information on programming in *PRESCRIBE* and *PCL*, which is not included in this manual.

If you have any queries on or problems with your PCL Barcode Flash solution, which are not resolved in this manual, please consult your service partner, the local Kyocera Document Solutions Hotline or your nearest Kyocera Document Solutions office.

1.2 How to use this manual

This manual contains information on how to use PCL Barcode Flash, and in particular information on installing and programming it. Below, we refer to the "barcode solution" for short.

The performance features and programming instructions described are illustrated with examples and – where necessary – printed barcode samples.

This manual does not claim to provide an introduction to the world of the barcode or to programming in PRESCRIBE or PCL. The following chapters contain information required specifically when using PCL Barcode Flash. The chapter on programming assumes that the reader has a basic knowledge of PCL or PRESCRIBE and a knowledge of how to work with the barcodes and/or matrix codes to be used. Information on programming in PRESCRIBE and PCL can be found in the appendix and in the additional Kyocera Document Solutions manuals included in the delivery (as a PDF file on CD).

In this text, the term "barcodes" is frequently used for the sake of simplicity as a generic term for linear barcodes and matrix codes so as to avoid needlessly long phrases.

If this document refers to a "printer", this also refers to a multifunction device (KM device), as only the "printing" part is important to the barcode solution. A precise list of devices supported can be found in the chapter entitled SUPPORTED DEVICES.

Various fonts are used in this manual in order to indicate information which is identical or similar. The use of these fonts is consistent throughout the entire manual:

Bold – Inputs to be keyed directly on the front panel of the printer are specified in bold type.

Italics – Programs or applications are cited in italics.

SMALL CAPS – Small caps are used to refer to chapters in this manual. For example, details on this can be found in the chapter entitled, HOW TO USE THIS MANUAL.

`Courier font` – Courier is used to refer to programming elements. These may include PRESCRIBE sequences or commands on the CF/SD card.

    ☞   In this manual, the pointing finger refers to tips that provide additional information and as a rule are of a general nature.

    ⚠   The warning symbol is used in this manual to refer to potential sources of errors in operation or hazards.

1.3 Purpose of the application

PCL Barcode Flash enhances Kyocera FS, KM and TASKalfa devices to include additional linear barcodes and matrix codes, as well as providing the option of calling these not only with the PRESCRIBE command "`BARC`", but also with PCL sequences. Furthermore, a number of resident and/or machine-readable fonts have been added.

As the PCL activation sequences follow industry-standard formats, there are usually no problems with integrating devices equipped with the Barcode Flash solution into existing environments (e.g. SAP), without having to make changes to the controlling software.

1.4 Items provided as standard

Once you have removed the packaging, please ensure that you have the following components:

- a CD containing manual files (in PDF format) and sample prints

- a Compact Flash/Secure Digital card (pre-programmed with PCL Barcode Flash software)

1.5 About PCL Barcode Flash

1.5.1 Overview

PCL Barcode Flash uses a special feature of Kyocera FS, KM and TASKalfa devices: the central unit – consisting of the PowerPC processor, motherboard and memory – normally used for calculation of the print job can also be used for programming purposes via a separate interface, the Application Programming Interface (API). In addition to simple tasks such as filtering and changing the incoming data stream, complex solutions are possible, in this instance the generation of complete barcodes.

The API program is located on the Compact Flash card and is automatically loaded into memory and activated when the printer is switched on. Practically all normal functions of the device are retained alongside the barcode solution; for example, the device can still be used as a completely normal workgroup printer.

The barcodes already available as standard with Kyocera devices can still be called using the `BARC` command. In addition, the activation option is now available via the general industry-standard PCL command, thus allowing these devices to be integrated with ease into existing systems. Thus costly work on the controlling application (e.g. in inventory software) can be avoided.

PCL Barcode Flash enhances the standard barcodes with a range of additional linear codes and matrix codes as well as machine-readable fonts and formatting options, so all current standard applications are covered to a great extent.

All activation controls can be undertaken in PRESCRIBE or PCL as required, giving the programmer the greatest possible flexibility. The PRESCRIBE `BARC` command has been extended so as to allow Data Matrix, QR Code and Australia Post 4-State, for example, to be controlled in the usual way.

The PCL sequences for controlling the barcodes have been borrowed from font control and are described in detail in the chapter entitled, "PROGRAMMING".

The PRESCRIBE sequences for barcodes already available as standard are described in detail in the PRESCRIBE IIe Programming Manual (see appendix).

1.5.2 Terminology

In this manual, in most cases we simply refer to "barcodes". In technical terminology, we refer to linear codes, stack codes and matrix codes, which differ in terms of what is known as "symbology" (= graphic structure).

The linear codes are the classic "barcodes" made up of parallel lines in which the information is arranged in one direction. For this reason, these are often referred to as one-dimensional ("1D") codes. These codes are very simple and quick to read, and because of this they are the most commonly used codes in the world.

Stack codes and matrix codes belong to the group of two-dimensional ("2D") codes, where the information is arranged not only horizontally, but also vertically.

Stack codes essentially consist of up to 90 rather flat linear codes "stacked one on top of the other", so that a data density several times higher can be generated without allowing the dimensions of the codes to become excessively large. Normally, data densities fifteen times higher are achieved than in the case of simple linear codes.

Matrix codes also have the advantage of high data density and to some extent offer enormous data security, but they use a completely independent symbology made up of square elements, segments of circles, stripes or hexagons of different sizes, or combinations of these. The development history and structure of these codes have been influenced by various demands, such as high data density, fast readability, great data security or the most well-featured codable character set possible. The various symbologies have developed due to optimisations for specific purposes.

The read technology for 2D codes is clearly more expensive and requires special readers, the structure of which is partly equivalent to that of a video camera. Matrix codes in particular are becoming established more and more in trade and industry due to the high information density and great resistance to interference. Properties and advantages

1.6 The main features of PCL Barcode Flash

■ Simple installation and operation

■ Compatibility with industry-standard activation sequences

■ Large number of supported 1D and 2D codes, including QR Code and Datamatrix

■ Compatible with every model in the list of devices supported, no model-dependent versions required

■ PRESCRIBE and PCL commands supported

■ No relevant influence on device speed

■ Demo version without time limit available which can be used to test compatibility with existing (print) solutions prior to purchase

1.7 Supported Barcodes

1.7.1 Linear codes (one-dimensional codes)

2/5 Interleaved with check digit
2/5 Interleaved without check digit
2/5 Industrial with check digit
2/5 Industrial without check digit
2/5 Matrix with check digit
2/5 Matrix without check digit
Code 93
Code 93 extended
Code 128 autoswitch
Code 128 Subset A
Code 128 Subset B
Code 128 Subset C
Code 39 (3 of 9)
Code 39 with Mod47 check digit
Code 39 extended
Code 39 extended with Mod47 check digit
Codabar without check digit
Codabar with check digit
EAN-128
EANJAN-8
EANJAN-8 + 2-character extension
EANJAN-8 + 5-character extension
EANJAN-13
EANJAN-13 + 2-character extension
EANJAN-13 + 5-character extension

UCC-128
UPC-A
UPC-A + 2-character extension
UPC-E
UPC-E + 2-character extension
MSI Plessey
MSI Plessey + Mod10 check digit
MSI Plessey + Mod10 + Mod10 check digit
MSI Plessey + Mod11 + Mod10 check digit
ZIP+4 Postnet 5
ZIP+4 Postnet 9
ZIP+4 Postnet 11
Australia 4-State 37-CUST
Australia 4-State 52-FF-MET
Australia 4-State 67-FF-MET
Australia 4-State FCC-45 REPLY
Singapore Post 4-State
USPS Intelligent Mail Barcode

1.7.2 Matrix codes (two-dimensional codes)

Datamatrix
QR Code
Maxicode
PDF417

1.8 Supported Devices

PCL Barcode Flash supports Kyocera printers and multifunction devices. At the time of publishing (version 3.02.17, 03-2013), the following devices are supported:

Multifunction Devices (alphabetical order)

Model Name	Card Type
FS-1016MFP	-
FS-1018MFP	-
FS-1020MFP	-
FS-1024MFP	-
FS-1025MFP	-
FS-1028MFP/KM-2810	B/C
FS-1030MFP	B/C
FS-1035MFP	B/C
FS-1116MFP	-
FS-1118MFP/KM-1820	-
FS-1120MFP	-
FS-1124MFP	-
FS-1125MFP	-
FS-1128MFP/KM-2820	B/C
FS-1130MFP	B/C
FS-1135MFP	B/C
FS-1220MFP	-
FS-1320MFP	-
FS-1325MFP	-
FS-3040MFP	B/C
FS-3040MFP+	B/C
FS-3140MFP	B/C
FS-3140MFP+	B/C
FS-3540MFP	B/C
FS-3640MFP	B/C
FS-6025MFP	B/C
FS-6030MFP	B/C
FS-6525MFP	B/C
FS-6530MFP	B/C

FS-C1020MFP	-
FS-C1020MFP+	-
FS-C2026MFP	B/C
FS-C2026MFP+	B/C
FS-C2126MFP	B/C
FS-C2126MFP+	B/C
FS-C2526MFP	B/C
FS-C2626MFP	B/C
FS-C8020MFP	B/C
FS-C8025MFP	B/C
FS-C8520MFP	B/C
FS-C8525MFP	B/C
KM-1500	-
KM-1620	-
KM-1635	-
KM-1650	A
KM-2020	-
KM-2035	-
KM-2050	A
KM-2530	A
KM-2540	A
KM-2550	A
KM-2560	B/C
KM-3035	A
KM-3040	A
KM-3050	B/C
KM-3060	B/C
KM-3530	A
KM-3650w	-
KM-4030	A
KM-4035	A
KM-4050	B/C
KM-4230	A
KM-4530	A
KM-4800w	-

KM-4850w	-
KM-5035	A
KM-5050	B/C
KM-5230	A
KM-5530	A
KM-6030	A
KM-6230	-
KM-6330	A
KM-7530	A
KM-8030	A
KM-C2520	A
KM-C2525E	A
KM-C2630	A
KM-C3225	A
KM-C3232	A
KM-C3232E	A
KM-C4035E	A
KM-C830	A
KM-C850	A
KM-P4845w	-
KM-P4850w	-
TASKalfa 180	-
TASKalfa 181	A
TASKalfa 220	-
TASKalfa 221	A
TASKalfa 250ci	B/C
TASKalfa 2550ci	B/C
TASKalfa 300ci	B/C
TASKalfa 300i	B/C
TASKalfa 3050ci	B/C
TASKalfa 3500i	B/C
TASKalfa 3550ci	B/C
TASKalfa 400ci	B/C
TASKalfa 420i	B/C
TASKalfa 4500i	B/C

TASKalfa 4550ci	**B/C**
TASKalfa 500ci	**B/C**
TASKalfa 520i	**B/C**
TASKalfa 5500i	**B/C**
TASKalfa 550c	-
TASKalfa 552ci	**B/C**
TASKalfa 5550ci	**B/C**
TASKalfa 620	**A**
TASKalfa 6500i	**B/C**
TASKalfa 650c	-
TASKalfa 6550ci	**B/C**
TASKalfa 750c	-
TASKalfa 7550ci	**B/C**
TASKalfa 8000i	**B/C**
TASKalfa 820	**A**

Printers (alphabetical order)

Model Name	Card Type
FS-1010	**A**
FS-1020D	**A**
FS-1030D	**A**
FS-1040	-
FS-1041	-
FS-1050	**A**
FS-1060DN	-
FS-1061DN	-
FS-1100	**A**
FS-1110	-
FS-1120D	**A**
FS-1300D	**A**
FS-1320D	**A**
FS-1350DN	**B/C**
FS-1370DN	**B/C**
FS-1800	**A**
FS-1800+	**A**
FS-1900	**A**
FS-1920	**A**

FS-2000D	A
FS-2020D	B/C
FS-2100D/DN	D
FS-3800	A
FS-3820N	A
FS-3830N	A
FS-3900DN	A
FS-3920DN	B/C
FS-4000DN	A
FS-4020DN	B/C
FS-4100DN	D
FS-4200DN	D
FS-4300DN	D
FS-6020	A
FS-6950DN	A
FS-6970DN	B/C
FS-7000	A
FS-820	A
FS-9000	A
FS-9100DN	A
FS-9120DN	A
FS-9130DN	A
FS-920	A
FS-9500DN	A
FS-9520DN	A
FS-9530DN	A
FS-C5015N	A
FS-C5016N	A
FS-C5020N	A
FS-C5025N	A
FS-C5030N	A
FS-C5100DN	B/C
FS-C5150DN	B/C
FS-C5150DN	B/C
FS-C5200DN	B/C
FS-C5250DN	B/C
FS-C5250DN	B/C
FS-C5300DN	B/C
FS-C5350DN	B/C

FS-C5400DN	**B/C**
FS-C8008N	**A**
FS-C8026N	**A**
FS-C8100DN	**A**
FS-C8100DN+	**A**
FS-C8500DN	**A**
FS-C8600DN	**B/C**
FS-C8650DN	**B/C**
KM-4800w	**-**
TASKalfa 2420w	**-**
TASKalfa 4820w	**-**

A   **=**   **Compatible with card type A (Compact Flash card)**

B/C   **=**   **Compatible with card type B/C (Compact Flash card)**

D   **=**   **Compatible with card type D (Secure Digital card)**

-   **=**   **Not applicable / not supported. Solution will not run on these devices.**

An up-to-date compatibility list is available in the service area of www.kyoceradocumentsolutions.eu

☞ **Note**
Please note that some KM series multifunction devices have to be fitted with a printer board first

☞ **Note**
For some of the older supported devices, it might in certain cases be necessary to make a firmware update.

☞ **Note**
PCL Barcode Flash assumes that you have a free CF/SD card slot in your printer or multifunction device.

☞ **Note**
The following minimum requirements for the device used should be met so that PCL Barcode Flash can run without complications:

- Firmware as of April 2001 or newer

- Min. 16 MB RAM recommended. Less RAM may lead to errors.

1.9 Emulations and restrictions supported

When PCL Barcode Flash software was developed, we attached a lot of importance to not making substantial changes to the standard functionality of the devices equipped with it.

In most cases, the appropriate printer or multifunction device can be used in future with no changes, although of course it will have more functions on account of the barcode solution.

In the following cases, it may be necessary to adapt the barcode solution or the printer settings/printer environment:

- PCL Barcode Flash supports printer control languages PCL and PRESCRIBE only. No barcodes can be controlled from within other emulations (e.g. KPDL, Line Printer, etc.) if they are activated.

- For technical reasons, it is not possible to use a number of API programs – the barcode solution is one of these – simultaneously. Therefore, solutions produced by Kyocera development partners, e.g. for data filtering or data encryption, cannot be run at the same time as PCL Barcode Flash.

- No barcode activations can be made from PRESCRIBE macros because a parameter passing into the macro from "outside" is not possible.

- Forms and macros based on PCL and PRESCRIBE, which are already stored on a CF/SD card can continue to be used and controlled as normal. This data has to be copied into the "Startup" partition on the PCL Barcode Flash CF/SD card so that it is automatically loaded into the RAM when the printer is switched on.

1.10 Barcode compatibility

PCL Barcode Flash has been designed to comply with industry-standard barcode activations as far as is possible. As there is no specific standardisation here more or less equivalent to the ISO or DIN standards, the software should be tested in the existing system environments prior to each use to see whether the desired barcode is generated with any data streams already preset.

An unlimited demo version is available for testing purposes, which can be used to establish the functionality of PCL Barcode Flash before buying it.

2 Installation and commissioning

2.1 Hardware installation

The software is supplied on a Compact Flash/Secure Digital card ('CF/SD card'). To use this card, it has to be kept permanently in the printer. All you have to do to install the card is to insert it into the slot in the printer.

Note that, for technical reasons, starting with Version 3.00 the Barcode Flash solution is available as Type A and Type B/C. *Please see the compatibility list to find the right version for your device.*

Where differences exist between the installation of Type A and Type B/C, these are described below. Please ensure that the proper installation is available on the compact flash card. You can use the directory to determine whether this is the case and reprogram the card if necessary. You can do so by copying the right file to the printer. Please observe the instructions in this regard at the beginning of the update file.

Please nothe that only the PCL Barcode Flash cards can be re-programmed by the end user. The KYOmulticode card (PCL Barcode Flash with Unicode font extension and additional fonts) has to be re-programmed by Kyocera customer service if a change between Type A and B/C or vice versa is necessary.

☞ Note
Please note carefully the general tips on working with CF/SD cards in your printer manual. This manual will also contain a description of where the CF/SD slot is on your particular device.

Data and programs for Kyocera printers always have to be present in a particular form on Compact Flash cards. In the case of the barcode solution, these are a number of what are known as 'partitions' in which additional fonts and macros for the status page and function calls are stored for the user in addition to the software.

The following partitions are available once the CF/SD card has been inserted:

Type A:

```
Barcode Status
Barcode Init
Barcode Remove
KM BC 3.02.1
BSTP
STP1
STP2
KM_Logo
Linedraw
Code39_4
Code39_8
EAN_10mil
```

```
EAN_13mil
LetterGothic_10
LetterGothic_12
LetterGothic_17
USPS_ZIP
OCR_A
OCR_B
OCR_B scal
Startup
ocr_a_9_land
ocr_a_10_land
ocr_a_9_port
ocr_a_10_port
ocr_b_12_ext_la
ocr_b_12_land
ocr_b_12_ext_po
ocr_b_12_port
```

Type B/C:

```
/PRE/kmbc3-02-1
/PRE/kmbc3-02-1.inf
/PRE/autostart
/PRE/bstp
/PRE/stp1
/PRE/stp2
/PRE/stp3
/PRE/KM_Logo.ttf
/PRE/code39_8
/PRE/ean10mil
/PRE/ean13mil
/PRE/linedraw
/PRE/ocra
/PRE/ocrb731
/PRE/ocrb732
/PRE/ocrb733
/PRE/ocrb734
/PRE/ocrb735
/PRE/ocrb736
/PRE/ocrb737
/PRE/ocrb738
/PRE/code39_4
/PRE/OCR-B_scal
/PRE/uspszip
/PRE/ocrb
/PRE/ltgot10a
/PRE/ltgot12a
/PRE/ltgot17a
```

Type D:

```
KMBC3-02-17.so
KMBC3-02-17.so.INF
autostart
BSTP
STP1
STP2
STP3
KM_Logo.ttf
Let_Got_10
Let_Got_12
Let_Got_17
Linedraw
OCR-B_scal
OCRA731
OCRA732
OCRA733
OCRA734
OCRB735
OCRB736
OCRB737
OCRB738
OCR_A
OCR_B
Code39_4
Code39_8
EAN_10mil
EAN_13mil
USPS_ZIP
```

If required, you can copy additional macro and font partitions to the CF/SD card without impairing the functionality of PCL Barcode Flash.

To see an overview of the partitions, print a partition list. Start printing this list using the front panel on the printer.

Example of a partition list of Type A:

The *Device Information* contains information on the physical size of the CF/SD card and the memory used. The *Partition Information* lists print data, macro data and font data which is stored on the card alongside PCL Barcode Flash.

This information is technical in nature and the actual figures may vary depending on the software version.

2.2 Activating and deactivating the software (Type A only)

Before you can use the software on the CF card in your printer, you have to make this available to the device by activating it.

To do this, you have two options:

- Call the partition `Barcode Init` via the front panel on your printer,

or

- Send the PRESCRIBE sequence

  `!R! FRPO I0, "KM BC 3.02.1"; EXIT;`

  to the printer. If you are using a later version, the name of the newer partition must be used instead of "KM BC 3.02.1". To see the exact name of the program partition, print a partition list.

Automatic loading of the software on the CF card is now activated, and once the device has been switched on the barcode solution will be immediately available.

☞ **Note**

You have to set your printer to 600 dpi minimum to be able to use the software You can optionally configure your printer via the front panel or the PRESCRIBE sequence
`!R! FRPO N8,1; EXIT;`
.

Depending on how you are using it, it may sometimes be sensible (e.g. for testing purposes) to deactivate the software occasionally. You have two options for doing this:

- Remove the CF card from the device

  or

- Deactivate the CF card by calling the `Barcode Remove` partition

The software will no longer be available the next time you start the printer.

2.3 Status page and test prints (Type A, Type B/C and D)

The status page contains information on fonts, software and the configuration.

When you have installed the software, you can call the `Barcode Status` partition to print the CF card's status page. (Type A only).

You can also print the status page by means of the PRESCRIBE sequence (Type A, Type B/C and Type D)

```
!R! CALL BSTP;EXIT;
```
.

☞ **Note**

If any problems or errors occur, we recommend that you always print a status page. In this way, you can easily check whether the software and all the requisite fonts have been loaded.
When you consult Support, you should always have a current status page to hand. As a rule, the status page helps Support to help you resolve the problem more easily.

The status page is divided into three sections: *Font Information, Internal Barcode Information* and *Application Information*:

Font Information contains an overview of the group of barcodes and fonts which are identified and can be used by means of simple font numbers. Some older applications on the market send activations in this form. If there are no print examples on the right-hand side of the list, the printer is not set to 600 dpi. This standard resolution should be set on the front panel of the device to ensure perfect results.

Internal Barcode Information refers to the codes hard-coded into the printer which form the full barcode solution together with PCL Barcode Flash.

Technical hints on PCL Barcode Flash can be found in the section entitled, *Application Information*: For service enquiries, the most important details are the Application Version, the API Version and the Printer Serial Number. The configuration codes are reserved for internal purposes.

Under *Additional Barcodes*, you will see the barcode groups which have been added to the printer by means of the barcode solution. Please note that as a rule there are a number of variants for each symbology, such as with check sums, which are also considered to be individual codes.

Example of a status page of Type A:

2.4 Startup partition (Type A only)

As described in the chapter entitled, HARDWARE INSTALLATION, there is a Startup partition on the CF/SD card. All data on this partition is automatically loaded when the printer is switched on and is immediately available. Therefore, all additional data which you would like to make available to users must be copied into this partition. As a rule, this data includes fonts, macros or print data.

3 Programming

3.1 General

Kyocera printers and multifunction devices are equipped as standard with a number of linear barcodes which can be controlled exclusively using the Kyocera language PRESCRIBE.

The PCL Barcode Flash software extends the number of barcodes supported to include important new types, which have been established in trade, industry and postal operations. In addition, an important feature is that almost all existing and new barcodes can also be controlled by means of PCL standard commands, which considerably enhances the field of application of printers equipped appropriately.

The following section describes control via PCL commands in detail and includes a number of examples of applications and prints.

Control by means of commands is already described precisely in the Kyocera manual "PRESCRIBE IIE PROGRAMMING MANUAL" (CHAPTER 4: BARCODES), and this is available in both German and English versions (PDF format) on the CD supplied with the product. Therefore, this manual deals only with special features of PRESCRIBE programming where new barcodes have been added.

A description of PDF417 code control can also be found on the CD. In the version of PCL Barcode Flash on which this manual is based, control of PDF417 is possible only via the PRESCRIBE command XBAR.

3.2 Control with PCL

PCL Barcode Flash uses PCL font activation sequences to control barcodes. The following instructions assume a basic knowledge of PCL programming. If you are not yet confident with using this, we recommend that you read the manuals mentioned in the previous chapter in which programming in PCL is looked at in more detail in addition to PRESCRIBE programming.

In particular, two sequences are important, the first one for controlling fonts: `<Esc>(s#T` and the second one for activating Transparent mode: `<Esc>&p#X`.

Below, when we mention `<Esc>` we mean ASCII character 27 decimal or 1B hexadecimal. Number parameters are referred to by # below.

The sequence `<Esc>(s#T` normally changes the font. In other words, the text which follows the sequence is printed from this point on in the font for which you have stated the number. Each font available in the printer has a number which has to be stated in the T parameter. The selected font remains active until the printer receives another `<Esc>(s#T` sequence with another number, then it changes the active font again.

The barcodes and matrix codes in PCL Barcode Flash are also controlled in this way, and each barcode and matrix code is allocated a number, just like every font. These numbers come from a reserved range of numbers in which there are no font numbers, from 24580 to 24900. Please see the following list in this chapter for the equivalent "font numbers" of the barcodes supported individually. If you would like to use one of the barcodes or matrix codes made available by PCL Barcode Flash, change the "font" with the sequence `<Esc>(s#T` so that further data that follows the sequence is interpreted by the printer from this point on as barcode information and no longer as text. This base data, as it is known, does not have to be clearly legible text: any binary data can also be sent, depending on the barcode. Just like fonts, barcodes remain active until the printer receives another `<Esc>(s#T` sequence. In this respect, please see the following chapter as well.

☞ **Note**
> If you would like to use your own fonts (regardless of whether you are using PCL or PRESCRIBE), please do not use any font numbers in the range of 24580 to 24900 as mentioned, as these numbers are reserved for the barcodes in PCL as soon as PCL Barcode Flash is activated. Otherwise this may lead to unwanted printer behaviour.

3.2.1 Stop characters

The base data has to be closed by means of a stop character.

In the case of numeric barcodes, this stop character may be:

`<CR>` (ASCII 13), `<LF>` (ASCII 10), `<FF>` (ASCII 12), `<Esc>` (ASCII 27) , `<Space Character>` (ASCII 32)

In the case of alphanumeric barcodes, this stop character may be:

<CR> (ASCII 13),< LF> (ASCII 10), <FF> (ASCII 12), <Esc> (ASCII 27)

Any barcode which permits only numeric characters according to the standard is valid as a numeric barcode. Barcodes which also permit alphanumeric characters must not be closed with the space character (ASCII 32) as the stop character even if the base data contains only numeric characters. Barcodes, which permit the space character in the base data, are essentially considered to be alphanumeric in this sense.

☞ **Note**

The stop character is interpreted. In other words: if you use <FF> as the stop character, this is interpreted not only as a stop character, but also as a page break. When, for example, the stop character <Esc> is used, this is interpreted not only as a stop character, but also as the start of an escape sequence.

The stop character informs PCL Barcode Flash that the base data collected since the sequence <Esc>(s#T is to be printed as a barcode symbol.

Example:

`<Esc>(s24670T12345<CR>`

In this example, the number 24670 stands for Code 39. Here, <CR> (ASCII 13) is used as the stop character. This sequence prints the character string "12345" as a Code 39 symbol.

3.2.2 Stopping Barcode mode

To stop (deactivate) Barcode mode to enable you to print text again, change the font again. Send once again a <Esc>(s#T sequence to the printer which includes the number of the font that the printer is to use for further text from now on.

Some barcodes allow all 256 characters of the ASCII code, including the characters that are valid as stop characters as described above. Because of this, the appropriate characters cannot, of course, be used as stop characters as they may appear in the middle of the base data, after all. For this reason, in addition to the procedure described above, a second option – "Transparent mode" – is available to you for isolating the base data. If you would like to create a barcode of this type and use the stop characters as base data, add a second Escape sequence to the barcode call sequence. This will indicate how many characters are to be assigned to the barcode symbol.

Example:

`<Esc>(s24820T12345<Esc>&p5X12345`

This example prints the base data "12345" as Datamatrix code. `<Esc>&p5X` makes it understood that the five characters following in the sequence belong to this. Any characters from the complete ASCII character set could have appeared instead of "12345".

3.2.3 Rules for base data

Rules are established for each individual barcode by means of standards which have to be observed when the base data is sent. These include which characters are allowed as base data. Some barcodes also require a specific minimum or maximum number of characters, or an odd or even number of characters, for example. If PCL Barcode Flash encounters incorrect base data, an error message is printed out instead of the barcode symbol.

3.2.4 Quiet zones

All barcodes require a specific quiet zone before and after the printed area (above and below too in the case of two-dimensional codes), and you have to take this into account by means of careful positioning when programming. The quiet zone is a white, unprinted area which the reader needs to be able to recognise the barcode reliably.

Normally, a quiet zone of at least 2.5 mm is required, but at least 6.5 mm is required in the case of applications with very varied or fairly large read distances. As a rule of thumb, the quiet zone should be about ten or fifteen times the width of the narrowest line element in a barcode. To be safe, the quiet zone should always be a little more generous if possible.

3.2.5 Positioning of the cursor

According to PCL rules, characters of a font are printed at the bottom left of the cursor position. This rule is also used in the case of barcode symbols. When a barcode is printed, the cursor moves to the lower right corner of the barcode symbol. See the PCL manual for how to position the cursor.

☞ **Note**
In most cases, it is recommended after producing a barcode symbol in PCL that absolute positioning be carried out.

3.2.6 Further parameters and options

☞ **Note**
The following description of parameters and options applies to one-dimensional barcodes. Different or additional rules apply to some barcodes, primarily two-dimensional ones. These are described in the chapter entitled, "BARCODES AND MATRIX CODES WITH DIFFERENT PARAMETERS". These barcodes are Datamatrix, QR Code and Australia 4-state.

As described in the preceding section, the sequence `<Esc>(s#T` is used. However, this is only the simplified basic form: you have the option of influencing the printout of the barcode by entering further parameters. These are detailed below. All of these parameters are optional, i.e. you can omit these if your application does not require plain text, for example, or if the width and length in default form are already sufficient for the requirements of your application.

Example:

`<Esc>(s1p301h20v4b4s24670T instead of just <Esc>(s24670T`

v parameter
Height of the barcode in dots (1/72 in.)
Values of 0.2 to 999 are permitted.

b parameter
Width of the bars
These values are separated by commas. In the case of a barcode with two bar widths, e.g. Code 39, enter two values, e.g. `<Esc>(s6,12b...`; or in the case of a code with four bar widths, e.g. Code 128, enter four values, e.g. `<Esc>(s6,12,18,24b...`

Values from 2 are permitted. Maximum values: 100 for the narrowest bar, 200 for the second-narrowest, 300 for the second-widest and 400 for the widest.

s parameter
Width of the spaces
These values are separated by commas. In the case of a barcode with two space widths, e.g. Code 39, enter two values, e.g. `<Esc>(s6,12s...`; or in the case of a code with four space widths, e.g. Code 128, enter four values, e.g. `<Esc>(s6,12,18,24s...` Values from 2 are permitted. Maximum values: 100 for the narrowest space, 200 for the second-narrowest, 300 for the second-widest and 400 for the widest.

p parameter
Positioning of plain text
0 = Use default, cf. the defaults entered for the individual codes for this purpose

1 = no plain text

Example: <1p>

2 = embedded beneath

Example: <2p>

3 = half embedded beneath

Example: <3p>

4 = beneath the barcode

Example: <4p>

5 = above the barcode

Example: <5p>

h parameter
Plain text font

When you use the h parameter, this always requires a three-digit number, as you can see from the example above.

First digit: Font style

1 = regular

2 = italics

3 = bold

4 = bold italics

Second digit: Font size

0 = select automatically

1 = use previously selected size for the font selected

Third digit: Typeface

0 = Courier

1 = Letter Gothic

2 = Univers

3 = Univers compressed

4 = CG-Times

5 = OCR-B

Example of <204h>: 2 = italics, 0 = automatic font size, 4 = CG-Times

1234567890123456789 0

T parameter
Number of the barcode

The number of the required barcode can be taken from the following table:

UPC-A	24600
UPC-A + 2-digit extension	24601
UPC-E	24610
UPC-E + 2-digit extension	24611
EAN/JAN-8	24620
EAN/JAN-8 + 2-digit extension	24621
EAN/JAN-8 + 5-digit extension	24622
EAN/JAN-13	24630
EAN/JAN-13 + 2-digit extension	24631
EAN/JAN-13 + 5-digit extension	24632
Interleaved 2 of 5	24640
Interleaved 2 of 5 with check digit	24641
Industrial 2 of 5 (25)	24650
Industrial 2 of 5 (25) with check digit	24651
Matrix 2 of 5	24660
Matrix 2 of 5 with check digit	24661
Code 39 (3 of 9) without space character	24670
Code 39 + Mod47 with check digit without space character	24671
Code 39 (3 of 9) with space character	24672
Code 39 + Mod47 with check digit with space character	24673
Code 39 Extended	24680
Code 39 Extended + Mod47 check digit	24681
Code 93	24690
Code 93 Extended	24691
Code 128 autoswitch	24700
Code 128 Subset A	24701
Code 128 Subset B	24702
Code 128 Subset C	24704
UCC-128	24710
EAN-128	24720
Codabar	24750
Codabar + Mod16 check digit	24751
MSI Plessey	24760
MSI Plessey + Mod10 check digit	24761
MSI Plessey + Mod10 + Mod10 check digit	24762
MSI Plessey + Mod11 + Mod10 check digit	24763
ZIP + 4 Postnet 5	24770
ZIP + 4 Postnet 9	24771
ZIP + 4 Postnet 11	24772
Australia 4-State 37-CUST	24785
Australia 4-State 52-FF-MET	24786
Australia 4-State 67-FF-MET	24787
Australia 4-State FCC-45 REPLY	24788
Singapore Post 4-State	24780
USPS Intelligent Mail Barcode	24790
Datamatrix	24820
QR Code	24861
Maxicode	24800
PDF417	24850

If you inadvertently enter a number in the T parameter which does not appear in the preceding table, the printer regards the complete sequence as a font call and *not* as a barcode call, so any unwanted fonts are activated and the printer attempts to print your base data instead of outputting the barcode.

⚠ **Warning**

If invalid values – i.e. values which differ from the validity ranges mentioned above – are entered for any of the parameters, PCL Barcode Flash prints out an error message instead of the barcode symbol. The same happens when the desired base data cannot be reproduced by means of the selected barcode type.

3.2.7 Alternative Escape characters

There are applications, primarily in the mainframe sphere, from which characters below ASCII 32 (= 20hex) cannot be output, or output only with difficulty. In such cases, you have the option of replacing the <Esc> character (ASCII character 27) with any other character of the list below.
Using the PRESCRIBE sequence

```
!R!VBST;F;EXIT;
```

you can activate a mode in PCL Barcode Flash which uses the character ~ (ASCII 126) as an alternative character for <Esc>. If you have activated this mode, you can use ~ instead of <Esc> for the Escape sequences, and PCL Barcode Flash will understand ~ as <Esc> as well. In spite of this, <Esc> can still be sent as <Esc> and ~ behave exactly the same in this mode.

However, you can also replace the alternative Escape character with another character by means of the following Escape sequence:

```
<Esc>**36J      or      <Esc>**$J
```

or

```
~**36J           or         ~**$J
```

This sequence causes the $ character (ASCII 36), for example, to be used from this point on instead of ~ as the alternative Escape character.

The following can also be used as alternative Escape characters:

"	(ASCII 34)
¥	(ASCII 92)
#	(ASCII 35)
｛	(ASCII 123)
$	(ASCII 36)
|	(ASCII 124)
/	(ASCII 47)
｝	(ASCII 125)
?	(ASCII 63)
~	(ASCII 126)

If so required, the Alternative Escape Character mode can be deactivated by means of the PRESCRIBE sequence

```
!R!VBST;E;EXIT;
```

or the Escape sequence

```
<Esc>**27J
```

or
```
~**27J
```

If you would like to use one or more of the sequences mentioned in this section, you can either send these from the computer to the printer or – better in most cases – store them on the Compact Flash card in the "Startup" partition. Cf. the chapter on this partition.

3.3 Barcodes and Matrix codes with different parameters

3.3.1 Datamatrix

Datamatrix code has a rectangular or square structure which is made up of individual square elements (black/white). The delimiting lines which are drawn on two adjacent sides and consist of alternate black and white elements on the opposite sides are characteristic. Datamatrix code can be used in a multitude of sizes, depending on the amount of data to be coded, the data security required and the space available in the document. As in the case of most other Matrix codes, a Datamatrix symbol can even be read without data loss if it is damaged or unreadable to a certain extent. This is achieved by means of coded and redundant distribution of the data over the entire area of the symbol.

Example of a square Datamatrix symbol:

There are 24 square and 6 rectangular Datamatrix symbols to choose from. In the case of PCL Barcode Flash, the symbols from 1 to 30 are numbered consecutively: cf. the table below.

You can choose which symbol is to be used or allow the software to use the smallest possible. If, for example, you want any square symbol, enter 1 to 24. If, on the other hand, you want to use any rectangular symbol, enter 25 to 30. However, you have to make your selection such that the symbol is large enough to accommodate your data. If you select too small a symbol, PCL Barcode Flash prints an error message instead of a symbol.

If you do not make a selection, the software automatically selects the smallest possible symbol in which the base data can be accommodated. Automatic selection always picks a square symbol. Therefore, if you want to use a rectangular symbol, you have to make a selection in any case.

You can also select a minimum number or a range from the 30 symbol numbers, e.g. 10 to 14. In this instance, no symbol smaller than no. 10 will be used, even if the base data could have been accommodated in a smaller symbol, and no symbol larger than no. 14 is used. In this example, if the base data cannot be accommodated in symbol no. 14, PCL Barcode Flash prints an error message instead of a symbol. This kind of range selection is sensible, for example, for applications in which the size of the data to be coded varies, but where the same symbol size is always required for visual or legibility reasons when printed, for example.

The size of the modules, the individual squares which go to make up the symbol, can also be stipulated. In PCL, the module size has to be entered in dots (pixels). In PRESCRIBE, on the other hand, the unit specified with the command UNIT is used; in this instance, UNIT D being strongly recommended for reasons of accuracy. The information in the case of UNIT does not refer to the entire symbol, but to each

individual module. The minimum size of a module is 1 dot, while the maximum permitted is 1000 dots at 600 dpi. The best read results are generally achieved at a symbol size of around 8 to 15 dots at 600 dpi. When making your selection, elements such as paper quality and colour, reading distance, scanning technique and light conditions are to be taken into account.

When programming, you should pay attention to the required quiet zone. This is the small white area that has to be left blank above, below and to the right and left of the symbol. According to the standard, the area equivalent to a module is required, but in practice a larger quiet zone is to be recommended.

☞ **Note**

> In specialist barcode language, "module" refers to the smallest square element of a symbol. "Symbol" refers to the combined modules in their entirety, mainly as squares or rectangles.

PCL Barcode Flash supports Datamatrix only in accordance with the widely applied ECC200 procedure. Extended Channel Interpretation and Structured Append Mode are not supported.

The following 30 Datamatrix symbols are available:

Square symbols

Symbol no	Rows	Columns	Data capacity		
			Num.	Alphanum.	Binary
1	10	10	6	3	1
2	12	12	10	6	3
3	14	14	16	10	6
4	16	16	24	16	10
5	18	18	36	25	16
6	20	20	44	31	20
7	22	22	60	43	28
8	24	24	72	52	34
9	26	26	88	64	42
10	32	32	124	91	60
11	36	36	172	127	84
12	40	40	228	169	112
13	44	44	288	214	142
14	48	48	348	259	172
15	52	52	408	304	202
16	64	64	560	418	278
17	72	72	736	550	366
18	80	80	912	682	454
19	88	88	1152	862	574
20	96	96	1392	1042	694
21	104	104	1632	1222	814
22	120	120	2100	1573	1048
23	132	132	2608	1954	1302
24	144	144	3116	2335	1556

Rectangular symbols

Symbol no.	Rows	Columns	Data capacity Num.	Alphanum.	Binary
25	8	18	10	6	3
26	8	32	20	13	8
27	12	26	32	22	14
28	12	36	44	31	20
29	16	36	64	46	30
30	16	48	98	72	47

Example of a rectangular Datamatrix symbol (symbol no. 29)
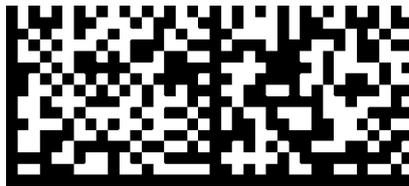
Datamatrix codes can be called with both PRESCRIBE and PCL commands.

Syntax in PCL

The T parameter always has to be entered. Another three optional parameters can also be entered. You can enter the parameters in the Escape sequence in any order. If PCL Barcode Flash encounters illegal values in one of the parameters, i.e. values outside the ranges mentioned below, an error message is printed instead of the Datamatrix symbol.

T parameter: identifier for Datamatrix
24820

h parameter: symbol number

Legal values: 1 to 30. Here, you can select either a specific symbol number or a range of symbol numbers separated by a comma. If you do not enter a value, PCL Barcode Flash selects the smallest symbol possible which can accommodate the data.

Examples:

8h: No symbol other than no. 8 is used. If the data cannot be accommodated in this, an error message is printed instead of the symbol.

5,10h: The software selects the smallest possible symbol which can accommodate the data, but it will not use any symbol smaller than no. 5 or larger than no. 10. If the data cannot be accommodated in symbol no. 10, an error message is printed instead of the symbol.

1,15h: The software selects the smallest possible symbol which can accommodate the data, but it will not use any symbol larger than no. 15. If the data cannot be accommodated in symbol no. 15, an error message is printed instead of the symbol.

25,30h: The software selects the smallest possible *rectangular* symbol which can accommodate the data. If the data cannot be accommodated in the largest rectangular symbol (= no. 30), an error message is printed instead of the symbol.

10,30h: The software selects the smallest possible *rectangular* symbol which can accommodate the data. If the data cannot be accommodated in the largest rectangular symbol (= no. 30), it switches to a square symbol (as you can see from the table above, square symbols that are larger than rectangular ones are available). Only when the data cannot be accommodated in the largest square symbol (no. 24) is an error message printed instead of the symbol.

b parameter: size of a module in dots, relative to 600 dpi

Legal values: 1 – 1000: it must be noted that a size of less than 6 cannot be recognised by most readers, and it may not be possible to print the symbol completely on the page in the case of very large module sizes. If you do not enter a value, 10 is used as the default.

v parameter: inversion

1 = Symbol is not printed inverted

2 = Symbol is printed inverted

If you do not enter the v parameter, 1 or not inverted is used as the default.

Examples:

```
<Esc>(s10h1v12b24820TData
```

```
<Esc>(s20b12,15h1v24820TData
```

Activation with no additional parameters, base data "1234567890":

```
<Esc>(s24820T1234567890<CR>
```

Syntax in PRESCRIBE

If you are not confident with programming in PRESCRIBE, we recommend that you read the "PRESCRIBE IIE PROGRAMMING MANUAL".

The `BARC` command is used. The syntax described in the PRESCRIBE manual for the `BARC` command is valid for the one-dimensional barcodes supported. As you will see below, the syntax for two-dimensional barcodes is slightly different.

The following parameters exist, all of which – except the "Barcode type" parameter – are optional. If these are entered, they must be separated by commas.

Barcode type, identity number for Datamatrix
100

Module size
The unit of module size is preceded by the `UNIT` command. We recommend that you use dots (pixels) => `UNIT D;`.

Lowest symbol number

Values from 1 to 30. To understand what this parameter means, compare the explanations for the h parameter in the section entitled "Syntax in PCL".

Highest symbol number

Values from 1 to 30. To understand what this parameter means, compare the explanations for the h parameter in the section entitled "Syntax in PCL".

Type of coding

0 must always be entered.

Error correction level

0 must always be entered.

Amount of base data in bytes

The base data can be passed in three different ways.

The amount of base data has to be stated only if the base data is passed in the first of the ways mentioned below:

1. By means of an attachment after the semicolon that closes the `BARC` command, in this case the number of bytes has to be stated.

2. In inverted commas.

3. By means of an attachment after the semicolon that closes the `BARC` command. The bytes are closed by means of the `WRED` command.

Example 1

`!R!UNIT D;BARC 100,10,13,14,0,0,11;<Base Data>EXIT;`

<Data> here is equivalent to exactly 11 bytes (including brackets).

Example 2

`!R!UNIT D;BARC 100,10,13,14,0,0,"<Base Data>";EXIT;`

In this case, the bytes do not have to be stated.

Example 3

`!R!UNIT D;BARC 100,10,10,14,0,0;<Base Data>WRED;EXIT;`

In this case too, the bytes do not have to be stated.

In example 3, the symbol number 10 is forced by entering the range from 10 to 10.

☞ **Note**

> `BARC 100` cannot be called from PRESCRIBE macros.

3.3.2 Datamatrix mode for printing from SAP

A second variant for printing Datamatrix from PRESCRIBE is also available. This method has been designed especially for printing from SAP. It uses the ID number 120 (instead of 100). The essential difference is that the useful data do not follow directly after the command, but rather must be located in one or more subsequent TEXT commands. Otherwise the BARC command is formed with the same parameters as in the variant with 100. In this connection, the BARC command is treated internally not so much as a barcode command, but rather to some extent as a kind of font command that simply switches from the font mode to the barcode mode. If multiple TEXT commands are used instead of a single TEXT command, their content is then combined in their direct sequence. In order to end the barcode mode

a FONT or MAP command must occur.

Example:

```
BARC 120,10,10,14; TEXT „0123456789"; TEXT „0123456789";
```

The advantage of this mode:

As lengthier useful data often have to be printed with Datamatrix, cases may occur in SAP where a single row cannot accommodate all of the useful data. If this should be the case, one of the lines would not be filled with useful data and the useful data would be disrupted by non-printable characters (e.g. printer control sequences) during programming. The advantage of the variant with 120 is therefore that in applications with SAP, the division of the useful data into multiple TEXT commands makes it possible to combine the data automatically nevertheless.

3.3.3 QR Code

This matrix code was developed by the company Denso and is particularly widespread in Japan, although more recently it has been used within the industry throughout the world.

Like the Datamatrix code, the QR Code can also be printed in a variety of sizes and resolutions.

QR Code symbols are available in 40 sizes. The smallest version (= version 1) forms a square of 21 x 21 modules, the largest one being 177 x 177 modules (= version 40), in which the distance between versions is always four modules. So version 2 consists of 25 x 25 modules, version 3 of 29 x 29 modules, etc. Module means the smallest quadratic element in a symbol. Symbol in the case of the QR Code means the combined entity of the module as a square.

An example for version 4, with 33 x 33 modules:

You can select the desired version using PCL Barcode Flash. But you must make your selection so that your data can be accommodated. If you select a version that is too small, the software prints an error message instead of the QR Code symbol.

If you fail to make a selection, the software automatically selects the smallest possible version in which the base data can be accommodated.

There are five types of character sets available for the QR Code in PCL Barcode Flash in which the data can be encoded in the symbol.

Numeric:
Digits 0 - 9

Alphanumeric:
Digits 0 – 9, letters A – Z, the space character and the characters $%*+-./:

Binary:

All 256 ASCII characters are accepted.

Kanji:

Practically all Kanji characters can be displayed. They must be available as two-byte characters in JIS X 0208 shift mode. If their encoding does not comply with the rules of the JIS Shift 0208 standard, instead of the symbol PCL Barcode Flash generates an error message: "Unknown Kanji".

Kanji and binary combined:

Kanji characters and non-Kanji characters are automatically combined, in which the latter are binary encoded.

For example, if only digits occur in your data, then you should select "Numeric" and not "Alphanumeric" or "Binary", as the data can be compressed more in "Numeric" mode and in some circumstances can thus be accommodated in a smaller symbol version.

If you select "Numeric" and characters occur other that the 10 digits and similarly if you select "Alphanumeric" and lowercase letters or special characters occur, for example, then PCL Barcode Flash rejects the data and an error message is printed instead of the QR Code symbol.

When selecting the encoding that best suits the application, you should know beforehand which characters may occur in your data and what the maximum symbol size is that is allowed. If symbol size is not particularly important for your application, then you are on the right track if you select "Binary". But be aware that the amount of time needed to read larger symbols can increase dramatically depending on the reading device.

You can also select a minimum version or a range of versions, e.g. 10 to 14. No versions lower than 10 will then be used, not even if the base data had also been accommodated in a lower version, and none greater than No. 14. If, as in this example, the base data is not accommodated in version 14, PCL Barcode Flash prints out an error message instead of a QR Code symbol. This kind of range selection is useful, for example, for applications in which the size of the data being encoded varies, but for the purposes of printing always having the same symbol size is desirable for visual or legibility reasons.

What is more, you can specify the error correction level. QR Code uses the Reed-Solomon procedure for error correction. You can choose between a proportion of the Reed-Solomon module for error correction of 7 %, 15 %, 25 % or 30 % (approximate values in each case). You should make this selection carefully based on the following criteria: When accommodating data, the version that is used and therefore the symbol size increases as the error correction level rises, consequently your space requirement and reading speed can decrease. There again, a higher error correction level may be recommended for poorer paper qualities or lighting conditions. The same applies if your application depends greatly on a high first read rate. When

selecting the error correction level, the following applies as when selecting the encoding: If the symbol size and the read speed are not particularly important for your application, then you are recommended to select 30 %.

Version 40, for example, with an error correction level of 7% is able to accommodate either 7089 digits, 4296 alphanumeric characters, 2953 binary bytes or 1817 Kanji characters. With an error correction level of 30 % there are only 3057 digits, 1852 alphanumeric characters, 1273 binary bytes or 784 Kanji characters, as the proportion of the Reed-Solomon module is higher.

You can specify the module size as an additional parameter. The minimum size is 1 dot and the maximum legal size is 1000 dots at 600 dpi. It is, of course, necessary to select a size at which the symbol will actually fit onto the paper format that you use. If you would like to print symbols with a very large module size, then, where applicable, this can only be achieved with smaller symbol versions. For instance, you may select a module size of up to 26 dots when using symbol version 40 at A4 format, whereas is may be 224 dots for symbol version 1, in each case relative to 600 dpi. In practice, these kinds of large module sizes are actually rarely needed. The most readable results are generally achieved at a module size of about 8 to 15 dots at 600 dpi. When making a selection, it is important to take into account paper quality and paper colour, reading distance, scanning technology and lighting conditions, for example.

When programming, you should also pay attention to the required quiet zone. This is the small white area, which has to remain free above, below, to the right and to the left of the symbol. The QR Code standard is required for the size of a module here, but in practice a larger quiet zone is recommended.

QR Code is available as model 1 and 2, where model 1 is only used in older applications. PCL Barcode Flash only supports the current model 2. International standards recommend only using model 2, which has been improved in a number of respects. If PCL Barcode Flash is given the instruction in a print job to print model 1, it automatically switches to model 2.

Extended Channel Interpretation (ECI mode) and Structured Append Mode are not supported by PCL Barcode Flash.

Syntax in PCL

The T parameter must always be specified as an identifier for the code. Another five parameters can also optionally be specified. The order in which you specify the parameters in the escape sequence doesn't matter. If PCL Barcode Flash encounters values that are not permitted for one of the parameters, i.e. values, which are outside the ranges set out below, then an error message is printed instead of the QR Code symbol.

T parameter: identifier for QR Code
24861

h parameter: symbol version
Legal values of 1 to 40. Here you can either select a specific version or a range of versions separated by commas. Do not specify the h parameter, as PCL Barcode Flash will select the smallest possible one for accommodating the data.

Examples:

8h: Version 8 will always be used. If the data cannot be accommodated, an error message will be printed instead of the symbol.

11,30h: The software selects the smallest possible version to accommodate the data, but does not use a version lower than 11 or a version higher than 30. If the data cannot be accommodated in version 30, an error message will be printed instead of the symbol.

1,15h: The software selects the smallest possible version to accommodate the data, but does not use a version higher than 15. If the data cannot be accommodated in version 15, an error message will be printed instead of the symbol.

s parameter: character set

Possible values:

0 = Automatic selection between Kanji and binary encoding

1 = Numeric: Digits 0 - 9

2 = Alphanumeric: Digits 0 – 9, uppercase letters A – Z, special characters $%*+-./:

3 = Binary: all 256 ASCII characters

4 = Kanji characters in two–byte encoding according to the JIS Shift 0208 standard

p parameter: Error correction level

Possible values:

1 =   7 %

2 = 15 %

3 = 25 %

4 = 30 %

If you do not specify the p parameter, then 15 % is used by default.

b parameter: Size of a module in dots relative to 600 dpi

Permissible values: 1 – 1000, where it should be noted that a size of less than 6 is too small to be recognised by most reading devices and that at very large module sizes the symbol cannot be printed in full on the page. If you do not specify the b parameter, then 10 is used by default.

v parameter: Inversion

1 = Symbol is not printed inverted

2 = Symbol is printed inverted

If you do not specify the v parameter, then 1 is used by default, i.e. not inverted.

Examples:

`<Esc>(s4p10h1v12b3s24861TDaten`

`<Esc>(s12,15h2v24861TDaten`

The following example including a print-out:

`<Esc>(s4p1v30b3s24861T` This is a QR Code symbol.

Syntax in Prescribe

In case you are not familiar with programming in PRESCRIBE, it is recommended that you read the "PROGRAMMING WITH PRESCRIBE IIE" manual.

The command `BARC` is used. The syntax described in the PRESCRIBE manual for the command `BARC` applies in respect of the one-dimensional barcodes that are supported. As you will be able to determine below, the syntax for two-dimensional barcodes is somewhat different from this.

There are the following parameters, which apart from the "Barcode type" parameter are all optional. If they are specified, they must be separated by commas.

Barcode type, identity number for QR Code:
102

Module size

The module size unit was previously specified using the `UNIT` command. We recommend that you use dots (pixels).

Lowest symbol version

Values from 1 to 40. For an explanation of the meaning of this parameter, cf. the explanatory notes on the h parameter in the section "SYNTAX IN PCL".

Highest symbol version

Values from 1 to 40. For an explanation of the meaning of this parameter, cf. the explanatory notes on the h parameter in the section "SYNTAX IN PCL".

Type of encoding

Values from 0 to 4. For an explanation of the meaning of this parameter, cf. the explanatory notes on the s parameter in the section "SYNTAX IN PCL".

Error correction level

Values from 1 to 4. For an explanation of the meaning of this parameter, cf. the explanatory notes on the b parameter in the section "SYNTAX IN PCL".

Quantity of base data in bytes

The base data can be passed in three different ways.

The quantity of base data only needs to be specified if the data is being passed in the first of the ways set out below:

1. By appending it after the semi-colon, which closes the `BARC` command, in this case the number of bytes must be indicated.

2. In inverted commas.

3. By appending it after the semi-colon, which closes the `BARC` command. The bytes are closed by the `WRED` command.

Example 1

`!R!UNIT D;BARC 102,10,13,14,2,4,11;<Base Data>EXIT;`

<Base Data> here corresponds to exactly 11 bytes.

Example 2

`!R!UNIT D;BARC 102,10,13,14,2,4,"<Base Data>";EXIT;`

In this instance it is not necessary to specify the number of bytes.

Example 3

```
!R!UNIT D;BARC 102,10,13,14,2,4;<Base Data>WRED;EXIT;
```

In this instance also, it is not necessary to specify the number of bytes.

Alphanumeric encoding and an error correction level of 30 % is used in these examples.

☞ **Note**

> `BARC 102` cannot be called from PRESCRIBE macros.

3.3.4 MaxiCode

MaxiCode is a matrix code developed by UPS. The content of the useful data is optimised for the information of the dispatch system and is primarily used in that area. Different information that is of relevance to shipment is specified (such as address components, weight etc.).

In contrast to the other 2D codes supported, the size of the individual modules as well as that of the complete MaxiCode symbol is fixed, i.e. they cannot be modified by the user. The dimensions of the complete symbol are specified as 25.4 mm x 25.4 mm. In contrast to the other 2D codes described, the printed modules comprise hexagons and there are circles at the centre (see figure below) that enable the reading device to locate it more rapidly. MaxiCode contains code words for the automatic correction of errors.

All 256 ASCII characters can be coded as useful data.

There is a standard, however, by which the useful data are divided into primary and secondary information.

Example of a MaxiCode symbol:
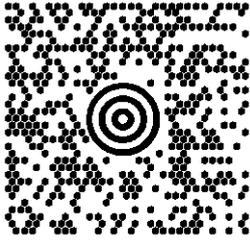
Syntax in PCL

The T-parameter must always be entered. No other parameters are specified with MaxiCode.

T-parameter: Identifier for MaxiCode

24800T

MaxiCode is initiated by the sequence:

<Esc>(s24800T

Additional parameters are not necessary in this sequence; apart from that, the same rules apply as those used for other barcodes.

The data following this sequence are interpreted and coded as useful data for the MaxiCode. The subsequent escape sequence terminates the data.

The data are divided into primary and secondary information and must appear in that order in the data. The primary information should contain information concerning the shipment. The secondary information should contain information for purposes of processing between consignor and consignee.

Other data received from your carrier firm, which often provides more precise guidelines for its customers, which are then also specially tailored for your application.

Primary information

The primary information is divided into six parts which must also appear in the

following order and be separated from each other by a comma:

1. No. of the symbol within a group, usually No. 1; the highest possible No. is 8.

2. Total number of symbols of this group, usually 1; the highest possible number of symbols in any given group is 8.

3. Mode; this is defined by the specification. As a general rule, it is mode 2 when using numerical postal codes and mode 3 when using alphanumeric postal codes. Mode 2 is used mainly in applications in the USA and mode 3 is generally used outside of the USA.

4. Postal code

5. Country code

6. Service class

Secondary information

The secondary information often begins with the character string "[)>"+ RS character (=field separator, ASCII character 30). The individual fields are separated by a GS character (=group separator, ASCII character 29) and the complete secondary information is closed with the RS character (=field separator, ASCII character 30) and the EOT character (=end of transmission, ASCII character 4), as shown in the following example.

Although the rules for the secondary information are not mandatory from the point of view of the MaxiCode, they are very widespread in dispatch systems throughout the world and should be observed for this reason. All data should be entered in capital letters.

The postal code may contain up to 9 digits in mode 2 (numerical) and up to 6 capital letters or digits in mode 3 (alphanumerical).

Specific secondary information:

1.
Ansi header: [)><RS>01<GS>96

2.
Tracking number<GS>
SCAC<GS>
UPS Shipper number<GS>
Julian day of pickup<GS>
Shipment ID<GS>
Package x of y<GS>

Package weight<GS>
Address validation<GS>
Ship to address<GS>
Ship to city<GS>
Ship to state<GS>

3.
End of format: <RS>
End of transmission <EOT>

As already mentioned, this must be followed directly by an escape sequence that requests a font.

Example of a MaxiCode request with primary and secondary information and subsequent font request sequence:

<Esc>(s24800T1,1,2,40670,840,001,[)><RS>01<GS>961Z00004951<GS>U SPN<GS>06X610<GS>159<GS>1234567<GS>1/1<GS>10<GS>Y<GS>OTT O-HAHN-STRASSE 12<GS>MEERBUSCH<GS>DE<RS><EOT><Esc>(10U><Esc>(s1p11vsb414 8T

Detailed explanation relative to this example:

1	= No. of symbol
1	= Number of symbols in this group
2	= MaxiCode mode
40670	= Postal code
840	= Country code
001	= Service class
1Z00004951	= Tracking number
USPN	= SCAC
06X610	= UPS Shipper number
159	= Julian day of pickup

1234567   = Shipment

1/1   = Package x of y

10   = Package weight

Y   = Address validation

OTTO-HAHN-STRASSE 12= Ship to address

MEERBUSCH= Ship to city

DE   = Ship to state

Syntax in PRESCRIBE

To control the MaxiCode from PRESCRIBE, the commands XBCP and XBAR are used.

The rules for the content of the useful data, i.e. the primary and secondary information, are described above in the syntax for controlling with PCL. They apply likewise in the case of a request issued from PRESCRIBE, with the difference that the primary information is formed with XBCP while the secondary information lies in the useful data. For the example below the same data are used as in the previous PCL example.

XBCP 0,8;
Reset and selection of MaxiCode. This command must appear before each MaxiCode request issued from PRESCRIBE.

XBCP 1,#;
stands for the MaxiCode mode; in the Barcode Flash solution, modes 2 through 6 are available.

XBCP 2,#;
For Structured Append mode: within a group of symbols, the sequential number of the symbol currently to be printed must be entered here. In the absence of this parameter, 1 is used as standard.

XBCP 3,#;
For Structured Append mode: the total number of symbols in the group of symbols currently to be printed must be entered here. In the absence of this parameter, the Barcode Flash solution assumes only one symbol as standard.

Note that in this case an error will, of course, occur if a value other than 1 is entered under XBCP 2,#.

XBCP 4,#;
stands for the postal code. Depending on the mode (which has been entered in XBCP 1,#), a numerical or alphanumerical postal code is permitted. If it is numerical, then a numerical parameter must be entered; if it is alphanumerical, on the other hand, a string must be entered in quotation marks. The postal code may contain up to 9 digits in mode 2 (numerical) and up to 6 capital letters or digits in mode 3 (alphanumerical). In modes 4 through 6, the entire entry XBCP 4,# is unnecessary.

XBCP 5,#;
Country number, which must comply with the specification. The Barcode Flash solution does not verify the validity of the country number; the requesting application is exclusively responsible for its accuracy. In modes 4 through 6, the entire entry XBCP 5,# is unnecessary.

XBCP 6,#;
Service class, which must comply with the specification. The Barcode Flash solution does not verify the validity of the service class; the requesting application is exclusively responsible for its accuracy. In modes 4 through 6, the entire entry XBCP 6,# is unnecessary.

These data are followed by the command XBAR, which is responsible for printing out the symbol using the parameters entered by the XBCP commands. This can be used here in either of two different ways. You should select one of these alternatives depending on the nature of your application.

1.
XBAR;<data>;ENDB;

Here the useful data follow directly after the command closed with the semicolon and are closed with the sequence ;ENDB;

2.
XBAR 19;<MAXICODE MAXICODE>

Here the number of bytes of the useful data is entered after the command and closed with a semicolon. This number of bytes is thereafter used as useful data. In this case, the 19-byte long string <MAXICODE MAXICODE> would have been printed out in the symbol.

Example of a complete request:

!R!RES;MZP 1,2;
XBCP 0,8;

```
XBCP 1,2;
XBCP 4,40470;
XBCP 5,840;
XBCP 6,001;
XBAR;[)><RS>01<GS>961Z00004951<GS>USPN<GS>06X610<GS>159
<GS>1234567<GS>1/1<GS>10<GS>Y<GS>OTTO-HAHN-STRASSE
12<GS>MEERBUSCH<GS>DE<RS><EOT>;ENDB;
PAGE;EXIT;
```

Notes

The order in which the different XBCP parameters are entered is of no importance. Unnecessary parameters need not be entered. For example, XBCP 2,#; and XBCP 3,#; can be left out if no Structured Append is to be used in the requesting application. Similarly, XBCP 4,#; can be left out in modes 4 through 6.

Please note that the commands XBAR and XBCP cannot be executed from PRESCRIBE macros in the current version of the Barcode Flash solution.

One additional point: the XBAR command is available in the printer language PRESCRIBE as standard in all printers and normally creates PDF417 therein. The Barcode Flash solution adds MaxiCode to the PRESCRIBE command XBAR.

3.3.5 PDF417

PDF417 is a matrix code invented by Symbol Technologies. PDF stands for 'Portable Data File'.

In contrast to the other 2D codes supported, the individual modules of the PDF417 are not specified as square, but rather in general as bars. The dimensions of the bars are not strictly defined by means of specifications, but rather can be defined by the user. For further details on this, please refer to the following description of the parameters.

A symbol comprises rows and columns, the number of which can also be determined by the user. In the case of PDF417, the term 'columns' refers to a total of 17 modules, each of which is divided into three rows and three empty spaces. Up to 90 rows and 30 columns can be selected. As a result, the range of options in selecting symbol dimensions is considerably broader than with the other matrix codes described. Identical line patterns are always located on the left and on the right in the symbol, and these identify it as a PDF417 as such. It is also possible to print without them, however. Please refer also to the detailed information under the parameters described below.

The level (degree) of error correction can also be selected by the user. Nine levels are available (0-8), whereby level 0 means 'no error correction desired'. Depending on the application and on the quality of the paper used, however, a higher level should be used.

All 256 ASCII characters can be coded as useful data.

Example of a PDF417 symbol with 18 rows and 5 columns:
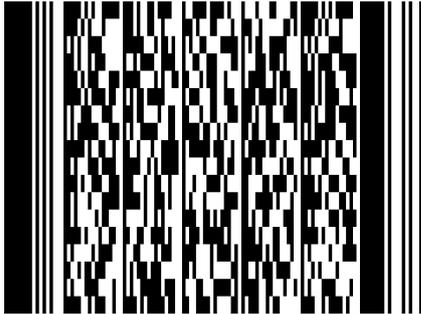
Syntax in PCL

The T-parameter must always be entered; the remaining three parameters, on the other hand, are optional. You can enter the parameters in the escape sequence in any order. If the Barcode Flash solution encounters impermissible values in one of the parameters, i.e. values that lie outside the ranges listed below, an error message is printed in lieu of the PDF417 symbol.

T-parameter: Identifier for PDF417

24850T

p-parameter: Error correction level

Permissible values: 0-8

These nine levels are predefined by the PDF417 specification. Note that this is a fixed number of error correction words, as follows:

Level	Error correction words
0	2
1	4
2	8
3	16
4	32
5	64
6	128
7	256
8	512

Example:

<Esc>(s3p24850T<data>

This entry means that you have selected level 3, in other words sixteen error correction words. The barcode module uses this regardless of the size of the useful data. However, you also have the option of making the number of error correction words dependent on the size of the useful data. This is particularly useful if your application is designed to output different kinds of useful data frequently. In this case you can enter the number of error correction words as a percentage (100 to 400) of the size of the useful data and add 1000 to it. Thus values of from 1000 to 1400 are permissible here. Example:

<Esc>(s1200p24850T<data>

This entry means that you have selected 200 per cent. Within the context of the nine possible level options, the barcode module will try to match the level of the error correction as closely as possible to 200 per cent of the useful data.

If you do not enter the p-parameter, a suitable value between 1 and 4 is used as standard.

Layout parameters

The following applies for the parameters below:

With these parameters, you can set four predefined values as sub-parameters. These must be separated by commas. You do not have to enter all four values if you do not wish to do so. You can also leave the b- and s-parameters out completely. In this case, the Barcode Flash solution selects the most appropriate standard values. If you want to leave out only certain sub-parameters, a comma is required as a placeholder. If you enter values outside the ranges listed below, the barcode module outputs an error message in lieu of the PDF417 symbol.

With certain sub-parameters relative to size relationships, the data might overlap. If they overlap as a result of input from the user, the Barcode Flash solution uses the last value entered and selects the most appropriate compromise.

b-parameter: Layout of the PDF417 symbol

1. Maximum number of rows. Possible values 3 to 90.

2. Maximum number of columns, up to 30.

3. Specification whether the entries in 1 and 2 should be fixed values or should represent the desired maximum size. If you enter 1 here, the entries are fixed values. Otherwise the barcode module tries to keep the symbol as small as possible. Entries 1 and 2 then apply as a maximum size, however. Please note that in this case an error message will be output in lieu of the symbol if the useful data encoded exceed this size.

4. The PDF417 specification also makes it possible to leave off the recognition bars on the right side of the symbol ('Truncated Mode'). In practice, however, this option is only rarely used. If you want to leave off the recognition bars, enter 1 here. In the absence of this parameter or if you enter another value, the symbol will be printed in the normal way with bars.

s-parameter: Layout of the individual modules of the symbol and proportions in the symbol

1. Size relationship of one module in dots based on 600 dpi. Standard value is 3; values from 1 to 10 are permissible.

2. X-value for the proportion of the symbol. In the absence of this sub-parameter as well as of sub-parameter 3 mentioned below, the Barcode Flash solution prints a

symbol with the proportion of 2 to 3 as standard. As the remaining rules for the layout of the symbol also apply, these values are always only approximations. The Barcode Flash solution always selects the most appropriate value here.

3. Y-value for the proportion of the symbol (cf. sub-parameter 2).

4. Width of a module in 1/100 inch; the standard value is 10.

The useful data follow directly after these parameters. Example of a PDF417 request with subsequent font request sequence:

<Esc>(s4,,,15s24850TABCDABCDABCDABCDABCDABCDABCDABC DABCDABCDABCDAB<Esc>(10U<Esc>(s1p12v0b0s4148T

3.3.6 Australia 4-state barcodes

The Australia Post 4-state barcode was introduced to improve sorting results and to enable the accurate selection of individual postal items. Australia Post uses a unique eight character Delivery Point Identifier (DPID), which is allocated to each delivery address in the postal address file (PAF).

The DPID is printed in barcode format on letters as part of an address. It is encoded to create the main part of the customer barcode.

PCL Barcode Flash converts the eight character DPID into a barcode, and information about the customer can also optionally be inserted.

The following four barcode structures are available:

1. **Australia 4-state 37-CUST** (Standard)

    PCL T parameter: 24785

    Activation: `<Esc>(s24785T<Base Data><Stop Character>`

    PRESCRIBE number: 45

    Activation: `!R!BARC 45,C,"Base Data"; EXIT;`

    Example: `<Esc>(s24785T23456789<CR>`

2. **Australia 4-state 52-FF-MET** (with 5-character customer information)

    PCL T parameter: 24786

    Activation: `<Esc>(s24786T<Base Data>,<ABCDE><Stop Character>`

    PRESCRIBE number: 46

    Activation: `!R!BARC 46,C,"Base Data,ABCDE"; EXIT;`

    Example: `<Esc>(s24786T23456789,Info<CR>`

3. **Australia 4-state 67-FF-MET** (with 10-character customer information)

PCL T parameter: 24787

Activation: `<Esc>(s24787T<Base Data><Stop Character>`

PRESCRIBE number: 47

Activation: `!R!BARC 47,C,"Base Data,ABCDEFGHIJ"; EXIT;`

Example: `<Esc>(s24787T23456789,Info<CR>`

4. **Australia 4-state FCC-45 REPLY** (reply paid)

PCL T parameter: 24788

Activation: `<Esc>(s24788T<Base Data><Stop Character>`

PRESCRIBE number: 48

Activation: `!R!BARC 48,C,"Base Data"; EXIT;`

Example: `<Esc>(s24788T23456789<CR>`

The data structures

The barcode data under 2 and 3 consist of the precise DPID incorporating eight numeric characters and the customer information, which have to be separated from one another by a comma.

The customer information may contain the following characters:

- The digits 0 to 9, the uppercase letters A to Z, the space character and the special character #.

- The barcode customer information under 2. may contain up to five characters, whereas the barcode 3 customer information may contain up to ten characters. Customer information is optional.

- The height and the width of the bars and the quiet zones are definitively prescribed in all Australia Post 4-state barcodes. Consequently, no optional parameters are necessary.

Please note that no plain text is printed with the Australia 4-state barcode.

☞ **Note**

BARC 45, 46, 47 and 48 cannot be called from PRESCRIBE macros.

3.4 Error messages

In practice, now and again errors occur when programming barcode activations, particularly during testing of an application. Because the least defective barcodes stand out without the appropriate reading device, error messages are an integrated part of the barcode solution. These are printed out in place of the barcode symbol to make the task of detecting errors easier for you.

The error messages are printed in plain text in a checked rectangle, e.g.:
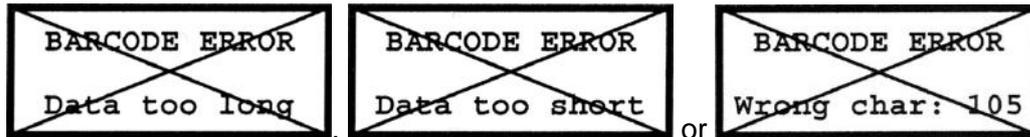
The following error messages exist:

Data too long
The length of the base data exceeds the defined maximum size for this barcode.

Data too short
The selected barcode requires a minimum number of characters, which has not been reached in the base data.

Wrong char: xxx *(where xxx = ASCII numeric value)*
The selected barcode does not allow the specified ASCII character, which actually occurred in the base data. Change the base data or use a different barcode, which contains the character in its character set. If, for instance, the error message "Wrong char: 121" occurs, then the letter "y" illegally appears in the base data, which is intended for a numeric barcode (121 as the ASCII value for "y").

Invalid param.
Values, which do not comply with the rules or are specified as being too high or too low, are set for the parameters p, h, v, b or s in the call sequence.

Number not odd
The selected barcode requires an odd number of characters. The base data, however, consisted of an even number of characters.

Number not even
The selected barcode requires an even number of characters. The base data, however, consisted of an odd number of characters.

Target too long

The base data was so long that it could not be accommodated by the selected barcode even following optimised compression.

Wrong data size

You requested QR Code or Datamatrix to use a specific maximum size or a set symbol format. However, they were not able to accommodate the extent of the base data. Reduce the amount of base data or select a different symbol size.

Unknown Kanji

A Kanji character represented by two bytes occurred in the base data for the QR Code, the encoding of which did not conform to the rules of the JIS Shift 0208 standard.

Comma expected

In the Australia Post 4-state barcode the DPID has to be separated from the customer information by a comma. The base data did not contain a comma.

Wrong DPID

Only eight-character digits are accepted as the DPID In the Australia Post 4–state barcode.

Wrong space

The base data is not allowed to start with a space character in the Australia Post 4–state barcode.

No N encoding

N encoding is not possible as characters other than the digits 0 to 9 occurred in the base data.

Encoding choice

You have selected an encoding in the Australia Post 4-state barcode in which the customer information cannot be represented.

Module too big

You have attempted to select a module size of over 1000 dots for a matrix code.

No Start Letter

You have specified an illegal character as the start character for Codabar. Codabar only allows the letters A,B,C and D as start characters.

No Stop Letter

You have specified an illegal character as the stop character for Codabar. Codabar only allows the letters A,B,C and D as stop characters.

Internal error

These errors should not really occur. Should any occur, then please contact your Support representative.

3.5 Examples and instructions on use

Do not set the toner density to the lowest contrast value. It is recommended that you set it to a density of 3 or higher. The setting can be made using the control panel or the PRESCRIBE sequence

```
!R! FRPO D4,3;EXIT;
```

The Toner Save mode should not be used when printing barcodes.

Where possible use white paper as this can enhance the contrast.

Carefully test your application. Have you selected the best settings for your printer and in your programming and optimised them for the read devices being used? A barcode application should always be viewed only as a whole in which all constituent parts should be attuned to one another.

One frequent cause of read errors is selecting quiet zones that are too small. They cannot be generated automatically by PCL Barcode Flash, but can only be done manually by careful programming and/or positioning.

If the read device used allows and different symbols don't have to be read alternately using the same device, you should permanently set the device to the symbology being read. Particularly when reading matrix codes, automatic recognition of barcodes and/or matrix codes should not be set as read speed will suffer as a result.

3.6 Barcode Types

3.6.1 UPC-A

Type / properties:

Linear code

Limitation: Plain text is only output in the default typeface in a set position below the code.

Activation sequence:

`<Esc>(s24600T<Base Data><Stop Character>`

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter	74
p parameter	3
b parameter	8,16,24,32

Additional properties:

Size of base data:	11
Character set:	0 to 9

Example:

`<Esc>(s24600T01234567890<CR>`

3.6.2 UPC-A + a 2 digit extension

Type / properties:

Linear code

Limitation: Plain text is only output in the default typeface in a set position below the code.

Activation sequence:

```
<Esc>(s24601T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter   74

p parameter   3

b parameter   8,16,24,32

Additional properties:

Size of base data: 13

Character set: 0 to 9

Example:

```
<Esc>(s24601T0123456789012<CR>
```

3.6.3 UPC-E

Type / properties:

Linear code

Limitation: Plain text is only output in the default typeface in a set position below the code.

Activation sequence:

```
<Esc>(s24610T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter   29

p parameter   3

b parameter   8,16,24,32

Additional properties:

Size of base data: 6

Character set: 0 to 9

Example:

```
<Esc>(s24610T012345<CR>
```

3.6.4 UPC-E + a 2 digit extension

Type / properties:

Linear code

Limitation: Plain text is only output in the default typeface in a set position below the code.

Activation sequence:

```
<Esc>(s24611T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter   29

p parameter   3

b parameter   8,16,24,32

Additional properties:

Size of base data: 8

Character set: 0 to 9

Example:

```
<Esc>(s24611T01234567<CR>
```

3.6.5 EAN/JAN-8

Type / properties:

Linear code

Limitation: Plain text is only output in the default typeface in a set position below the code.

Activation sequence:

```
<Esc>(s24620T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter	50
p parameter	3
b parameter	8,16,24,32

Additional properties:

Size of base data:	7
Character set:	0 to 9

Example:

```
<Esc>(s24620T0123456<CR>
```

3.6.6 EAN/JAN-8 + a 2 digit extension

Type / properties:

Linear code

Limitation: Plain text is only output in the default typeface in a set position below the code.

Activation sequence:

`<Esc>(s24621T<Base Data><Stop Character>`

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter   50

p parameter   3

b parameter   8,16,24,32

Additional properties:

Size of base data: 9

Character set: 0 to 9

Example:

`<Esc>(s24621T012345678<CR>`

3.6.7 EAN/JAN-8 + a 5 digit extension

Type / properties:

Linear code

Limitation: Plain text is only output in the default typeface in a set position below the code.

Activation sequence:

`<Esc>(s24622T<Base Data><Stop Character>`

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter   50

p parameter   3

b parameter   8,16,24,32

Additional properties:

Size of base data: 12

Character set: 0 to 9

Example:

`<Esc>(s24622T0123456,78901<CR>`

3.6.8 EAN/JAN-13

Type / properties:

Linear code

Limitation: Plain text is only output in the default typeface in a set position below the code.

Activation sequence:

`<Esc>(s24630T<Base Data><Stop Character>`

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter	62
p parameter	3
b parameter	8,16,24,32

Additional properties:

Size of base data:	12
Character set:	0 to 9

Example:

`<Esc>(s24630T12345678901<CR>`

3.6.9 EAN/JAN-13 + a 2 digit extension

Type / properties:

Linear code

Limitation: Plain text is only output in the default typeface in a set position below the code.

Activation sequence:

`<Esc>(s24631T<Base Data><Stop Character>`

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter	62
p parameter	3
b parameter	8,16,24,32

Additional properties:

Size of base data:	14
Character set:	0 to 9

Example:

`<Esc>(s24631T1234567890123<CR>`

3.6.10 EAN/JAN-13 + a 5 digit extension

Type / properties:

Linear code

Limitation: Plain text is only output in the default typeface in a set position below the code.

Activation sequence:

`<Esc>(s24632T<Base Data><Stop Character>`

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter	62
p parameter	3
b parameter	8,16,24,32

Additional properties:

Size of base data:	17
Character set:	0 to 9

Example:

`<Esc>(s24632T1234567890123456<CR>`

3.6.11 Interleaved 2 of 5

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24640T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter

p parameter

b parameter

Additional properties:

Size of base data: 2, 4, 6, ... ,98, 100 (even number)

Character set: 0 to 9

Example:

```
<Esc>(s24640T1234567890<CR>
```

3.6.12 Interleaved 2 of 5 with check digit

Type / properties:

Linear code

Activation sequence:

`<Esc>(s24641T<Base Data><Stop Character>`

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter

p parameter

b parameter

Additional properties:

Size of base data: 1, 3, 5, ... ,97, 99 (odd number)

Character set: 0 to 9

Example:

`<Esc>(s24641T1234567895<CR>`

3.6.13 Industrial 2 of 5 (25)

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24650T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter

p parameter

b parameter

Additional properties:

Size of base data: 2, 4, 6, ... ,98, 100 (even number)

Character set: 0 to 9

Example:

```
<Esc>(s24650T1234567890<CR>
```

3.6.14 Industrial 2 of 5 (25) with check digit

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24651T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter

p parameter

b parameter

Additional properties:

Size of base data: 1, 3, 5, ... ,97, 99 (odd number)

Character set: 0 to 9

Example:

```
<Esc>(s24651T1234567895<CR>
```

3.6.15 Matrix 2 of 5

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24660T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter

p parameter

b parameter

Additional properties:

Size of base data: 2, 4, 6, ... ,98, 100 (even number)

Character set: 0 to 9

Example:

```
<Esc>(s24660T0123456789<CR>
```

3.6.16 Matrix 2 of 5 with check digit

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24661T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:
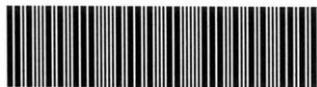
v parameter

p parameter

b parameter

Additional properties:

Size of base data: 1, 3, 5, ... ,97, 99 (odd number)

Character set: 0 to 9

Example:

```
<Esc>(s24661T0123456789<CR>
```

3.6.17 Code 39 (3 of 9) without space character

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24670T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter	29
p parameter	1
b parameter	6,18

Additional properties:

Size of base data:	1 to 100
Character set:	0 to 9, A to Z, a to z, $ % + - . / Space

Example:

```
<Esc>(s24670T1234567890ABC<CR>
```

3.6.18 Code 39 (3 of 9) with space character

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24672T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter	29
p parameter	1
b parameter	6,18

Additional properties:

Size of base data:	1 to 100
Character set:	0 to 9, A to Z, a to z, $ % + - . / Space

Example:

```
<Esc>(s24672T  1234567890ABC<CR>
```

3.6.19 Code 39 + Mod47 with check digit without space character

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24671T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter	29
p parameter	1
b parameter	6,18

Additional properties:

Size of base data:	1 to 100
Character set:	0 to 9, A to Z, a to z, $ % + - . / Space

Example:

```
<Esc>(s24671T1234567890ABCZ<CR>
```

3.6.20 Code 39 + Mod47 with check digit with space character

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24673T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter	29
p parameter	1
b parameter	6,18

Additional properties:

Size of base data:	1 to 100
Character set:	0 to 9, A to Z, a to z, $ % + - . / Space

Example:

```
<Esc>(s24673T  1234567890ABCP<CR>
```

3.6.21 Code 39 Extended

Type / properties:

Linear code

Activation sequence:

`<Esc>(s24680T<Base Data><Stop Character>`

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter	29
p parameter	1
b parameter	6,18

Additional properties:

Size of base data:	1 to 100
Character set:	All printable characters from ASCII 00 to 127

Example:

`<Esc>(s24680T 1234567890abc<CR>`

3.6.22 Code 39 Extended + Mod47 check digit

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24681T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter	29
p parameter	1
b parameter	6,18

Additional properties:

Size of base data:	1 to 100
Character set:	All printable characters from ASCII 00 to 127

Example:

```
<Esc>(s24681T 1234567890abc<CR>
```

3.6.23 Code 93

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24690T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter	29
p parameter	1
b parameter	6,12,18,24

Additional properties:

Size of base data:	1 to 100
Character set:	0 to 9, A to Z, a to z, $ % + - . / Space

Example:

```
<Esc>(s24690T1234567890ABC<CR>
```

3.6.24 Code 93 Extended

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24691T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter	29
p parameter	1
b parameter	6,12,18,24

Additional properties:

Size of base data:	1 to 100
Character set:	All printable characters from ASCII 00 to 127

Example:

```
<Esc>(s24691T 123456789!"#<CR>
```

3.6.25 Code 128 autoswitch

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24700T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter

p parameter

b parameter

Additional properties:

Size of base data: 1 to 100

Character set: All printable characters from ASCII 00 to 126

Example:

```
<Esc>(s24700T1234567890<CR>
```

3.6.26 Code 128 Subset A

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24701T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter

p parameter

b parameter

Additional properties:

Size of base data: 1 to 100

Character set: All printable characters from ASCII 00 to 95

Example:

```
<Esc>(s24701T1234567890ABC<CR>
```

3.6.27 Code 128 Subset B

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24702T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter

p parameter

b parameter

Additional properties:

Size of base data: 1 to 100

Character set: All printable characters from ASCII 32 to 126

Example:

```
<Esc>(s24702T1234567890abc<CR>
```

3.6.28 Code 128 Subset C

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24704T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter

p parameter

b parameter

Additional properties:

Size of base data: 1 to 100

Character set: 0 to 9

Example:

```
<Esc>(s24704T1234567890<CR>
```

3.6.29 UCC-128

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24710T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter	29
p parameter	1
b parameter	6,12,18,24

Additional properties:

Size of base data:	15 to 21
Character set:	0 to 9

Example:

```
<Esc>(s24710T123456789012345<CR>
```

3.6.30 EAN-128

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24720T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter	29
p parameter	1
b parameter	6,12,18,24

Additional properties:

Size of base data:	1 to 44, maximum size is limited to 35 symbols
Character set:	All printable characters from 00 to 126

Example:

```
<Esc>(s24720T1234567890<CR>
```

3.6.31 Codabar

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24750T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter

p parameter

b parameter

Additional properties:

Size of base data:	1 to 32
Character set:	0 to 9, $ + - . / :
Start and stop characters:	A B C D

Example:

```
<Esc>(s24750TA1234567890B<CR>
```

3.6.32 Codabar + Mod16 check digit

Type / properties:

Linear code

Activation sequence:

```
<Esc>(s24751T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter

p parameter

b parameter

Additional properties:

Size of base data:	1 to 32
Character set:	0 to 9, $ + - . / :,
Start and stop characters:	A B C D

Example:

```
<Esc>(s24751TA1234567890B<CR>
```

3.6.33 MSI Plessey

Type / properties:

Linear code

Limitation: Plain text is only output in the default typeface in a set position below the code, which means that the h parameter does not have any effect and in the case of the p parameter only the values 1 and 4 are allowed.

Activation sequence:

```
<Esc>(s24760T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter	29
p parameter	1
b parameter	6,12

Additional properties:

Size of base data:	1 to 14
Character set:	0 to 9

Example:

```
<Esc>(s24760T0123456789<CR>
```

3.6.34 MSI Plessey + Mod10 check digit

Type / properties:

Linear code

Limitation: Plain text is only output in the default typeface in a set position below the code, which means that the h parameter does not have any effect and in the case of the p parameter only the values 1 and 4 are allowed.

Activation sequence:

```
<Esc>(s24761T<Base Data><Stop Character>
```

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter   29

p parameter   1

b parameter   6,12

Additional properties:

Size of base data: 1 to 14

Character set: 0 to 9

Example:

```
<Esc>(s24761T12345678<CR>
```

3.6.35 MSI Plessey + Mod10 + Mod10 check digit

Type / properties:

Linear code

Limitation: Plain text is only output in the default typeface in a set position below the code, which means that the h parameter does not have any effect and in the case of the p parameter only the values 1 and 4 are allowed.

Activation sequence:

`<Esc>(s24762T<Base Data><Stop Character>`

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter	29
p parameter	1
b parameter	6,12

Additional properties:

Size of base data:	1 to 14
Character set:	0 to 9

Example:

`<Esc>(s24762T12345678<CR>`

3.6.36 MSI Plessey + Mod10 + Mod10 check digit

Type / properties:

Linear code

Limitation: Plain text is only output in the default typeface in a set position below the code, which means that the h parameter does not have any effect and in the case of the p parameter only the values 1 and 4 are allowed.

Activation sequence:

`<Esc>(s24763T<Base Data><Stop Character>`

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter   29

p parameter   1

b parameter   6,12

Additional properties:

Size of base data: 1 to 14

Character set: 0 to 9

Example:

`<Esc>(s24763T12345678<CR>`

3.6.37 ZIP + 4 Postnet 5

Type / properties:

Linear code, used in postal traffic within the U.S.A.

Limitation: Plain text is not recommended. This code can only output plain text in the default typeface in a set position below the code.

Activation sequence:

`<Esc>(s24770T<Base Data><Stop Character>`

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter   9 (cannot be modified)

p parameter   1 (cannot be modified)

b parameter

Additional properties:

Size of base data: 5

Character set: 0 to 9

Example:

`<Esc>(s24770T12345<CR>`

3.6.38 ZIP + 4 Postnet 9

Type / properties:

Linear code, used in postal traffic within the U.S.A.

Limitation: Plain text is not recommended. This code can only output plain text in the default typeface in a set position below the code.

Activation sequence:

`<Esc>(s24771T<Base Data><Stop Character>`

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter	9 (cannot be modified)
p parameter	1 (cannot be modified)
b parameter	

Additional properties:

Size of base data:	9
Character set:	0 to 9

Example:

`<Esc>(s24771T123456789<CR>`

IııılIıılıılıılIıılıılılılIıılIıılıılIıılılılııılılıl

3.6.39 ZIP + 4 Postnet 11

Type / properties:

Linear code, used in postal traffic within the U.S.A.

Limitation: Plain text is not recommended. This code can only output plain text in the default typeface in a set position below the code.

Activation sequence:

`<Esc>(s24772T<Base Data><Stop Character>`

For an explanation on selecting the stop character and the optional parameters, see chapters 3.2.1. and 3.2.6.

Default - parameters:

v parameter	9 (cannot be modified)
p parameter	1 (cannot be modified)
b parameter	^

Additional properties:

Size of base data:	11
Character set:	0 to 9

Example:
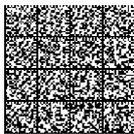
`<Esc>(s24772T12345678901<CR>`

3.6.40 Datamatrix

Printing examples:

The following text is printed out with different parameters as the DataMatrix symbol:

'Data Matrix is a two-dimensional matrix symbology which is made up of square modules arranged within a perimeter finder pattern that is used for finding the symbol and determining its orientation. Two of the outside rows are solid, forming the letter "L", and the opposite two perimeter rows use an alternating black-white pattern which helps synchronize the decoding process.'

```
<Esc>(s5b24820T<above text><CR>
```



```
<Esc>(s11b24820T<above text><CR>
```



```
<Esc>(s23b24820T<above text><CR>
```

3.6.41 QR code

Printing examples:

The following text is printed out with different parameters as the QR Code symbol:

'QR CODE IS A MATRIX SYMBOLOGY CONSISTING OF AN ARRAY OF NOMINALLY SQUARE MODULES ARRANGED IN AN OVERALL SQUARE PATTERN INCLUDING A UNIQUE FINDER PATTERN LOCATED AT THREE CORNERS OF THE SYMBOL AND INTENDED TO ASSIST IN EASY LOCATION OF ITS POSITION SIZE AND INCLINATION. A WIDE RANGE OF SIZES OF SYMBOL IS PROVIDED FOR TOGETHER WITH FOUR LEVELS OF ERROR CORRECTION. MODULE DIMENSIONS ARE USER-SPECIFIED TO ENABLE SYMBOL PRODUCTION BY A WIDE VARIETY OF TECHNIQUES.'

```
<Esc>( s5b1p2s24861T<above text><CR>
```



```
<Esc>( s11b1p2s24861T<above text><CR>
```



```
<Esc>( 23b1p2s24861T<above text><CR>
```

3.7 Expanding typefaces

PCL Barcode Flash expands the integrated default printer typefaces to include Letter Gothic, Line Draw, OCR-A and OCR-B. Otherwise barcodes 3 of 9, EAN/UPC and USPS ZIP are added in the form of typefaces. These barcodes are composed of individual fixed elements, comparable with the way in which a word is made up of individual letters.

There are some applications on the market, mainly older in origin, which control the barcode using this method. The printers used in this environment normally need a so-called 'Barcode Font Cartridge' - a special memory module - as an optional accessory.

Kyocera printers and multifunction devices can be used with PCL Barcode Flash without having to intervene into the applications.

What is more, 'Barcode Font Cartridges' can be controlled using some Microsoft Windows standard printer drivers, and Kyocera devices can now be used for this too.

3.7.1 Checking Typefaces

It is possible to check that the typefaces work properly as follows:

- Print out a list of the optional fonts (printer display, the 'Font Option' > 'List of Fonts' menu item). This might also necessarily result in layout changes on the test page caused by the dimensions of the barcode characters, but this is a harmless side effect and doesn't impair functionality

- Print out the "PCL Barcode Flash Status Page" ('Memory Card' > 'Read data' > 'Barcode Status' menu item)

- Activate one of the fonts and print out data. Example:

```
!R! FONT 830; TEXT "*1234*"; PAGE; EXIT;
```

3.7.2 Activating fonts using a PRESCRIBE command

Either using the individual FONT command or the pair of commands, CSET and FSET (CSET and FSET should always be used together as otherwise the font activation cannot be correctly recognised by the printer).

The zero character is displayed as 0 and it should not be confused with the letter O!

Barcode 3 of 9;   8.1 cpi;   12 point:
FONT 830; or CSET 0Y; FSET 0p8.1h12v0s0b0T;

Barcode 3 of 9;   4.6 cpi;   12 point:
FONT 831; or CSET 0Y; FSET 0p4.6h12v0s0b0T;

EAN/UPC 10 Mil; proportional;   12 point:
FONT 832; or CSET 8Y; FSET 1p12v0s0b0T;

EAN/UPC 13 Mil; proportional;   12 point:
FONT 833; or CSET 8Y; FSET 1p12v0s3b0T;

Letter Gothic;   17 cpi;   9.5 point:
FONT 834; or CSET 8U; FSET 0p16.66h9.5v0s0b0T;

Letter Gothic;   12 cpi;   12 point:
FONT 835; or CSET 8U; FSET 0p12h12v0s0b0T;

Letter Gothic;   10 cpi;   14 point:
FONT 836; or CSET 8U; FSET 0p10h14v0s0b0T;

Line Draw;   10 cpi;   12 point:
FONT 837; or CSET 0B; FSET 0p10h12v0s0b0T;

OCR-A;   10 cpi;   12 point:
FONT 838; or CSET 0O; FSET 0p10h12v0s0b104T;

OCR-B;   10 cpi;   12 point:
FONT 839; or CSET 1O; FSET 0p10h12v0s0b110T;

USPS ZIP;   proportional;   12 point:
FONT 840; or CSET 15Y; FSET 1p12v0s0b0T;

3.7.3 Activating fonts using a PCL command

In the PCL emulation of Kyocera devices, the fonts can also be selected using the corresponding PCL command. The Escape character (ASCII character 27 in decimal or 1B in hexadecimal is displayed here as [esc]. Please do not use the individual characters in brackets, just the actual Escape character.

Barcode 3 of 9;	8.1 cpi;	12 point:	[esc](0Y[esc](s0p8.1h12v0s0b0T
Barcode 3 of 9;	4.6 cpi;	12 point:	[esc](0Y[esc](s0p4.6h12v0s0b0T
EAN/UPC 10 Mil;	prop.;	12 point:	[esc](8Y[esc](s1p12v0s0b0T
EAN/UPC 13 Mil;	prop.;	12 point:	[esc](8Y[esc](s1p12v0s3b0T
Letter Gothic;	17 cpi;	9.5 point:	[esc](8U[esc](s0p16.66h9.5v0s0b0T
Letter Gothic;	12 cpi;	12 point:	[esc](8U[esc](s0p12h12v0s0b0T
Letter Gothic;	10 cpi;	14 point:	[esc](8U[esc](s0p10h14v0s0b0T
Line Draw;	10 cpi;	12 point:	[esc](0B[esc](s0p10h12v0s0b0T
OCR-A;	10 cpi;	12 point:	[esc](0O[esc](s0p10h12v0s0b104T
OCR-B;	10 cpi;	12 point:	[esc](1O[esc](s0p10h12v0s0b110T
USPS ZIP;	prop.;	12 point:	[esc](15Y[esc](s1p12v0s0b0T

☞ Note

Please note that the barcodes in this section do not have any 'inherent intelligence' like the barcodes controlled using BARC. You have to integrate into the barcode text all of the checksums, the start and stop symbols at the right places with their proper values. Otherwise your barcode reader will not be able to read the barcode properly.

3.7.4 Examples

Print out a barcode 39 with PRESCRIBE:
```
!R! FONT 831; TEXT "*0987654321*"; PAGE; EXIT;
```
("*" required as the start and stop character)

Print out of a UPC barcode with ORC-A script embedded below in PCL:

```
[cr][lf]
[esc](8Y[esc](s1p12v0s0b0T(012345-GHIJAF([cr][lf]
(012345-GHIJAF([cr][lf]
(012345-GHIJAF([cr][lf]
(        -        ([cr][lf]
[esc](0O[esc](s0p10h12v0s0b104T[esc]*p-50y+25X12345 67890
```

The line feed character has been displayed as [lf] and the carriage return character as [cr].

4 Appendix

4.1 References

These references are intended as a starting point for familiarising yourself with the variety of subject matter on barcodes. The list is in no way complete and does not claim to offer a representative choice in terms of quality.

The ISO/IEC documentation can be purchased from www.iso.ch

It is generally recommended that you familiarise yourself with the following official manuals, which are normally included on CD as a PDF document with every Kyocera device:

- PRESCRIBE IIe Programming Manual (English)

- PDF417 Two Dimensional Bar Code Implementation Manual (English)