dLSoft

Barcodes

By dLSoft

This manual was produced using *ComponentOne Doc-To-Help.*™

Contents

Barcodes 1

Barcodes

Introduction

dLSoft products support a wide range of barcode types and we endeavour to keep up to date with barcode specifications. However, it is important to understand that the standards specified for barcodes have arisen from a wide range of sources, and some barcode specifications have been modified over a period of time. Furthermore some barcode types have been largely superseded by more modern code types, usually because modern types have a higher reliability.

In these notes we aim to provide:

1) Details of the barcode types supported by dLSoft barcode products; note that not all products support all barcode types included here.

2) Some general information about the codes you need to provide to produce satisfactory barcode images

3) The code # numbers required to access the barcode types if you are programming the dLSoft products that support multiple barcode types, such as the Universal products (dBarcode DLL, dBarcode.NET, dBarcode-2k, Active Barcode Component Universal). Programmers who are using multiple code types are advised to print out the code type table.

The dLSoft barcode library forms the basis of the dBarcode range of products and a number of label and form printing applications. For this reason a number of references to dBarcode appear in the text.

1D Barcodes

Several fundamental characteristics of barcodes need to be understood by users of dLSoft barcode products:

1. The thickness of bars in barcodes is important. The size of the smallest element of a barcode is know as its X unit size or X dimension and in standards this is usually specified in units of Mils (0.001 inches).

dLSoft barcode products may refuse to create a barcode image if the bar thickness within the metafile becomes too small. However, even when dLSoft barcode products creates an image you may resize it within another application so that when it is printed by the other application its lines may too small for the printer's resolution. Consequently it is essential that you check that a printed barcode is readable using an appropriate scanner or reader.

Barcodes printed by laser printer will, in general, be printed correctly, but codes printed by matrix printers must be reproduced at a large enough scale that the barcodes unit size is at least as large as the printer's pins.

Bar reduction: All dLSoft barcode products allow the thickness of bars to be reduced (for example to allow for ink spread during wet ink printing processes), but this adjustment should only be made when the knowledge of the extent of reduction required is available. Random guesses usually produce unreadable images!

2. Many barcode types may use codes only of a specific length. (e.g. EAN13 requires 13 digits in the code). Some barcode type use specific digits of the code as a checksum - so not every combination of digits can form a legal barcode. dLSoft barcode products can optionally calculate checksum digits, requiring only the other digits to be entered by the user. Furthermore most coding schemes are limited to 32 characters or less.

3. The barcode types supported in this release are shown in the barcodes table below. If you plan to use a specific barcode type you should examine the notes on that type before printing any barcode images.

4. Users should be aware that it is possible to generate barcodes of a specific type and find that normal retail scanners are unable to decode the images. This does not necessarily mean that there is anything wrong with the barcode image. Most scanners aimed at the retail market are not programmed to interpret barcode codes reserved for other (eg. military) use.

5. The Extra options. All of our products, which use the DLSBARxx barcode library, provide access to two options not detailed in the product manuals. These are the options EXTRA1 and EXTRA2, which may appears as checkboxes in dialogs, or as additional bit flags in the DLL or OCX. These options are used only for a limited number of barcodes, which have “unusual” features. The effect of these options is described under the barcode types, which use them. For all other barcode types these options may be ignored or set to 0.

Barcode types supported

In the table below the types of barcodes supported by this release of the library are summarised, together with the type and number of characters, which are specified for that barcode type. In this connection "any" means capital letter and number characters; in some cases additional characters are defined, but lower case letters are not permitted. The code# represents the code type number used in calls to the dBarcode DLL (DLSBAR32.DLL).

Telepen provides codes for the first 127 ASCII characters, but with restrictions on the ordering of letters and numbers. Only Extended Code 39 and the EAN 128 and Code 128 codes provide symbols for the full ASCII character set.

There are many named barcode types, which are actually derivatives of major types. To avoid the table (and user-programming) becoming excessively complex, both the table and calls to the library report only the generic name.

For example: The ISBN, ISSN and JAN coding scheme are all variants of the EAN scheme.

There are several coding schemes (such as DEFCON) which are actually Code 39, and some countries use Code 128 under other names for mail tracking (as in the UK).

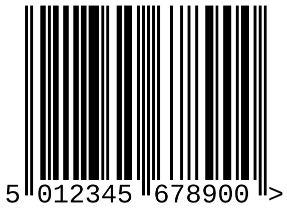
Barcode Types Table

code#	.NET CodeType	Code type	no. of characters	check digit
0	12	EAN13	13 numbers	1
1	15	EAN-8	8 numbers	
2	13	EAN13+2	15 numbers	1
3	14	EAN13+5	18 numbers	1
4	18	UPC-A	12 numbers	1
5	21	UPC-E	7 numbers	1
6	40	ITF-14	14 numbers	1 (EAN optional)
7	39	ITF-6	6 numbers	
8	0	Code 39	any	1 optional
9	8	Code 128	any*	automatic
10	9	EAN/UCC-128	any*	automatic
11	3	2 of 5	any numbers	
12	4	Interleaved-2 of 5	number pairs	1 optional
13	0	3 of 9	any	
14	38	Code B	any numbers	
15	37	Code 11	any	1 or 2
16	2	Codabar/NW-7	any	
17	22	MSI	any numbers	1 or 2
18	1	Ext. Code 39	any (full ASCII)	1 optional
19	19	UPCA+2	14 numbers	1
20	20	UPCA+5	17 numbers	1
21	16	EAN8+2	10 numbers	1
22	17	EAN8+5	13 numbers	1
23		UPCE+2	9 numbers	1
24		UPCE+5	12 numbers	1
25	33	Telepen standard	any	1 optional
26	34	Telepen ASCII	any	1 optional
27	35	Telepen numeric	any	1 optional
28	25	PostNet type A	5 numbers	1
29	25	PostNet type C	9 numbers	1
30	25	PostNet type C'	11 numbers	1
31		FIM A	fixed code	0
32		FIM B	fixed code	0
33		FIM C	fixed code	0

34	28	RM4SCC	any	1
35	29	4-State	any	1 optional
36	6	Code 93	any	2 optional
37	7	Ex Code 93	any (full ASCII)	2 optional
38	30	ISBN	10/13 digit ISBN	1 automatic
39	5	Matrix 2/5	numeric	1 optional
40	23	Plessey	numeric/some alpha	2
41		Australia Post	numeric	automatic
42		Swiss Post	fixed code	0
43	24	Deutsche Post	numeric	
44		SISAC	SICI codes	1
45	11	EAN/UCC-14	13	1
46	26	Planet 12	12	1 automatic
47	27	Planet 14	14	1 automatic
48	31	ISSN	9/10 digit ISSN	1 automatic
49	32	ISMN	8/9 digit ISMN	1 automatic
50	10	SSCC	17	1
51		Korean Postal Authority	6	1
52		Italian Postal 3/9	2+8+c+2	1
53		Italian Postal 2/5	12	1 automatic
54	30	ISBN+2 digit	any / 2	1 automatic
55	30	ISBN+5 digit	any / 5	1 automatic
56	31	ISSN+2 digit	any / 2	1 automatic
57	31	ISSN+5 digit	any / 5	1 automatic
58	36	Japan Post	any	automatic
59	41	IATA 2 of 5	Any numeric	

EAN

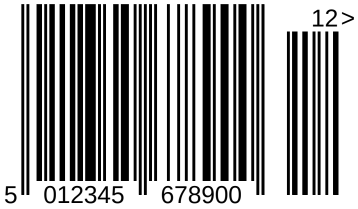
EAN-13 is the main scheme used throughout Europe for retail article numbering. It is a numeric only coding scheme. The > symbol in the right margin is a light margin indicator. In the left margin the first code digit is used as the margin indicator. No other marking should appear in the light margins.



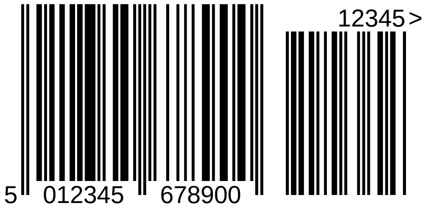
EAN-13

EAN codes require 13 digits (12 if the check digit is calculated automatically. Numbers used for EAN article numbering are assigned by the country's Article Number Association (the ANA in the UK).

EAN codes may contain 2 or 5 digit supplementaries:



EAN13+2



EAN13+5

The ISBN coding scheme is EAN13, with the first three digits being 978 or 979, and 9 digits the ISBN number of the book (without check digit). The final digit is the EAN calculated check digit. (See also the ISBN entry below).

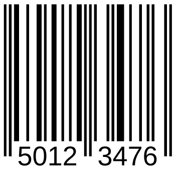
The ISSN coding scheme is EAN13, with the first three digits being 977, 7 digits showing the ISSN number of the periodical (without check digit), and 2 spare digits (used in the UK to indicate price code changes, but which are otherwise 00). The final digit is the EAN calculated check digit.

The ISMN coding scheme is EAN13, with the first three digits being 979, followed by 0 and the first 8 digits of the ISMN number. The final digit it the EAN calculated check digit.

The JAN coding scheme is EAN13 with the first two digits being 49.

Note that there is not a one to one correspondence between bars and the code numbers.

EAN-8 is a smaller and shortened version of the EAN code.



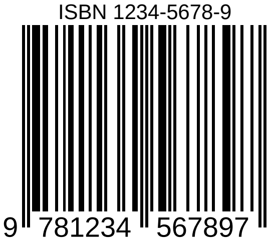
EAN-8

EAN-8 requires 8 digits (7 if the check digit is calculated automatically), and support 2 and 5 digit supplementaries.

ISBN

The ISBN-10 coding scheme is EAN13, with the first three digits being 978 or 979, and 9 digits the ISBN number of the book (without check digit). The final digit is the EAN calculated check digit.

Users can produce the ISBN barcode by selecting EAN as the barcode type and entering the EAN number. Alternatively the ISBN barcode complete with the ISBN text above the barcode may be obtained by selecting ISBN as the barcode type and entering the ISBN 9 or 10 digit ISBN value (which may include dashes, eg. 1-2345-6789-1). The barcode image which results is as shown below.

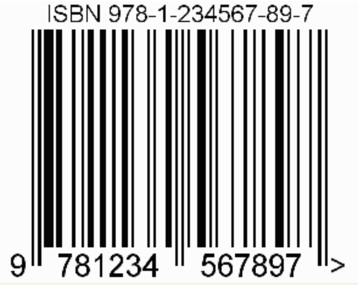


ISBN

Note that the final digit of a 10 digit ISBN number is an ISBN check digit and this is NOT included in the barcode image. The barcode image will contain the EAN check digit.

The spacing of the text above the barcode may be modified by entering a character spacing value between 50 and 100%.

NOTE: In January 2007 (or soon thereafter), the US ISBN Agency will begin assigning 13-digit identifiers to books and book-related products. These identifiers, "ISBN-13s", will incorporate the existing Bookland EAN into a new book identifier where the prefix can be either '978' (current value) or '979'.



ISBN-13

dLSoft Barcode components will automatically recognize ISBN-13 barcode data starting with 978 or 979 and generate the appropriate ISBN-13 barcode.

dLSoft Barcode components will generate ISBN-13 barcodes using the ‘978’ prefix from data supplied as the ISBN 10 data, if the EXTRA1 property is set to TRUE; otherwise it will generate an ISBN-10 barcode as before.

ISBN with 2 and 5 digit supplementaries are supported by some dLSoft products. The supplementary characters must be separated from the ISBN numbers with a / character.

Bookland barcodes are unique numbers that are printed on the covers of books. They contain the book's ISBN number and pricing information encoded using EAN 13 bar codes with a 5 digit supplementary code.

For a book with **ISBN 1-234-5678-9** retailing at **$19.95** in the US, the data to encode is generated by taking **978**, followed by the ISBN number 12345678 *(the last digit of the ISBN number is a check digit and is not included*), followed by a currency digit (5 for US$) and a four digit price (51995), ie.

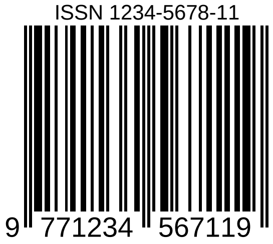
9781234567851995

These numbers may be entered in dLSoft products using the \ separator between the ISBN number and the supplementary, eg

1-2345-6789-0/51995

ISSN

The ISSN coding scheme is EAN13, with the first three digits being 977, 7 digits showing the ISSN number of the periodical (without check digit), and 2 spare digits (used in the UK to indicate price code changes). The final digit is the EAN calculated check digit.

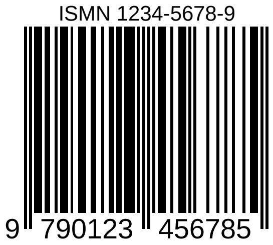


ISSN

ISSN with 2 and 5 digit supplementaries are supported by some dLSoft products. The supplementary characters must be separated from the ISSN numbers with a / character.

ISMN

The ISMN coding scheme is EAN13, with the first four digits being 9790, followed by the first 8 digits of the assigned Music number. The Music number check digit is not required. The final digit in the barcode is the EAN check digit.



JAN

The JAN coding scheme is then same as EAN13 with the first two digits being 49.

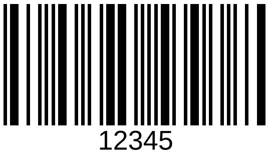
Codabar

Codabar is a discrete system developed by Monarch Marking Inc in 1972. It is know as NW-7 in Japan, as it has Narrow and Wide bars and 7 elements per character. It is also known as USD-4 and 2 of 7 code.

The Codabar coding scheme is a self-checking system which has 16 characters in its character set; the digits 0 -9, and the characters $: / . + -. It has a choice of four start & stop characters, although some versions allow a choice of eight!. By default dLSoft barcode products uses A and C for start and stop respectively. However, by prefixing the barcode with a caret (^) and two symbols, any of the allowed Codabar characters may be used for start and stop; ie.

^AT

causes A to be used as the start character and T to be used for the stop character.



Codabar

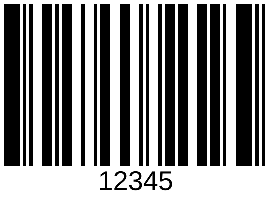
The allowed Codabar start and stop characters are: A B C D E N T *

The start and stop characters are not displayed in text form.

Codabar does not define a check digit. However, Mod 10 check digits are often used in Codabar symbols, so the dLSoft products will generate a Mod 10 check digit if required.

Matrix 2/5

Matrix 2/5 is an older numeric-only code, with an optional modulo 10 check digit. Not recommended for new applications.



Matrix 2/5

NW-7

See Codabar

Telepen

Telepen provides three coding schemes, each having its own pair of start and stop characters:

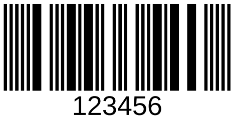
Full ASCII – encodes the lower 128 ASCII characters

Compressed Numeric – where the encoding starts in compressed numeric mode (encoding two digits per group of bars) and may be followed by ASCII encoding. This scheme is for encoding an even number of digits. If an odd number of digits is to be encoded then either a switch from Compressed numeric to ASCII is required before the final digit or the string of digits must be prefixed with a 0.

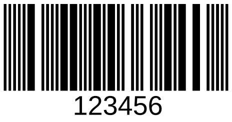
ASCII – where the encoding starts in full ASCII mode and is followed by digits in compressed numeric encoding.

The switch from ASCII to Compressed numeric (or vice versa) is accomplished by the insertion of an ASCII DLE character, but this is permitted only once in any symbol.

The dLSoft barcode library provides standard Telepen (in which the above are followed), Telepen N (which uses the above scheme but forces the full ASCII start and stop bars), and Telepen A (which encodes all data as ASCII without compressed numeric). The Telepen ASCII mode provides the full ASCII character set. Codes below 32 (space) may be entered as <ALT>0XYZ, where XYZ is the 3 digit ASCII code +128. The ASCII ESC character required on some Telepen Numeric systems as the first character may be obtained by checking the EXTRA1 checkbox in applications, or setting the flags parameter bit DL_FLAG_EXTRA1 (bit 4 of the flags variable) or BarCode.Extra1 in the OCX. Normally a switch from Compressed numeric to ASCII is used to handled odd length digit strings, but by checking the EXTRA2 checkbox in applications, or setting the flags parameter bit DL_FLAG_EXTRA2 (bit 4 of the flags variable) or BarCode.Extra1 in the OCX, the library will prefix the digits with a 0 to make the total number of digits even.



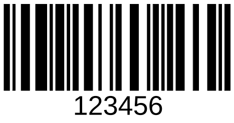
Telepen



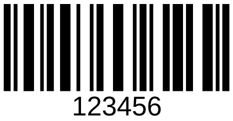
Telepen N

Code 128 & EAN/UCC128

Code 128 and EAN/UCC-128 are modern very high density coding schemes. They have three coding schemes each and permit the inclusion of special characters not present on the keyboard. If no coding scheme is specified scheme B is used by default unless the automatic conversion option (see below) is chosen. For EAN/UCC-128 scheme C is used for any code, which has numbers in the first four digits (as recommended by the ANA). An alternative scheme may be selected by making the first character one of the start characters specified below.



EAN/UCC-128



Code-128

The special characters may be entered as <ALT>0XYZ, where XYZ is the 3 digit ASCII code (+128 for values <32), or according to the following table:

XYZ	character	Code A	Code B	Code C
197	Å	DEL		
198	Æ	func. 3	func. 3	
199	Ç	func. 2	func. 2	
200	È	shift	shift	
201	É	code C	code C	
202	Ê	code B	func. 4	code B
203	Ë	func. 4	code A	code A
204	Ì	func. 1	func. 1	func. 1
205	Í	Start A	Start A	Start A
206	Î	Start B	Start B	Start B
207	Ï	Start C	Start C	Start C
208	Ð	NUL		

Code C codes only the digit pairs 00-99.

Note that EAN/UCC-128 codes have parentheses removed before coding, so (and) may appear in the human readable form but will be omitted from the barcode. Parentheses may not be used as part of the code data.

Spaces may be stripped from the text provided for input by checking the EXTRA1 checkbox in applications, or setting the flags parameter bit DL_FLAG_EXTRA1 (bit 4 of the flags variable) or BarCode.Extra1 in the OCX. This allows spaces to appear in the text under the symbol while not being included in the symbol itself.

dBarcode normally provides the control codes for switching between subtypes automatically, but this facility can be turned off by checking the EXTRA2 checkbox in applications, or setting the flags parameter bit DL_FLAG_EXTRA2 (bit 5 of the flags variable) or BarCode.Extra2 in the OCX. When dBarcode is to provide control codes no additional control codes should be provided by the user, although an initial Start A or Start B code may be given if it is desired to force the symbol to start in a particular code type. Users should note that if this option is chosen then the barcode produced may not appear identical to a sample obtained from another source - although it will scan to produce the same characters.

EAN/UCC-128 defines the use of Application Identifiers (AIs) – which are numbers with a predefined meaning and usually enclosed in brackets in the human readable form. While many AIs are followed by fixed length strings, some may be followed by a variable length string – in which case the string is terminated with a Function 1 character.

The majority of support calls result from users not using the correct 128 code variants (ie. A, B or C) or not being aware of which code variant a customer is expecting. Some customers expect only Code C, while others start in Code A and then switch to Code C, etc. It is important to be aware that the three code variants exist and will commonly be encountered within the same barcode. For this reason it is essential to ascertain which type the customer wants and if and where the code variant should change along the barcode.

Users of Code 128/EAN 128 should note that while there is a nominal size for these symbols (31.8 mm high and 11*n+2 mm long, where n is the number of characters including control codes), many applications of these codes use recommended sizes of between 50% and 84% of nominal.

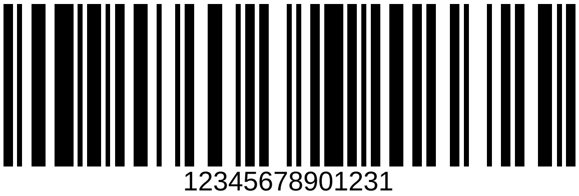
DUN-14

DUN-14 is an older name for the EAN-14 barcode type.

EAN/UCC-14

EAN/UCC-14 barcodes may be represented by ITF or EAN/UCC128 barcodes, and modern implementations should use EAN/UCC-128 – so that is what dBarcode uses. EAN/UCC-14 barcodes may be constructed from 12/13 digit retail UPC/EAN barcode numbers by left-filling the numbers with zeros, and uses a special checkdigit.

EAN/UCC-14 in EAN/UCC-128 barcode formed may be created by providing 13 digits; the Logistical Variant digit (normally 0 in the UK) followed by the first 12 of the retail digits (eg. from EAN-13 numbers but without EAN's check digit). Creation from UPC-A numbers requires two 0s followed by the first 11 digits of the UPC-A barcode number without the check digit. dBarcode calculates the EAN/UCC-14 check digit if Auto-checkdigit is enabled, and then produces the EAN/UCC-128 barcode.



If the EAN/UCC-14 checkdigit is to be provided the 14 digits are required, and Auto-checkdigit should be disabled.

SSCC

The Serial Shipping Container Code is a unique identification of individual shipping containers. The standard includes a unique barcode symbology, EAN/UCC-128, using the EAN/UCC Application Identifier Standard.

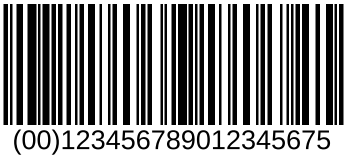
The SSCC uses an 18 digit number which consists of:

a) a single extension digit assigned by the company that constructs the SSCC

b) the UCC/EAN company prefix. Those assigned by UCC are prefixed with 0.

c) a serial reference number that must remain unique for at least 12 months

d) a single Mod 10 check digit.



When an SSCC barcode is generated using dLSoft components the data is prefixed by the (00) Application Identifier. The Mod 10 check digit may be generated by selecting the Auto-check digit option. The Show check digit option is ignored, as the Mod 10 check digit must always be shown in the human readable form.

UPC

The UPC (Universal Product Code) is widely used in the USA as a retail code. However, it has wider application and this can result in some confusion.

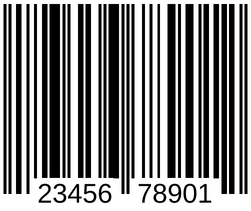
The actual UPC code is a 10 digit code. The 10 digit number is preceded by a "number system" digit, which is 0 for the retail version, and followed by a check digit. In many retail systems only the 10 digits of the UPC code need to be entered in the event of a mis-scan, so there have been times when the leading 0 has not been included in the human readable form. However, other values of the number system digit are used for specific purposes (eg. 6 or 7 are used for manufacturing identification numbering, 3 for drug products, etc.).

The UPC-A code is one variant of a number of 12 digit codes widely used in the USA. Retail codes are usually thought of as those with 10 digits (or 11 if the checkdigit is being entered explicitly), and in fact are 12 digit codes made up of a leading 0, followed by 10 product digits and 1 checkdigit.

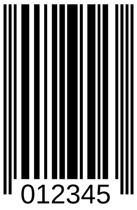
The library generates the barcode images for UPC-A if the leading 0 is provided, followed by the 10 digit product code. The check digit may either be entered explicitly or calculated by the library. This technique allows alternative leading digits to be used for their intended purposes. Users of such alternative codes will know what those leading digits may be, or may obtain the information from the authorised code provider.

The UPCC has produced more than one specification of the UPC codes. The current specification suggests that the country code (always 0 in the USA) and the codes checkdigit should be printed aligned with the coded digits, but in the light margins. Earlier specifications suggested that these digits should be printed in different positions or not at all.

The library offers the choice of not printing the digits or of printing them in the light margins (using the Margin Indicators ON option) for both UPC-A and UPC-E codes.



UPC-A

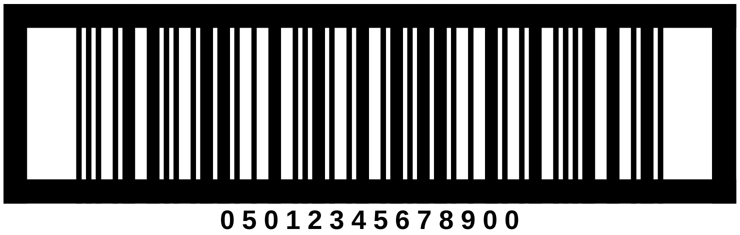


UPC-E

UPC codes support 2 and 5 digit supplementaries.

ITF

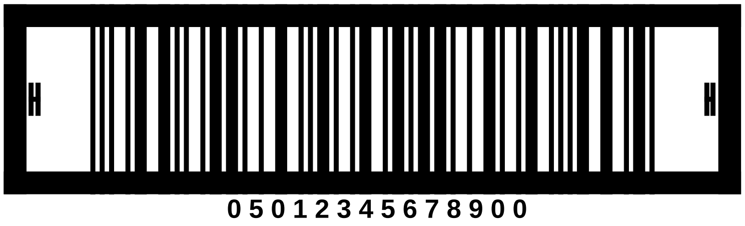
ITF is a larger code intended for use on the outside of packing cases and scanning a distance. In this form it most commonly uses the same data as EAN-13 but with a LEADING 0. If a check digit is calculated by dBarcode for this code then the EAN-13 check digit is produced.



(shown reduced in size)

The horizontal bars supporting the bars of the barcode are called Bearer Bars, and these are recommended rather than mandatory.

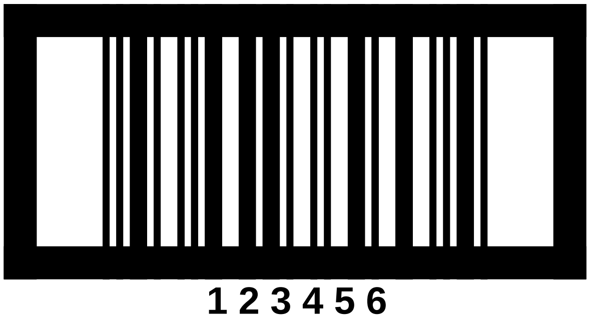
dBarcode can produce the bearer bars at any size, although the normal size recommended is about 5 mm. This size is largely historical and allowed for the spreading of pressure during metal plate printing; the actual size has no effect on the scanning of the barcode. An Extra option is provided for producing the vertical bearer bars at the same thickness and these may be obtained by checking the EXTRA1 checkbox in applications, or setting the flags parameter bit DL_FLAG_EXTRA1 (bit 4 of the flags variable) or BarCode.Extra1 in the OCX.



(shown reduced in size)

Similarly dBarcode does not normally include the optional H printer gauge marks (nor the accompanying extra light margin space), because these were also features of older printing technologies, designed to check for impression depth and ink spread. If the H gauges are required they may be obtained by checking the EXTRA2 checkbox in applications, or setting the flags parameter bit DL_FLAG_EXTRA2 (bit 5 of the flags variable) or BarCode1.Extra2 in the OCX.

A shortened version of this code is ITF-6



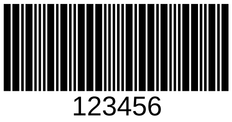
ITF-6

The ITF 6 code is not intended to have H gauges or a check digit.

Note that these ITF codes are not the same library selection as Interleaved 2 of 5 (I-2of5)

2 of 5

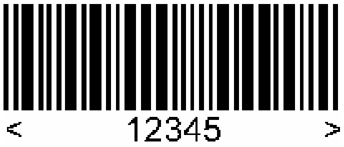
2 of 5 is a numeric only coding scheme, which is not very efficient and not recommended for new applications. 2 of 5 is also known as Standard 2 of 5 or Industrial 2 of 5.



2 of 5

IATA 2 of 5

IATA 2 of 5 is still a widely used barcode type, essentially the same as Standard/Industrial 2 of 5 but with different start and stop bars.

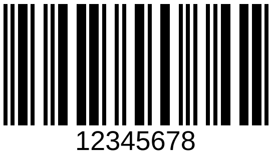


IATA 2 of 5

Most dLSoft product generate IATA 2 of 5 when Standard 2 of 5 is selected and the Extra 1 flag is set. Checking the EXTRA1 checkbox in applications, or setting the flags parameter bit DL_FLAG_EXTRA1 (bit 4 of the flags variable) in libraries sets this flag.

Interleaved 2 of 5

One of the most common codes outside the retail area is Interleaved 2 of 5, a high density, continuous numeric symbology that codes digit pairs. Because of this I-2 of 5 can only be used for even numbers of digits. If an odd number of digits is used in the DLSoft library a leading 0 is added automatically to the front of the number.



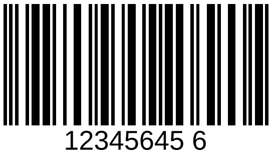
Interleaved 2 of 5

2 of 7 Code

See Codabar

Deutschen Post

The I-2 of 5 barcode also forms the basis of the German Identcode and Leitcode symbols used by Deutschen Post. These are 12 and 14 digit barcodes respectively, but they do use a different check digit calculation from the standard I-2 of 5 symbol.

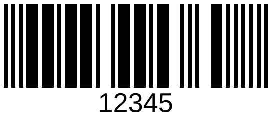


Checking the EXTRA1 checkbox in applications, or setting the flags parameter bit DL_FLAG_EXTRA1 (bit 4 of the flags variable) turns on the Deutschen Post check digit when auto-check digit is also on, enabling the I-2 of 5 barcode image to be used as Identcode and Leitcode symbols.

Note that spaces and periods are removed from the strings supplied for I-2 of 5 barcodes before the barcode image is created – so the correct layout for Identcode and Leitcode text may be used to create the symbols.

Code B

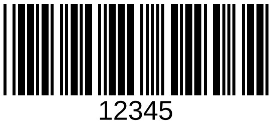
Code B is a “basic” numeric only code, which is fairly efficient in use of space.



Code B

Code 39

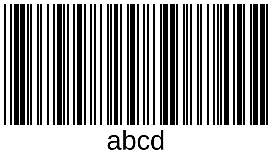
Code 39 is by far the most common barcode scheme outside the retail area and is read by most scanners, although it is not as compact as Code 93 or Code 128. The normal Code 39 scheme encodes both numbers and upper case letters, and was the first alphanumeric symbology:



Standard Code 39

Code 39 has an optional checkdigit.

The Extended Code 39 scheme also includes the lower case letters and much punctuation.



Extended Code 39

It should be noted that Extended Code 39 represents most of the additional characters by using two characters from the standard Code 39 character set. Consequently Extended Code 39 symbols are about twice as long as standard Code 39 symbols.

Code 39 is a discrete symbology - so the gap between ciphers may be larger than a unit space. Some users mistake that inter-cipher gap for a space and become concerned because it is not the same size as in another barcode representing the same characters. There is no substitute for testing the barcode with a scanner!

Also the start and stop characters are the same, and sometimes may be represented in the human readable form by an asterisk.

If iExtra1 is set the start and stop characters are shown as an asterisk in the human-readable form.

If iExtra1 is not set then the start and stop characters are not represented in human-readable form.

If iExtra2 is set an inter-cipher gap of 1 unit is used. If iExtra2 is not set there is no inter-cipher gap.

Code 93

Code 93 was designed to complement Code 39 and is a more compact code than the latter. The library supports both the standard Code 93 (numbers and upper-case letters) and the Extended (full ASCII) Code 93.

Code 93

Code 11

Code 11 is an older numeric code that is used by a number of large organisations, but is rarely found on retail scanners.

Code 11

Japan Post

The Japan Post barcode is a clocked barcode similar in appearance to 4 State code, with a mod 19 checkdigit. The elements are normally reproduced at 8, 9,10, or 11.5 point, although values between 7 and 12 point are permitted.

The symbol will accept digits and uppercase letters and the hyphen. The data consists of a 7 digit postal code plus address data. If the address data is less than 13 characters the remaining character positions are filled with control characters to make the length 20.

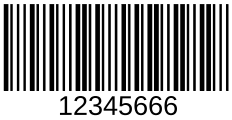
The postal code section may have a hyphen at the 4th character position (eg. 123-4567) although this hyphen does not appear in the encoded data. There may also be a hyphen between the postal code and the address data (eg. 154-0023-1-3-2-A-507). Again this hyphen does not appear in the encoded data. Note that the remaining hyphens are encoded.

MSI

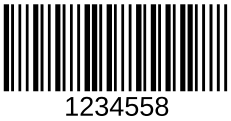
MSI, also known as the Modified Plessey Code, is a relatively weak code that is inefficient in use of space.

MSI with single checkdigit

Normally this code has a single Modulo 10 check digit. However, there are two variations of a double check digit form in common use. One uses a Mod 11 check digit before the normal Mod 10 check digit, the other uses two Mod 10 check digits.



MSI with extra Mod 10 checkdigit



MSI with extra Mod 11 checkdigit

These two-checkdigit forms are accessible through the use of the Extra1 or Extra2 parameters - ie by either checking the Extra1 or Extra2 check boxes in applications, or setting the DL_FLAG_EXTRA1 (bit 4) or DL_FLAG_EXTRA2 (bit 5) flags in the DLL, or by setting the BarCode1.Extra1 or BarCode1.Extra2 parameters in the OCX. The effects are as shown below. Note that BOTH options also require the autocheckdigit calculation to be enabled.

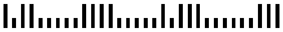
If Extra1 is set then a Modulo 10 check digit is calculated and inserted before the normal checkdigit.

If Extra2 is set then a Modulo 11 check digit is calculated and inserted before the normal checkdigit.

Some scanning equipment cannot read both forms. (In fact some scanning equipment cannot read either of the two checkdigit forms). Check your scanners documentation to ensure that you choose an appropriate combination. DO NOT SET BOTH Extra1 and Extra2.

PostNet

PostNet codes are the clocked codes used in the US mail system. There are three types of PostNet code (identified as A, C and C'), which differ in the number of characters encoded. These codes are based on the US ZIP code system.

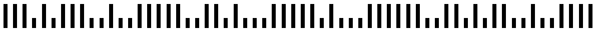


PostNet A

The dBarcode library also allows the creation of the US postal FIM symbols – FIM A, FIM B and FIM C. There is no text content associated with these codes.

Planet

Planet codes are clocked codes used within the US Postal Service for the confirmation of incoming or outgoing mail.

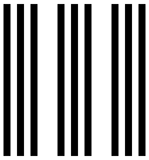


Planet_OC

Planet codes consist of 9 or 11 digits prefixed by a code to indicate which Origin Confirm or Destination Confirm service is required, and postfixed by a mandatory check digit. The dLSoft library automatically provides the check digit. There is no text content associated with these codes.

Swiss Post

The related Strichcode for A-Post and B-Post as used by the Swiss Postal Service is also supported. The B-Post symbol is produced by checking the EXTRA1 checkbox in applications, or setting the flags parameter bit DL_FLAG_EXTRA1 (bit 4 of the flags variable).



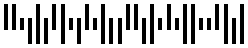
There is no text content associated with these codes.

RM4SCC

RM4SCC is the Royal Mail (UK) version of the 4 State clocked barcode used for directing mail. The codes contain a start and stop bit, while the 4 State code (below) does not. While both codes offer the option of a checkdigit, it should be noted that the Royal Mail code must include the checkdigit (which should be calculated automatically).

These codes are based on the UK Post Code system, but may also contain an International Prefix and a Delivery Point Suffix.

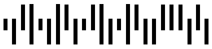
Note that in both RM4SCC and 4 State (see below) all characters are converted to upper case prior to encoding and any illegal characters with ASCII codes >32 are converted to X. Illegal characters with ASCII codes <= 32 are ignored -- so spaces and carriage returns are ignored.



RM4SCC

4 State

4 State is similar to the RM4SCC code and is used in some European countries without the start and stop bits and in some cases without the Checkdigit. This code is referred to as 4 State.



4 State

Australia Post

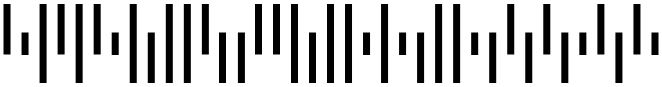
The Australian Postal Service uses a variant of the 4 State code with Reed-Solomon Error Correction, and four versions of the code are supported:

Standard Customer code: by unchecking both the EXTRA1 and IEXTRA2 checkboxes in applications, or clearing the flags parameter bits DL_FLAG_EXTRA1 (bit 4 of the flags variable) and DL_FLAG_EXTRA2 (bit 5 of the flags variable).

Customer 1: by checking the EXTRA1 checkbox in applications, or setting the flags parameter bit DL_FLAG_EXTRA1 (bit 4 of the flags variable).

Customer 2: by checking the EXTRA2 checkbox in applications, or setting the flags parameter bit DL_FLAG_EXTRA2 (bit 5 of the flags variable).

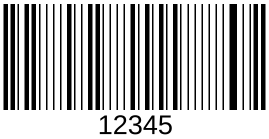
Reply Paid coupon: by checking both the EXTRA1 and IEXTRA2 checkboxes in applications, or setting the flags parameter bits DL_FLAG_EXTRA1 (bit 4 of the flags variable) and DL_FLAG_EXTRA2 (bit 5 of the flags variable).



Australia Post

Plessey

An older code still popular in some industries, the Plessey code supports numbers and the characters X, B, C, D, E and F, plus a two character crc check.

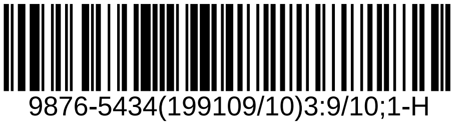


Plessey

It is common practice in some industries using Plessey barcodes to separate the barcode characters from their checkdigits. This can be done by setting the EXTRA2 flag in applications or setting the flags parameter bit DL_FLAG_EXTRA2 (bit 5 of the flags variable), or by manually including a space character at the end of the barcode data.

SISAC

Unlike most other barcodes the SISAC barcode symbol does not have a one-to-one correspondence with the SICI code printed underneath it. DLSoft barcode products generates the SISAC barcode from the SICI code, and it can only do this if the SICI code itself is correct. If the SICI code is not correct then the library will report error number 10.



The SICI code must be entered into the Code edit box or supplied as a database field, and it **must contain at least the following items:**

The ISSN number complete with a hyphen between digits 4 and 5, e.g. 1234-5678

A date item enclosed in brackets. If no date item is required the () symbols MUST STILL BE PRESENT.

A number item is optional. e.g. 14:1

Index or supplement numbers are optional, e.g. *1

The standard version number which is currently ;1- and all three characters MUST BE PRESENT

A SICI check digit may immediately follow the - of the version number. The check digit may be entered manually or may be calculated by dBarcode by enabling Auto Checksum.

A typical SICI code is thus: 9876-5432(199109/10)3:9/10;1-

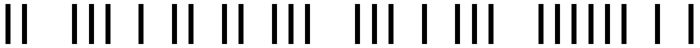
and the check digit (H in this case) may be added automatically.

Note: When copying SICI codes from publications it is not always easy to distinguish the : and ; characters. SICI codes ALWAYS end with (semicolon) ;1-n where n is a check digit.

DLSoft barcode products do not currently support SICI location codes.

Korean Postal Authority

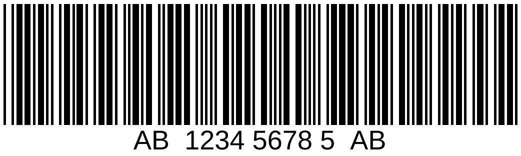
The Korean Postal Authority code is a clocked code consisting of a 6 digit Zip code plus a single parity digit.



The Zip code may be provided with a dash between the first three and last three digits. The dash is not encoded and a human readable form is not included under the barcode.

Italian Postal 3/9

The Italian Postal 3/9 code is encoded using Code 39 symbology, except that the check digit is calculated using a different algorithm from that used by Code 39.



Italian Postal 3/9

The code consists of

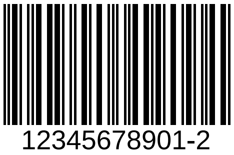
two alphabetical characters
eight digits

one check digit
two alphabetical characters.

The dLSoft library calculates the check digit when required, and inserts spaces between the components of the code. The data may be supplied without spaces.

Italian Postal 2/5

The Italian Postal 2/5 code is encoded using Interleaved 2 of 5 symbology, except that the check digit is calculated using a different algorithm from that used by normal Interleaved 2/5.



Italian Postal 2/5

The code consists of

eleven digits
one check digit

The dLSoft library calculates the check digit when required, and inserts a dash or a space between the components of the code. A dash is normally inserted, but a space may be inserted by checking the EXTRA1 checkbox in applications, or setting the flags parameter bit DL_FLAG_EXTRA1 (bit 4 of the flags variable). The data may be supplied without a space/dash.

USD-4

See Codabar

EAN and UCC barcodes

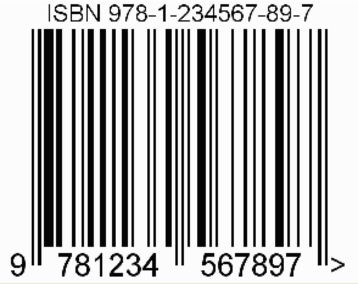
EAN International (European Article Numbering International) and the United States UCC (Uniform Coding Council) have agreed on common barcoding standards. At the time of writing a number of dLSoft barcoding products display the name of only one barcode type although the type is also known by another name.

Thus EAN-14 is the same as UCC-14, and is referred to as EAN/UCC-14, and EAN-128 is the same as UCC-128, and referred to as EAN/UCC-128.

As products are updated the display names will be updated.

ISBN barcode changes

In January 2007 (or soon thereafter), the US ISBN Agency will begin assigning 13-digit identifiers to books and book-related products. These identifiers, "ISBN-13s", will incorporate the existing Bookland EAN into a new book identifier where the prefix can be either '978' (current value) or '979'.



ISBN-13

dLSoft Barcode components will automatically recognize ISBN-13 barcode data starting with 978 or 979 and generate the appropriate ISBN-13 barcode.

dLSoft Barcode components will generate ISBN-13 barcodes using the ‘978’ prefix from data supplied as the ISBN 10 data, if the EXTRA1 property is set to TRUE; otherwise it will generate an ISBN-10 barcode as before.

ISBN with 2 and 5 digit supplementaries are supported by some dLSoft products. The supplementary characters must be separated from the ISBN numbers with a / character.

Location numbering

In 1995, EAN International has agreed that all numbering organisations will standardise on the product numbering (ie standard EAN-13) check digit algorithm when calculating check digits for location numbers. The cut-off date for using older check digit algorithms was January 1st 1997.

dLSoft products will produce the recommended check digits and will not produce older variants.

2D Barcodes

dLSoft 2D barcode software include products that support the generation of barcode images for the following types of two-dimensional barcodes:

Code 16k
Code 49
PDF417
MicroPDF417
Aztec code
Data Matrix
RSS
Codablock F

This document describes some of the important features of these barcode types and the character sets they support.

Details of the structure and translation algorithms are beyond the scope of this document, and users are referred to the technical documentation available from the following sources:

AIM
1326 Freeport Road
Pittsburgh
PA 15238
USA

Information on PDF417 is also available from

Symbol Technologies Inc.
116 Wilbur Place
Bohemia
NY 11716
USA

Information on the Aztec barcode is also available from

Welch Allyn
Data Collection Division
4619 Jordan Road
PO Box 187
Skaneateles Falls
NY 13153-0187
USA

2D Barcode types supported

Current versions of the dBarcode-2D Dynamic Link Library, dBarcode.NET 2D Component, and Active 2D-Barcode Component support the following code types:

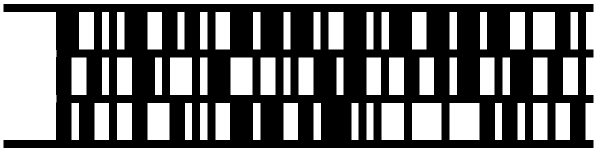
dBarcode Code type	dBarcode.NET type	Active Barcode Component Code Type	Barcode type

0		-	Code 16k
1		-	Code 49
2	3	2	PDF417
3	0	0	Aztec
4	1	1	DataMatrix
5	2	3	Maxicode
6	4	4	MicroPDF417
7	5	5	RSS
8		-	Coablock F
9	6	6	QR Code

Note that most common 1D barcodes are supported by the standard dBarcode library (DLSBAR32.DLL) and Active Barcode Component 1D Universal.

Code 16k

Code 16k is a stacked barcode of the type illustrated below



Code 16k encodes characters using a reverse video version of Code 128. Each row starts and ends with a UPC digit, which indicates the row number and the direction of scan.

Each row encodes 5 data characters and the Code 128 type C mode is used for numeric-only strings. A maximum of 16 rows are permitted.

Code 16k has three coding schemes and permit the inclusion of special characters not present on the keyboard. If no coding scheme is specified scheme B is used by default. Scheme C is used for any code, which has numbers in the first four digits. An alternative scheme may be selected within dBarcode-2D from the allowed modes:

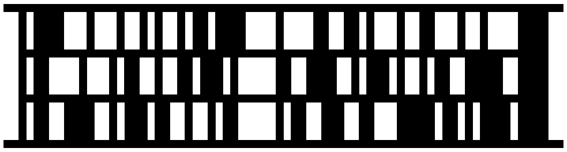
Start mode	Code set	leading char
0	A	
1	B	
2	C	
3	B	Fnc1
4	C	Fnc1
5	C	Shift B
6	C	Double Shift B

Other function and shift characters are as specified below (and may be typed as <ALT>0XYZ).

XYZ	character	Code A	Code B	Code C
197	Å	DEL		
198	Æ	func. 3	func. 3	
199	Ç	func. 2	func. 2	
200	È	shift	shift	
201	É	code C	code C	
202	Ê	code B	func. 4	code B
203	Ë	func. 4	code A	code A
204	Ì	func. 1	func. 1	func. 1
205	Í	Start A	Start A	Start A
206	Î	Start B	Start B	Start B
207	Ï	Start C	Start C	Start C
208	Ð	NUL		

Code 49

A code 49 symbol is a stacked symbol containing between 2 and 8 rows, each separated by a separator bar.



Each row contains 16 “words” (which are generated from character pairs) and a start and stop character. The last row also contains the number of rows in the symbol and the check digit characters.

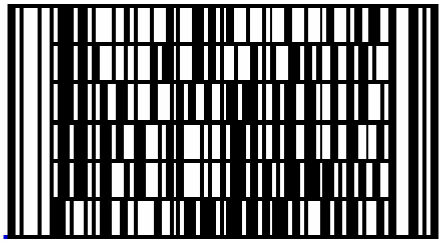
There are 2400 possible words which can be generated from each pair of characters (by taking the value of the right hand character plus 49 time the value of the left hand character). Symbols with less than 7 rows contain 2 check digits in the final row. Symbols with 7 or 8 rows contain 3 check digits in the final row.

A numeric mode allows 5 digits to be encoded in the same space as three alphanumeric characters, so offering a higher density.

Encoding code 49 symbols may be accomplished in Auto mode.

Codablock F

Codablock F is a stacked barcode symbology based on Code 128. It can encode the full ASCII character set in a symbol which consists of multiple rows of Code 128 type symbols, using a common “Start A” start character and a common “Stop” stop character. Apart from the start and stop character the other characters in adjacent rows have a horizontal line between them.



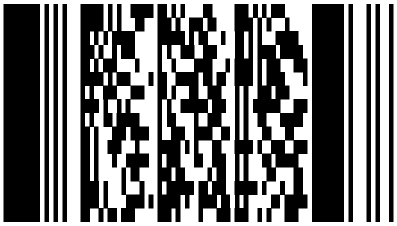
Each row in a Codablock symbol contains (in addition to the common start and stop characters) a subset selector, a row indicator and a check digit, along with a number of data characters. This enables each row to be read by Code 128 scanners.

Codablock symbols can hold a maximum of 2725 characters, although this value is significantly reduced by the presence of subset change characters. Codablock symbols can have between 2 and 44 rows, and each row can hold between 4 and 62 characters plus the subset selector, row indicator and check digit.

Codablock symbols are character self-checking and may be scanned in any direction. The subtypes used in the symbol are essential identical with those use in Code 128 symbology.

PDF417

PDF417 (Portable Data File 417) is a stacked barcode symbology capable of encoding over a kilobyte of data in a symbol.



PDF417 symbols may include extensive error-correction enabling data to be recovered from a symbol which has been damaged or corrupted.

There are 900 different patterns (codewords) which may be incorporated into a PDF417 symbol, and several modes available for encoding.

Encoding Modes

EXC (Extended Alphanumeric Compaction mode) - allows encoding of all printable ASCII characters into about 2 characters per codeword. Within the EXC mode there are several submodes:

EXC Alpha - starts in upper case
EXC Lower - starts in lower case
EXC Mixed - numeric and other punctuation
EXC Punctuation - punctuation & bracket characters.

Binary/ASCII Plus mode - allows encoding of 256 international characters including the full ASCII set plus any 8-bit value in the range 0 - 255. This mode allows encoding approximately 1.2 bytes per codeword and so is considerable less efficient than EXC mode.

Numeric mode - allows encoding of a string of digits with a density of approximately 2.95 digits per codeword. Recommended where more than 13 digits are to be encoded. Numeric mode symbols may not contain non-numeric characters.

The maximum capacity of PDF417 symbols is approximately as follows:

Numeric mode – maximum capacity 2700 digits*
Alphanumeric text – maximum capacity 1800 characters*
Byte values – maximum capacity 1100 bytes

*Note that capital letters, punctuation, mode shifts and new lines/paragraphs reduce these value.

Security level

One of PDF417's most valuable features is its ability to allow correction of errors. This ability is provided by the inclusion of additional codewords within the symbol, so that the data codewords may be reconstructed even if some are defaced or misread.

The number of damaged codewords (Nmax) which may be recovered depends on the security level (which in turn determines the number of additional caharacters included in the symbol)

Security level	Nmax
0	0
1	2
2	6
3	14
4	30
5	62
6	126
7	154
8	510

Aspect ratio

PDF417 symbols may be reproduced with different number of codewords per row, and therefore in several different height to width ratios.

The user may select the target ratio (as either the height to width ratio or the number of rows/number of codewords per row). In general tall, thin barcodes read more reliable than short, wide symbols, although there may be restrictions applied by the type of barcode scanner employed.

The dBarcode library will attempt to form a symbol which meets a target height to width ratio, but users should note that some things are just not possible!

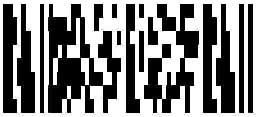
The following character translations are made if the Flag is set (the flag checkbox in applications or the AztecFlag property in components)

String	converted to
{ET}	ASCII 4
{FS}	ASCII 28
{GS}	ASCII 29
{RS}	ASCII 30

Appended symbols are not supported at this time.

MicroPDF417

MicroPDF417 is a multi-row symbology based on PDF417 designed for applications requiring a greater area efficiency but lower data capacity than PDF417. A specific and limited set of symbol sizes is available, each size including a fixed level of error correction.



MicroPDF417 provides for three encoding modes: Text, Byte and Numeric compaction. Text is for general text, Numeric for encoding data consisting only of digits, and Byte to allow for the first 127 ASCII characters but with a reduced level of efficiency. Four symbol widths are permitted, each specifying the number of data columns (1 – 4). Within each symbol width a variable number of rows provide for a maximum data capacity of:

Text compaction mode 0: 250 characters (2 data characters per codeword)

Byte compaction mode 1: 150 characters (1.2 data characters per codeword)

Numeric compaction mode 2: 366 characters (2.93 data characters per codeword)

The following additional parameters are allowed for each symbol:

Level/Columns: Number of data columns in the symbol. (1 – 4)
X-unit: Thickness of thinnest bar in mils (4 – 200)
Y-multiplier: Height of each bar in X-units (2 – 5)

The industry standard Macro sequences [)><RS>05<GS> and [)><RS>06<GS> are supported (where <RS> represents ASCII 30 and <GS> is ASCII 29).

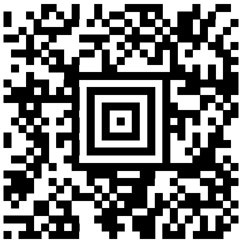
The following character translations are made if the Flag is set (the flag checkbox in applications or the AztecFlag property in components)

String	converted to
<ET>	ASCII 4
<FS>	ASCII 28
<GS>	ASCII 29
<RS>	ASCII 30
<US>	ASCII 31

Appended symbols are not supported at this time.

Aztec

Aztec is a matrix symbology which supports the entire ASCII character set and offers several error checking modes. Aztec symbols are square and may be read at any orientation.



The symbol is made up of squares which "grow" from the centre around a centre mark. The size of the symbol is characterised by the number of "layers" outside the centre mark, and this can range from 1 – 32.

Security and Layers

For Aztec symbols in Normal mode the user may select any percentage of the symbol to contain error checking data within the range 1 – 99%. This is the Security Level in dBarcode-2D. If a value of 0% is used then the symbol will actually be produced with the default amount of error correction (which is 23% + 3 codewords). The higher the security level the greater will be the number of layers required to contain the symbol – and hence its overall size.

In Compact mode the user specifies the number of layers used to contain the symbols information – and this is restricted to the range 1 – 4. dBarcode will insert error correction data in the space available.

In Full range mode again the user specifies the number of layers allowed for the symbol, but this time the range is 1 – 32. Again error correction data is used to fill otherwise unused space in the specified number of layers.

In Full range mode the maximum capacity of Aztec Code symbols is approximately as follows:

Digits only – maximum capacity 3800 digits
Alphanumeric text – maximum capacity 3000 characters*
Byte values – maximum capacity 1900 bytes

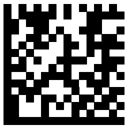
*Note that capital letters, punctuation and new lines/paragraphs reduce this value.

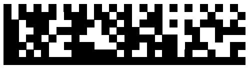
Runes are special Aztec symbols which consist of a single layer and have no security options.

Appended symbols are not supported at this time.

DataMatrix

DataMatrix is a two dimensional matrix symbology which is made up of square modules arranged within a finder pattern. DataMatrix symbols may be square or rectangular.





Security

DataMatrix symbols can include a user-selected amount of error-correction data.

For Square symbols the Security Level setting may be in the range 0 – 24, while for Rectangular symbols the allowed range is 0 – 6.

DataMatrix symbols can encode the entire ASCII character set and uses multiple encoding modes which are, in order of efficiency:

Double digits – maximum capacity 3100 digits
Alphanumeric text – maximum capacity 2300 characters*
Byte values – maximum capacity 1550 bytes

*Note that capital letters, punctuation and new lines/paragraphs reduce this value.

dLSoft 2D barcode products support only ECC200 symbols – the older ECC140 and below are not supported. Also at this time ECI and Appended symbols are not supported.

If the first data character is FNC1 (ASCII 232) the remaining data is treated as EAN/UCC data; i.e. brackets are removed from the data. Any AIs included in the data that are NOT fixed length should be terminated with a FNC1 character. (The FNC1 character may be entered from the keyboard by holding down the <Alt> key and typing 0232 on the numeric keypad.)

The industry standard Macro sequences [)><RS>05<GS> and [)><RS>06<GS> are supported (where <RS> represents ASCII 30 and <GS> is ASCII 29).

The following character translations are made for data following the [)> start sequence:

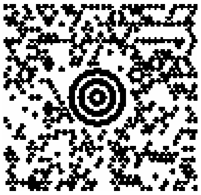
String	converted to
<ET>	ASCII 4
<FS>	ASCII 28
<GS>	ASCII 29
<RS>	ASCII 30
<US>	ASCII 31

MaxiCode

MaxiCode is not really a barcode – it doesn't have any bars. It is a fixed size matrix symbology made up of offset rows of hexagonal modules, with a Finder pattern in the centre.

MaxiCode symbols are reproduced at a width of 25.5 mm and a height of 24.4 mm (in each case there is a tolerance of 1.5 mm).

Intended primarily for encoding addresses for postal and delivery applications, it has a relatively low information content. On the other hand the fixed physical size of the image simplifies the facilities required for both printing and scanning the symbols.



MaxiCode is characterised by the two "messages" which make up its data content – the Primary message and Secondary message. The symbology may be used in 5 modes which utilise the two message in different ways and with different levels of error correction, as shown in the table below:

Mode	Use
0	obsolete (not supported)
1	obsolete (not supported)
2	Primary message encode numeric postal code, country code and service code, secondary message encodes additional data
3	Primary message encode alphanumeric postal code, country code and service code, secondary message encodes additional data
4	Any data up to 84 data characters automatically split between primary and secondary messages.
5	Any data up to 68 data characters automatically split between primary and secondary messages. Enhanced error correction used
6	Similar to 4, but used for reader control.

In modes 4, 5, and 6 virtually any ASCII data may be encoded up to the maximum number of data characters allowed. However, encoding introduces additional shift and latch characters whenever characters other than uppercase letters and numbers are used and the maximum length of text becomes correspondingly smaller.

Modes 2 and 3 are for Structured Carrier Messages and require specific data in the correct order to produce a scan able symbol. Basically the data consists of the sequence:

postal code G country code G service class G secondary message

where the 4 components are separated by ASCII 29 characters, shown as G.

In mode 2 the postal code must be numeric only and up to 9 digits. In mode 3 the postal code may be up to 6 alphanumeric characters. The country code and service class elements must be three digits each.

Mode 2 or 3 message which begin with the sever character sequence [)>R01G

(where G is ASCII 29 and R is ASCII 30) are treated in the special way described in the MaxiCode specification.

Any secondary message can terminated with an End of Transmission character (ASCII 4).

Because the non-printable ASCII characters are a required part of MaxiCode messages in Modes 2 and 3, dLSoft software recognises the following keyboard sequences for the entry of such characters:

Sequence	Alternative	ASCII value
<FS>	{FS}	28
<GS>	{GS}	29
<RS>	{RS}	30
<US>	{US}	31
<ET>	{ET}	4

A typical string for a Mode 2 Maxicode is:

[)><RS>01<GS>96152382802<GS>840<GS>001<GS>1Z00004951<GS>UPSN<GS>06X610<GS>159<GS>1234567<GS>1/1<GS><GS>Y<GS>634 ALPHA DR<GS>PITTSBURGH<GS>PA<RS><ET>

Appended symbols are not supported by dLSoft software at this time.

MaxiCode symbols are sensitive to the gap between hexagons. For printing on a 300 dpi laser we recommend the line width reduction parameter be set to 12.

QR Code

QR Code is a matrix symbology consisting of a square array of modules with a finder pattern located at three corners. A wide range of symbol sizes is supported along with four levels of error correction, and the symbology is noted for its high data density.

The dLSoft libraries support QR Code Version 2 in Numeric, Alphanumeric and Byte modes. (Kanji mode is not supported at this time).

The maximum data capacity of the symbols (at Level 0 error correction) are:

Numeric data	7089 digits
Alphanumeric data	4296 characters
Byte data	2953 bytes

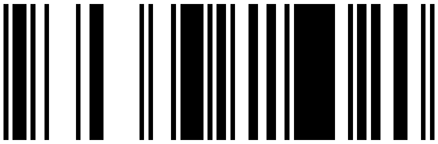
Note that Alphanumeric includes digits and uppercase letters, the space and the $ % * + - . / : characters only.

The Reed-Solomon error correction allows the following recovery of damaged codewords:

Level 0: 7%
Level 1: 15%
Level 2: 25%
Level 3: 30%

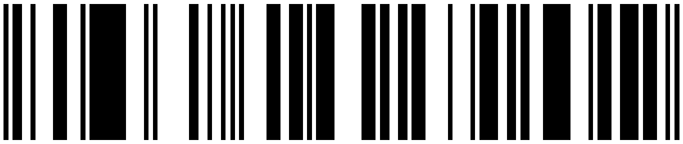
Reduced Space Symbology

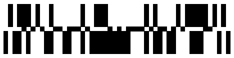
The EAN.UCC Reduced Space Symbology (RSS) is a family containing three linear symbologies and three stacked variants.



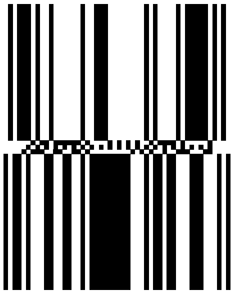
RSS-14 encodes the full 14 digit EAN.UCC item identification in a linear symbol that can be scanned in any direction. RSS Limited encodes item identification suitable for use on small items, but not for use at point-of-sale.

RSS-Expanded encodes the 14 digit EAN.UCC item identification plus supplementary AI elements.





RSS-14 Stacked is a variant which is stacked in two rows, either as a truncated version (above) used for small item marking, or as an omni-directional version (below) designed to be read by omnidirectional scanners.



The RSS-14 based versions encode the full 14 digit EAN.UCC item identification number and this may be prefixed with the (01) AI – although this is NOT encoded.

The dLSoft RSS implementation uses the following Start Mode property values to specify the symbol:

Start Mode	RSS version
0	RSS-14
1	RSS-14 Truncated
2	RSS Limited
3	RSS-14 Stacked
4	RSS-14 Stacked Omnidirectional
5	RSS Expanded
6	RSS Expanded Stacked

All RSS symbols are based on Xunit size, where the Yunit (height) is treated as a multiplier of the Xunit value.

The following height values should be used:

RSS-14: a minimum height of 33 X

RSS Truncated: a minimum height of 13 X

RSS-14 Stacked: a fixed height (top row 5X, bottom row 7X, separator 1X)

RSS-14 Stacked Omnidirectional: minimum height 33X (each row)

RSS Limited: minimum height 10 X.

RSS Expanded: minimum height 34 X

RSS Expanded may also be used in a stacked version, and the extent of stacking may be specified using the Aspect Ratio property, which can range be set between 1 and 10, where the value represents the width of the symbol in segment pairs. Again each row shall have a minimum height of 34 X.

RSS Expanded may be used to encode AIs in addition to the item identification number as listed below:

(01)………(xx)………

(01)………(3103)…… - 6 digit metric weight

(01)………(3202)…… - 6 digit pound weight

(01)………(392x)…… - price (variable length)

(01)………(393x)…… - 3 digit ISO currency code followed by price

(01)..……..(310x)…….(yy)…….. – metric weight and six digit date

(01)…..…..(320x)…….(yy)…….. – English weight and six digit date

(where yy may be 11 (production date), 13 (packaging date), 15 (best before date) or 17 (expiration date).

Variable length AI data must be terminated using a FNC1 character (#).

For Expanded types the Level setting value (default=0) is used as the Linkage. (At the present time the dLSoft library does not support linked barcode symbols, but the Linkage parameter may be used by developers wishing to support linked symbols.)

Pattern strings

For many reasons a developer may prefer to draw the bars to generate the barcode, rather than use the metafile image returned by the dBarcode-2D library.

To do this use the “pattern” string returned from the DLL or OCX.

The choice of size for the drawn bars is a matter for the developer. However, some features of the drawn image should be considered in relation to the notes below for the different barcode types supported. In particular note that some stacked barcode types require separator lines to be drawn between each row. In general separator lines are drawn with the thickness of a single unit (the same as the thinnest bar in the symbol).

Code 16k strings

The pattern string returned for a Code 16k barcode consists of 0s and 1s, where the 0s represent bars and the 1s represent spaces. Each row of the barcode image terminates with a newline character (\n or CHR$(10)). The stacked barcode separators extend 7 units in front of the first bar and 1 unit after the last bar and the spaces, which allow for this are included in the pattern string. Separator lines are also required above the first row and below the last row.

Code 49 strings

The pattern string returned for a Code 49 barcode consists of 0s and 1s, where the 1s represent bars and the 0s represent spaces. Each row of the barcode image terminates with a newline character (\n or CHR$(10)). The stacked barcode separators extend 2 units in front of the first bar and 2 units after the last bar, but the spaces, which allow for this are NOT included in the pattern string. Separator lines are also required above the first row and below the last row.

PDF417 strings

The pattern strings returned for PDF417 barcodes consist of a stream of digit characters, with alternate characters in each row representing first bars, then spaces. So the pattern 82131 represents an 8 unit bar followed by a 2 unit space, a 1 unit bar, a 3 unit space and a 1 unit bar.

Each row of the barcode symbol is terminated by a newline character (\n or CHR$(10)) in the pattern string.

PDF417 symbols do not contain row separators.

Aztec strings

The pattern string returned for an Aztec barcode consists of the sequence of characters used to create the symbol, arranged by row, with a ‘1’ for a black square and a ‘0’ for a white square. Each row is separated by a newline (\n or CHR(10)) characters. Two successive zero bytes indicate the end of the pattern.

DataMatrix strings

The pattern string return for a DataMatrix barcode consists of the sequence of characters used to create the symbol, arranged by row, with a ‘1’ for a black square and a ‘0’ for a white square. Each row is separated by a newline (\n or CHR(10)) characters. Two successive zero bytes indicate the end of the pattern.

MaxiCode strings

Owing to the nature of MaxiCode, no pattern string is produced for this barcode type.

MicroPDF417 strings

The pattern string returned for MicroPDF417 has the same properties as that for normal PDF417.

QR Code strings

The pattern string return for a QR Code barcode consists of the sequence of characters used to create the symbol, arranged by row, with a ‘1’ for a black square and a ‘0’ for a white square. Each row is separated by a newline (\n or CHR(10)) characters. Two successive zero bytes indicate the end of the pattern.

RSS strings

The Pattern string for a linear RSS symbol consists of a stream of digit characters, with alternate characters representing first bars, then spaces. So the pattern 12131 represents an 1 unit bar followed by a 2 unit space, a 1 unit bar, a 3 unit space and a 1 unit bar, etc.

The Pattern strings for Stacked RSS symbols are more complex! The string consists a stream of digit characters, with each row of the symbol (including the separator patterns) being separated by a newline (\n or CHR(10)) character. The FIRST character of each row is not a part of the symbol, but represents whether the remaining characters of the row occur in the sequence bar – space – bar (for which the first character is a 1) or space – bar – space (for which the first character is a 0). The remaining characters in each row represent the length of the relevant element. So the row pattern 112131 represents an 1 unit bar followed by a 2 unit space, a 1 unit bar, a 3 unit space and a 1 unit bar, etc., while the row pattern 042131 represents an 4 unit space followed by a 2 unit bar, a 1 unit space, a 3 unit bar and a 1 unit space, etc.

The row heights are given in the RSS barcode section. Users requiring further information on the allowed sizes for RSS codes should refer to the RSS standard specification..

Notes on Metafiles

The picture images placed on the clipboard by dLSoft barcode products are ANISOTROPIC metafiles. This means that they can be resized within applications (usually by dragging a corner).

While the barcode bars can be resized over very wide ranges, any text included within the image may not resize as expected. In general changing the height of the image by resizing within another application will change the fontsize used to render the text. Changing the width of the image within another application may cause the position of any text under the barcode to change.

To overcome text size problems caused by resizing metafile images choose an alternative fontsize within the product. The use of TrueType fonts is recommended to prevent unusual effects caused by resizing of text.

When metafiles are printed the most commonly encountered problem is that the thickness of bars may be greater than expected - which results in the barcodes not scanning correctly. To overcome this problem all dLSoft barcode products allow bar thickness reduction.

When the bar thickness reduction is positive the thickness of a bar is reduced in proportion to its nominal width. So a bar that is three units wide is reduced in thickness by three times as much as a bar that is one unit wide. This technique is designed for wet-ink printing - where the amount of ink spread is proportional to the area inked.

When the bar reduction is negative the thickness of each bar is reduced by the same amount - a proportion of the smallest bar - and a bar which is three units wide is reduced by the same amount as a bar which is one unit wide. This technique can be used to overcome problems with printers that print all lines thicker than instructed - a common problem with 300 dpi laser printers.

In general if a barcode will not scan correctly the symbol should be examined under a magnifier to ensure that the thinnest space in the symbol is at least as wide as the thinnest bar. If it isn't then some reduction in bar thickness is probably required.

It should be noted that the correctness of the barcode bar thickness cannot be judged from the screen image - this will always be correct.

Index