BS ISO/IEC 16022:2024

BSI Standards Publication

Information technology — Automatic identification and data capture techniques — Data Matrix bar code symbology specification

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 16022:2024

National foreword

This British Standard is the UK implementation of ISO/IEC 16022:2024. It supersedes BS ISO/IEC 16022:2006 which is withdrawn.

The UK participation in its preparation was entrusted to Technical Committee IST/34, Automatic identification and data capture techniques.

A list of organizations represented on this committee can be obtained on request to its committee manager.

Contractual and legal considerations

This publication has been prepared in good faith, however no representation, warranty, assurance or undertaking (express or implied) is or will be made, and no responsibility or liability is or will be accepted by BSI in relation to the adequacy, accuracy, completeness or reasonableness of this publication. All and any such responsibility and liability is expressly disclaimed to the full extent permitted by the law.

This publication is provided as is, and is to be used at the recipient's own risk.

The recipient is advised to consider seeking professional guidance with respect to its use of this publication.

This publication is not intended to constitute a contract. Users are responsible for its correct application.

© The British Standards Institution 2024
Published by BSI Standards Limited 2024

ISBN 978 0 539 18349 8

ICS 01.080.50; 35.040.50

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 31 May 2024.

Amendments/corrigenda issued since publication

Date	Text affected

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 16022:2024

International Standard

ISO/IEC 16022

Third edition
2024-05

Information technology — Automatic identification and data capture techniques — Data Matrix bar code symbology specification

Technologies de l'information — Techniques automatiques d'identification et de capture des données — Spécification de symbologie de code à barres Data Matrix

Reference number
ISO/IEC 16022:2024(en)

© ISO/IEC 2024

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 16022:2024

COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2024

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 16022:2024

Contents

Page

Foreword ... **v**
Introduction ... **vi**
1 Scope ... **1**
2 Normative references ... **1**
3 Terms and definitions ... **1**
4 Symbols ... **2**
5 Mathematical or logical notations ... **2**
6 Symbol description ... **2**
6.1 Basic characteristics ... 2
6.2 Summary of additional features ... 3
6.3 Symbol structure ... 3
6.3.1 General ... 3
6.3.2 Finder pattern ... 4
6.3.3 Symbol sizes and capacities ... 4
7 Data Matrix code requirements ... **4**
7.1 Encode procedure overview ... 4
7.1.1 General ... 4
7.1.2 Step 1: data encodation ... 4
7.1.3 Step 2: error checking and correcting codeword generation ... 4
7.1.4 Step 3: module placement in matrix ... 5
7.2 Data encodation ... 5
7.2.1 Overview ... 5
7.2.2 Default character interpretation ... 5
7.2.3 ASCII encodation ... 5
7.2.4 Symbology control characters ... 6
7.2.5 C40 encodation ... 7
7.2.6 Text encodation ... 9
7.2.7 ANSI X12 encodation ... 9
7.2.8 EDIFACT encodation ... 10
7.2.9 Base 256 encodation ... 11
7.3 ECI ... 11
7.3.1 General ... 11
7.3.2 Encoding ECIs ... 12
7.3.3 ECIs and Structured Append ... 12
7.3.4 Post-decode protocol ... 12
7.4 Data Matrix symbol attributes ... 13
7.4.1 Symbol sizes and capacity ... 13
7.4.2 Insertion of Alignment Patterns into larger symbols ... 14
7.5 Structured Append ... 14
7.5.1 Basic principles ... 14
7.5.2 Symbol sequence indicator ... 14
7.5.3 File identification ... 15
7.5.4 FNC1 and Structured Append ... 15
7.5.5 Buffered and unbuffered operation ... 15
7.6 Error detection and correction ... 15
7.6.1 Reed-Solomon error correction ... 15
7.6.2 Generating the error correction codewords ... 15
7.6.3 Error correction capacity ... 16
7.7 Symbol construction ... 17
7.7.1 General ... 17
7.7.2 Symbol character placement ... 17
7.7.3 Alignment Pattern module placement ... 17

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

7.7.4 Finder Pattern module placement 18

8 Symbol dimensions 18

9 Symbol quality 18
9.1 General 18
9.2 Symbol quality parameters 18
9.2.1 Fixed pattern damage 18
9.2.2 Overall symbol grade 18
9.2.3 Decode 18
9.2.4 Grid non-uniformity 18
9.3 Process control measurements 19

10 Reference decode algorithm for Data Matrix 19

11 User guidelines 30
11.1 Human readable interpretation 30
11.2 Autodiscrimination capability 30
11.3 System considerations 30

12 Transmitted data 30
12.1 General 30
12.2 Protocol for FNC1 30
12.3 Protocol for FNC1 in the second position 30
12.4 Protocol for Macro characters in the first position 31
12.5 Protocol for ECIs 31
12.6 Symbology identifier 31
12.7 Transmitted data example 31

Annex A (normative) **Data Matrix interleaving process 33**

Annex B (normative) **Data Matrix pattern randomising 37**

Annex C (normative) **Data Matrix encodation character sets 39**

Annex D (normative) **Data Matrix alignment patterns 42**

Annex E (normative) **Data Matrix Reed-Solomon error detection and correction 44**

Annex F (normative) **Symbol character placement 48**

Annex G (normative) **Data Matrix print quality – symbology-specific aspects 64**

Annex H (normative) **Symbology identifier 75**

Annex I (informative) **Encode example 76**

Annex J (informative) **Encoding data using the minimum symbol data characters 79**

Annex K (informative) **Autodiscrimination capability 83**

Annex L (informative) **System considerations 84**

Annex M (informative) **User considerations 85**

Bibliography 86

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

ISO and IEC draw attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO and IEC take no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO and IEC had not received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at www.iso.org/patents and https://patents.iec.ch. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 31, *Automatic identification and data capture techniques*.

This third edition cancels and replaces the second edition (ISO/IEC 16022:2006), which has been technically revised.

The main changes are as follows:

— the extended channel interpretations and rectangular formats have become a mandatory feature;

— the historic data matrix variant "ECC 000" to "ECC 140" has been removed;

— continuous grading according to ISO/IEC 15415 has been introduced to all quality measurements;

— transition ratio grading has been changed;

— new quality parameter “print growth” has been added;

— the reference decode algorithm has been revised;

— the interleaving blocks for 144 x 144 matrix size have been clarified.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

Introduction

Data Matrix is a two-dimensional matrix symbology which is made up of nominally square modules arranged within a perimeter finder pattern. Though primarily shown and described in this document as a dark symbol on light background, Data Matrix symbols can also be printed to appear as light on dark.

Manufacturers of bar code equipment and users of the technology need publicly available standard symbology specifications to which they can refer when developing equipment and application standards. The publication of standardised symbology specifications is designed to achieve this.

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

Information technology — Automatic identification and data capture techniques — Data Matrix bar code symbology specification

1 Scope

This document defines the requirements for the symbology known as Data Matrix. It specifies the Data Matrix symbology characteristics, data character encodation, symbol formats, dimensions and print quality requirements, error correction rules, decoding algorithm, and user-selectable application parameters.

It applies to all Data Matrix symbols produced by any printing or marking technology.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 19762, *Information technology — Automatic identification and data capture (AIDC) techniques — Harmonized vocabulary*

ISO/IEC 15415, *Information technology — Automatic identification and data capture techniques — Bar code symbol print quality test specification — Two-dimensional symbols*

ISO/IEC 646, *Information technology — ISO 7-bit coded character set for information interchange*

ISO/IEC 8859-1, *Information technology — 8-bit single-byte coded graphic character sets — Part 1: Latin alphabet No. 1*

ISO/IEC 29158, *Information technology — Automatic identification and data capture techniques — Direct Part Mark (DPM) Quality Guideline*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 19762 and the following apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at https://www.electropedia.org/

3.1
codeword
symbol character value
intermediate level of coding between source data and the graphical encodation in the symbol

3.2
module
single cell of element in a matrix symbology symbol used to encode one bit of the *codeword* (3.1)

[SOURCE: ISO/IEC 19762:2016, 04.02.06]

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

3.3
pattern randomising
procedure to convert an original bit pattern to another bit pattern by inverting selected bits

Note 1 to entry: The resulting bitstream is less likely to have repeating patterns.

4 Symbols

e number of erasures

k total number of error correction codewords

n total number of data codewords

N numerical base in an encodation scheme

p number of codewords reserved for error detection

S symbol character

t number of errors

X horizontal and vertical width of a module

ε error correction codeword

5 Mathematical or logical notations

div integer division operator

mod integer remainder after division

XOR exclusive-or logic function whose output is one only when its two inputs are not equivalent

6 Symbol description

6.1 Basic characteristics

Data Matrix is a two-dimensional matrix symbology.

The characteristics of Data Matrix are:

a) Encodable character set:

   1) values 0 – 127 in accordance with ISO/IEC 646 IRV, i.e. all 128 ASCII characters;

   2) values 128 - 255 in accordance with ISO/IEC 8859-1 (these are referred to as extended ASCII);

   3) additional characters can be encoded using the ECI capabilities.

b) Representation of data: A dark module is a binary one and a light module is a zero.

   This document specifies Data Matrix symbols in terms of dark modules marked on a light background. However, 6.2 provides that symbols can also be produced with the module's colours reversed. In such symbols, dark modules would be a binary zero, and light modules would be a binary one.

c) Symbol size in modules (not including quiet zone) ranging from 10 x 10 to 144 x 144 square and rectangular versions ranging from 8 x 18 to 16 x 48 (see Table 10).

   NOTE Additional rectangular symbol sizes are defined in ISO/IEC 21471 (see Reference [4]).

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

d) Data characters per symbol (for maximum symbol size):

   1) Alphanumeric data: up to 2 335 characters

   2) 8-bit byte data: 1 555 characters

   3) Numeric data: 3 116 digits.

e) Code type: Matrix

f) Orientation independence: Yes

g) Error detection and correction: Reed Solomon.

6.2 Summary of additional features

The following summarises additional features which are inherent or optional in Data Matrix:

a) Reflectance reversal: (inherent): Symbols are either dark on light or light on dark (see Figure 1). The specifications in this document are based on dark images on a light background, therefore references to dark or light modules should be taken as references to light or dark modules respectively in the case of symbols produced with reflectance reversal.

b) Extended Channel Interpretations: (ECI), (inherent): This mechanism enables characters from other character sets (e.g. Arabic, Cyrillic, Greek, Hebrew) and other data interpretations or industry-specific requirements to be represented.

c) Rectangular symbols: (inherent): Six symbol formats are specified in a rectangular form.

   NOTE Additional rectangular symbol formats are available by ISO/IEC 21471 (see Reference [4]).

d) Structured append: (optional): This allows files of data to be represented in up to 16 Data Matrix symbols. The original data can be correctly reconstructed regardless of the order in which the symbols are scanned. If the feature is not implemented, reader should not transmit data in case of a structured append symbol.

6.3 Symbol structure

6.3.1 General

Each Data Matrix symbol consists of data regions which contain nominally square modules set out in a regular array. In larger symbols, data regions are separated by alignment patterns. The data region, or set of data regions and alignment patterns, is surrounded by a finder pattern, and this shall in turn be surrounded on all four sides by a quiet zone border. Figure 1 illustrates two representations of a Data Matrix symbol, dark on light and reflectance reversal.

a) Data Matrix, dark on light

b) Data Matrix, light on dark (reflectance reversal)

Figure 1 — Data Matrix "A1B2C3D4E5F6G7H8I9J0K1L2"

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

6.3.2 Finder pattern

The finder pattern is a perimeter to the data region and is one module wide. Two adjacent sides, the left and lower sides, forming the L boundary, are solid lines; these are used primarily to determine physical size, orientation and symbol distortion. The two opposite sides are made up of alternating dark and light modules. These are used primarily to define the cell structure of the symbol, but also can assist in determining physical size and distortion. The extent of the quiet zone is indicated by the corner marks in Figure 1.

6.3.3 Symbol sizes and capacities

Data Matrix symbols have an even number of rows and an even number of columns. Some symbols are square with sizes from 10 x 10 to 144 x 144 not including quiet zones. Some symbols are rectangular with sizes from 8 x 18 to 16 x 48 not including quiet zones. For all Data Matrix code symbols, the upper right corner module has the opposite reflectance state (i.e. light or dark) of the "L" finder pattern (see Figure 1). The complete attributes are given in Table 10.

7 Data Matrix code requirements

7.1 Encode procedure overview

7.1.1 General

Subclause 7.1 provides an overview of the encoding procedure. Following sections will provide more details. An encoding example is given in Annex I. The following steps convert user data to a Data Matrix code symbol.

7.1.2 Step 1: data encodation

As Data Matrix includes various encodation schemes that allow a defined set of characters to be converted into codewords more efficiently than the default scheme, analyse the data stream to identify the variety of different characters to be encoded. Insert additional codewords to switch between the encodation schemes and to perform other functions. Add pad characters as needed to fill the required number of codewords. If the user does not specify the matrix size, then choose the smallest size that accommodates the data. A complete list of matrix sizes is shown in Table 10.

Table 1 — Encodation schemes for Data Matrix code

Encodation scheme	Characters	Bits per data character
ASCII	double digit numerics	4
	ASCII values 0 to 127	8
	Extended ASCII values 128 to 255	16
C40	Upper case alphanumeric	5,33
	Lower case and special characters	10,66[a]
Text	Lower case alphanumeric	5,33
	Upper case and special characters	10,66[b]
X12	ANSI X12 EDI data set	5,33
EDIFACT	ASCII values 32 to 94	6
Base 256	All byte values 0 to 255	8

[a] Encoded as two C40 values as result of use of a shift character.

[b] Encoded as two Text values as result of use of a shift character.

7.1.3 Step 2: error checking and correcting codeword generation

For symbols with more than 255 codewords, sub-divide the codeword stream into interleaved blocks to enable the error correction algorithms to be processed as shown in Annex A. Generate the error correction

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

codewords for each block. The result of this process expands the codeword stream by the number of error correction codewords. Place the error correction codewords after the data codewords.

7.1.4 Step 3: module placement in matrix

Place the codeword modules in the matrix. Insert the alignment pattern modules, if any, in the matrix. Add the finder pattern modules around the matrix.

7.2 Data encodation

7.2.1 Overview

The data may be encoded using any combination of six encodation schemes (see Table 1). ASCII encodation is the basic scheme. All other encodation schemes are invoked from ASCII encodation and return to this scheme. The compaction efficiencies given in Table 1 need to be interpreted carefully. The best scheme for a given set of data may not be the one with the fewest bits per data character. If the highest degree of compaction is required, account has to be taken of switching between encodation schemes and between code sets within an encodation scheme (see Annex J). It should also be noted that even if the number of codewords is minimised, the codeword stream sometimes needs to be expanded to fill a symbol. This fill process is done using pad characters.

7.2.2 Default character interpretation

The default character interpretation for character values 0 to 127 shall conform to ISO/IEC 646 IRV. The default character interpretation for character values 128 to 255 shall conform to ISO/IEC 8859-1. The graphical representation of data characters shown throughout this document complies with the default interpretation. This interpretation can be changed using ECI escape sequences, see 7.3. The default interpretation corresponds to ECI 000003.

7.2.3 ASCII encodation

ASCII encodation is the default set for the first symbol character in all symbol sizes. It encodes ASCII data, double density numeric data and symbology control characters. Symbology control characters include function characters, the pad character and the switches to other code sets. ASCII data is encoded as codewords 1 to 128 (ASCII value plus 1). Extended ASCII (data values 128 to 255) is encoded using the upper shift symbology control character (see 7.2.4.3). The digit pairs 00 to 99 are encoded with codewords 130 to 229 (numeric value plus 130). The ASCII code assignments are shown in Table 2.

NOTE ASCII encodation is the name of the character set in Data Matrix. It is not to be confused with the ASCII character set (ISO/IEC 646 IRV).

Table 2 — ASCII encodation values

Codeword	Data or function
0	Not to be used in ASCII encodation
1 to 128	ASCII data (ASCII value + 1)
129	Pad
130 to 229	2-digit data 00 to 99 (Numeric Value + 130)
230	Latch to C40 encodation
231	Latch to Base 256 encodation
232	FNC1
233	Structured Append
234	Reader Programming
235	Upper Shift (shift to Extended ASCII)
236	05 Macro

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

Table 2 *(continued)*

Codeword	Data or function
237	06 Macro
238	Latch to ANSI X12 encodation
239	Latch to Text encodation
240	Latch to EDIFACT encodation
241	ECI Character
242 to 255	Not to be used in ASCII encodation

7.2.4 Symbology control characters

7.2.4.1 General

Data Matrix symbols have several special symbology control characters, which have particular significance to the encodation scheme. These characters shall be used to instruct the decoder to perform certain functions or to send specific data to the host computer as described in 7.2.4.2 to 7.2.4.10. These symbology control characters, with the exception of values from 242 through 255, are found in the ASCII encodation (see Table 2).

7.2.4.2 Latch characters

A latch character shall be used to switch from ASCII encodation to one of the other encodation schemes. All codewords which follow a latch character shall be compacted according to the new encodation scheme. The encodation schemes have different methods for returning to the ASCII encodation.

7.2.4.3 Upper Shift character

The Upper Shift character is used in combination with an ASCII value (1 to 128) to encode an extended ASCII character (129 to 255). An extended ASCII character encoded in the ASCII, C40, or Text encodation scheme requires a preceding Upper Shift character and the extended ASCII character value decreased by 128 is then encoded according to the rules of the encodation scheme. In ASCII encodation, the Upper Shift character is represented by codeword 235. The reduced data value (i.e. ASCII value minus 128) is transformed into its codeword value by adding 1. For example, to encode ¥ (Yen currency symbol) (ASCII value 165), an upper shift character (codeword 235) is followed by value 37 (165 to 128), which is encoded as codeword 38. If there are long data strings of characters from the extended ASCII range, a latch to Base 256 encodation should be more efficient.

7.2.4.4 Pad character

If the encoded data, irrespective of the encodation scheme in force, does not fill the data capacity of the symbol, pad characters (value 129 in ASCII encodation) shall be added to fill the remaining data capacity of the symbol. The pad characters shall only be used for this purpose. Before inserting pad characters, it is necessary to return to ASCII encodation if in any other encodation mode. The 253-State pattern randomising algorithm shall be applied to the pad characters starting at the second pad character as specified in Annex B.

7.2.4.5 ECI character

An ECI character is used to change from the default interpretation used to encode data. The ECI protocol is common across a number of symbologies and its application to Data Matrix is defined more fully in 7.3. The ECI character shall be followed by one, two or three codewords which identify the ECI being invoked. The new ECI remains in place until the end of the encoded data, or until another ECI character is used to invoke another interpretation. See also Annex M.

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

7.2.4.6 Shift characters in C40 and Text encodation

In C40 and Text encodation, three special characters, called shift characters, are used as a prefix to one of 40 values to encode about three quarters of the ASCII characters. This allows the remaining ASCII characters to be encoded in a more condensed way with single values.

7.2.4.7 FNC1 alternate data type identifier

To encode data to conform to specific industry standards as authorised by AIM Inc., a FNC1 character shall appear in the first or second symbol character position (or in the fifth or sixth data positions of the first symbol of Structured Append). FNC1 encoded in any other position is used as a field separator and shall be transmitted as $^{G}{}_{S}$ control character (ASCII value 29).

7.2.4.8 Macro characters

Data Matrix provides a means of abbreviating an industry specific header and trailer in one symbol character. This feature exists to reduce the number of symbol characters needed to encode data in a symbol using certain structured formats. A Macro character shall be in the first character position of a symbol. They shall not be used in conjunction with Structured Append and their functions are defined in Table 3. The header shall be transmitted as a prefix to the data stream and the trailer shall be transmitted as a suffix to the data stream. The symbology identifier, if used, shall precede the header.

Table 3 — Macro functions

Macro codeword	Name	Interpretation	
		Header	Trailer
236	05 Macro	$[)>{}^{R}{}_{S}05{}^{G}{}_{S}$	${}^{R}{}_{S}\ {}^{E}o_{T}$
237	06 Macro	$[)>{}^{R}{}_{S}06{}^{G}{}_{S}$	${}^{R}{}_{S}\ {}^{E}o_{T}$

7.2.4.9 Structured Append character

A Structured Append character is used to indicate that the symbol is part of a Structured Append sequence according to the rules defined in 7.5.

7.2.4.10 Reader Programming character

A Reader Programming character indicates that the symbol encodes a message used to program the reader system. The Reader Programming character shall appear as the first codeword of the symbol and Reader Programming shall not be used with Structured Append.

7.2.5 C40 encodation

7.2.5.1 General

The C40 encodation scheme is designed to optimise the encoding of upper-case alphabetic and numeric characters but also enables other characters to be encoded by the use of shift characters in conjunction with the data character.

C40 characters are partitioned into 4 subsets. Characters of the first set, called the basic set, are the three special shift characters, the space character, and the ASCII characters A to Z and 0 to 9. They are assigned to a single C40 values. Characters of the other sets shall be assigned to one of the three shift characters, pointing to one of the 3 remaining subset, followed by one of the C40 values (see Table C.1).

As a first stage, each data character is converted into a single C40 value or a pair of C40 values. The complete string of C40 values is then decomposed into groups of three values (special rules apply if one or two values remain at the end, see 7.2.5.3). Each triplet (*C1*, *C2*, *C3*) is then encoded into a 16-bit value according to the formula: (1600 * *C1*) + (40 * *C2*) + *C3* + 1. Each 16-bit value is then separated into 2 codewords by taking the most significant 8 bits and the least significant 8 bits.

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

7.2.5.2 Switching to and from C40 encodation

It is possible to switch to C40 encodation from ASCII encodation using the appropriate latch codeword (230). Codeword 254 immediately following a pair of codewords in C40 encodation acts as an unlatch codeword to switch back to ASCII encodation. Otherwise, the C40 encodation remains in effect to the end of the data encoded in the symbol.

7.2.5.3 C40 encodation rules

Each pair of codewords represents a 16-bit value where the first codeword represents the most significant 8 bits. Three C40 values (*C1*, *C2*, *C3*) shall be encoded as:

$$(1\,600 * C1) + (40 * C2) + C3 + 1$$

which produces a value from 1 to 64 000. Table 4 illustrates three C40 values compacted into two codewords. Characters in the Shift 1, Shift 2 and Shift 3 sets shall be encoded by first encoding the appropriate shift character, and then the C40 value for the data. C40 encodation may be in effect at the end of the symbol's codewords which encode data.

The following rules apply when only one or two symbol characters remain in the symbol before the start of the error correction codewords:

a) If two symbol characters remain and three C40 values remain to be encoded (which may include both data and shift characters) encode the three C40 values in the last two symbol characters. A final unlatch codeword is not required.

b) If two symbol characters remain and two C40 values remain to be encoded (the first C40 value may be a shift or data character but the second shall represent a data character) append a “Shift 1” character (C40 value 0) to the input to increase to three characters so that it can be processed normally as with any three character C40 input. A final unlatch codeword again is not required.

c) If two symbol characters remain and only one C40 value (data character) remains to be encoded, the first symbol character is encoded as an unlatch character and the last symbol character is encoded with the data character using the ASCII encodation scheme.

d) If one symbol character remains and one C40 value (data character) remains to be encoded, the last symbol character is encoded with the data character using the ASCII encodation scheme. The unlatch character is not encoded, but is assumed, before the last symbol character.

In all other cases, either an unlatch character is used to exit the C40 encodation scheme before the end of the symbol, or a larger symbol size is required to encode the data.

Table 4 — Example of C40 encoding

Data characters	AIM
C40 values	14, 22, 26
Calculate 16-bit value	(1600 * 14) + (40 * 22) + 26 + 1 = 23307
1st codeword: (16-bit value) div 256	23 307 div 256 = 91
2nd codeword: (16-bit value) mod 256	23 307 mod 256 = 11
Codewords	91, 11

7.2.5.4 Use of Upper Shift with C40

In C40 encodation the Upper Shift character is not a symbology function character but a shift within the encodation set. When a data character from the extended ASCII character range is encountered, three or four values in C40 encodation need to be encoded according to the following rule:

IF [ASCII value – 128] is in the Basic Set then:

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

[1(Shift 2)] [30(Upper Shift)] [V(ASCII value – 128)]

ELSE

[1(Shift 2)] [30(Upper Shift)] [0, 1, or 2(Shift 1, 2, or 3)] [V(ASCII value – 128)]

In the rule the number in [ ] equates to the C40 values from Table C.1; V has been used to indicate the appropriate C40 value.

7.2.6 Text encodation

7.2.6.1 General

Text encodation is designed to encode normal printed text, which is predominantly lowercase characters. It is similar in structure to the C40 encodation set, except that lowercase alphabetic characters are directly encoded (i.e. without using a shift). Upper-case alphabetic characters are preceded by a Shift 3. The full Text encodation character set assignments shall be as shown in Table C.2.

7.2.6.2 Switching to and from Text encodation

It is possible to switch to Text encodation from ASCII encodation using the appropriate latch codeword (239). Codeword 254 immediately following a pair of codewords in text encodation acts as an unlatch codeword to switch back to ASCII encodation. Otherwise, the Text encodation remains in effect to the end of the data encoded in the symbol.

7.2.6.3 Text encodation rules

The rules for C40 encodation apply.

7.2.7 ANSI X12 encodation

7.2.7.1 General

ANSI X12 encodation is used to encode the standard ANSI X12 electronic data interchange characters, which are compacted three data characters to two codewords in a manner similar to C40 encodation. It encodes upper-case alphabetic characters, numerics, space and the three standard ANSI X12 terminator and separator characters. The ANSI X12 code assignments are shown in Table 5. There are no shift characters in the ANSI X12 encodation set.

Table 5 — ANSI X12 encodation set

X12 value	Encoded characters	ASCII values
0	X12 segment terminator <CR>	13
1	X12 segment separator *	42
2	X12 sub-element separator >	62
3	space	32
4 to 13	0 to 9	48 to 57
14 to 39	A to Z	65 to 90

7.2.7.2 Switching to and from ANSI X12 encodation

It is possible to switch to ANSI X12 encodation from ASCII encodation using the appropriate latch codeword (238). Codeword 254 immediately following a pair of codewords in ANSI X12 encodation acts as an unlatch codeword to switch back to ASCII encodation. Otherwise, the ANSI X12 encodation remains in effect to the end of the data encoded in the symbol.

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

7.2.7.3 ANSI X12 encodation rules

The rules of C40 encodation apply. The exception is at the end of encoding ANSI X12 data. If the data characters do not fully utilise pairs of codewords, then following the last complete pair of codewords switch to ASCII using codeword 254 and continue using ASCII encodation, except when a single symbol character is left at the end before the first error correction character. This single symbol character uses the ASCII encodation scheme without requiring an unlatch codeword.

7.2.8 EDIFACT encodation

7.2.8.1 General

The EDIFACT encodation scheme includes 63 ASCII values (values from 32 to 94) plus an unlatch character (binary 011111) to return to ASCII encodation. EDIFACT encodation encodes four data characters in three codewords. It includes all the numeric, alphabetic and punctuation characters defined in the EDIFACT Level A character set without any of the shifts required in C40 encodation.

7.2.8.2 Switching to and from EDIFACT encodation

It is possible to switch to EDIFACT encodation from ASCII encodation using the appropriate latch codeword (240). The unlatch character in EDIFACT encodation shall be used as a terminator at the end of EDIFACT encodation, which reverts to ASCII encodation.

7.2.8.3 EDIFACT encodation Rules

The EDIFACT encodation character set is defined in Table C.3. There is a simple relationship between the 6-bit EDIFACT value and the ASCII 8-bit byte. The leading two bits of the 8-bit byte are ignored to create the EDIFACT 6-bit value, as illustrated in Table 6. Strings of four EDIFACT characters are encoded in three codewords. For a simple encodation process, the leading two bits of the 8-bit byte are removed. The remaining 6-bit byte is the EDIFACT value and shall be directly encoded into the codeword as illustrated in Table 7. When EDIFACT encodation is terminated with the unlatch character, any remaining bits left in the single symbol character shall be filled with zeros. ASCII encodation starts with the next symbol character. If EDIFACT encodation is in effect at the end of the symbol before the first error correction character, and only one or two codewords remain after the last EDIFACT codeword triplet, these remaining codewords shall be encoded in ASCII encodation without requiring an unlatch character.

Table 6 — Relationship between the EDIFACT value and the 8-bit byte value

Data character	ASCII		EDIFACT value
	Decimal value	8-bit binary value	
A	65	01000001	000001
9	57	00111001	111001

NOTE During the decode process, if the leading (6th) bit is 1, the bits 00 are prefixed to create the 8-bit byte. If the leading (6th) bit is 0, the bits 01 are prefixed to create the 8-bit byte. The exception to this is the EDIFACT value 011111 which is the symbology control unlatch character to return to ASCII encodation.

Table 7 — Example of EDIFACT encodation

Data characters	D			A			T			A		
Binary values (Table C.3)	00	01	00	00	00	01	01	01	00	00	00	01
Divide into 3 8-bit bytes	00	01	00	00	00	01	01	01	00	00	00	01
Codeword values	16			21			1					

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

7.2.9 Base 256 encodation

7.2.9.1 General

The Base 256 encodation scheme shall be used to encode any 8-bit byte data, including extended channel interpretations and binary data. The default interpretation is defined in 7.2.2. The 255-State pattern randomising algorithm shall be applied to each Base 256 sequence within the encoded data (see B.2). It starts after the latch to Base 256 encodation and ends at the last character specified by the Base 256 field length.

7.2.9.2 Switching to and from Base 256 encodation

It is possible to switch to Base 256 encodation from ASCII encodation using the appropriate latch codeword (231). At the end of Base 256 encodation, encodation automatically reverts to ASCII encodation. The appropriate ECI, if other than the default, shall be invoked prior to switching. The ECI sequence need not occur immediately before switching to Base 256 encodation.

7.2.9.3 Base 256 encodation rules

After switching to Base 256 encodation, the first one (*d1*) or two (*d1*, *d2*) codewords define the data field length in bytes. Table 8 specifies how the field length is defined. Thereafter, all encodation shall be of the byte values.

Table 8 — Base 256 field length

Field length	Values of *d1*, *d2*	Permitted values of *d*
Remainder of symbol	*d1* = 0	*d1* = 0
1 to 249	*d1* = length	*d1* = 1 to 249
250 to 1555	*d1* = (length DIV 250) + 249	*d1* = 250 to 255
	d2 = length MOD 250	*d2* = 0 to 249

7.3 ECI

7.3.1 General

The ECI protocol allows the output data stream to have interpretations different from that of the default character set. The ECI protocol is defined consistently across a number of symbologies. Four broad types of interpretations are supported in Data Matrix:

a) international character sets (or code pages);

b) general purpose interpretations such as encryption and compaction;

c) user defined interpretations for closed systems;

d) control information for structured append in unbuffered mode.

The ECI protocol is fully specified in Reference [1]. The protocol provides a consistent method to specify particular interpretations on byte values before printing and after decoding. The ECI is identified by a 6-digit number which is encoded in the Data Matrix symbol by the ECI character followed by one to three codewords. Specific interpretations are listed in AIM Inc. Extended Channel Interpretations Character Set Register. The ECI can only be used with readers enabled to transmit the symbology identifiers. Readers that are not enabled to transmit the symbology identifier shall not transmit the data from any symbol containing an ECI. An exception can be made if the ECI(s) can be handled entirely within the reader.

A specified Extended Channel Interpretation may be invoked anywhere in the encoded message.

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

7.3.2 Encoding ECIs

The various encodation schemes of Data Matrix (defined in Table 1) may be applied under any of the ECIs. The ECI can only be invoked from ASCII encodation; once this has occurred, switching may take place between any of the encodation schemes. The encodation mode used is determined strictly by the 8-bit data values being encoded and does not depend on the Extended Channel Interpretation in force. For example, a sequence of values in the range 48 to 57 (decimal) would be most efficiently encoded in numeric mode even if they were not to be interpreted as numbers. The ECI assignment is invoked using codeword 241 (ECI character) in ASCII encodation. One, two, or three additional codewords are used to encode the ECI Assignment number. The encodation rules are defined in Table 9.

The following examples illustrate the encodation:

ECI = 015000

Codewords:

[241] [(15000 - 127) div 254 + 128] [(15000 - 127) mod 254 + 1]

= [241] [58 + 128] [141 + 1]

= [241] [186] [142]

ECI = 090000

Codewords:

[241] [(90000 - 16383) div 64516 + 192] [((90000 - 16383) div 254) mod 254 + 1] [(90000 - 16383) mod 254 + 1]

= [241] [1 + 192] [289 mod 254 + 1] [211 + 1]

= [241] [193] [36] [212]

Table 9 — Encoding ECI assignment numbers in Data Matrix code

ECI assignment value	Codeword sequence	Codeword values	Ranges
000000 to 000126	C_0	241	
	C_1	ECI_no + 1	C_1 = (1 to 127)
000127 to 016382	C_0	241	
	C_1	(ECI_no - 127) div 254 + 128	C_1 = (128 to 191)
	C_2	(ECI_no - 127) mod 254 + 1	C_2 = (1 to 254)
0016383 to 999999	C_0	241	
	C_1	(ECI_no - 16383) div 64516 +192	C_1 = (192 to 207)
	C_2	[(ECI_no - 16383) div 254] mod 254 + 1	C_2 = (1 to 254)
	C_3	(ECI_no - 16383) mod 254 + 1	C_3 = (1 to 254)

7.3.3 ECIs and Structured Append

ECIs may occur anywhere in the message encoded in a single or Structured Append (see 7.5) set of Data Matrix symbols. Any ECI invoked shall apply until the end of the encoded data, or until another ECI is encountered. Thus, the interpretation of the ECI may straddle two or more symbols.

7.3.4 Post-decode protocol

The protocol for transmitting ECI data shall be as defined in 12.5. When using ECIs, symbology identifiers (see 12.6) shall be fully implemented and the appropriate symbology identifier transmitted as a preamble.

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

7.4 Data Matrix symbol attributes

7.4.1 Symbol sizes and capacity

There are 24 square symbols and 6 rectangular symbols available in Data Matrix code. These are as specified in Table 10.

NOTE Additional rectangular symbol formats are specified in ISO/IEC 21471.

Table 10 — Data Matrix symbol attributes

Symbol size[a]		Data region		Mapping matrix size	Total codewords		Reed-Solomon block		Inter-leaved blocks	Maximum data capacity			% of codewords used for error correction	Max. correctable codewords Error/ erasure[b]
Row	Col	Size	No.		Data	Error	Data	Error		Num.	Alphanum.[d]	Byte		
10	10	8 x 8	1	8 x 8	3	5	3	5	1	6	3	1	62,5	2/0
12	12	10 x 10	1	10 x 10	5	7	5	7	1	10	6	3	58,3	3/0
14	14	12 x 12	1	12 x 12	8	10	8	10	1	16	10	6	55,6	5/7
16	16	14 x 14	1	14 x 14	12	12	12	12	1	24	16	10	50	6/9
18	18	16 x 16	1	16 x 16	18	14	18	14	1	36	25	16	43,8	7/11
20	20	18 x 18	1	18 x 18	22	18	22	18	1	44	31	20	45	9/15
22	22	20 x 20	1	20 x 20	30	20	30	20	1	60	43	28	40	10/17
24	24	22 x 22	1	22 x 22	36	24	36	24	1	72	52	34	40	12/21
26	26	24 x 24	1	24 x 24	44	28	44	28	1	88	64	42	38,9	14/25
32	32	14 x 14	4	28 x 28	62	36	62	36	1	124	91	60	36,7	18/33
36	36	16 x 16	4	32 x 32	86	42	86	42	1	172	127	84	32,8	21/39
40	40	18 x 18	4	36 x 36	114	48	114	48	1	228	169	112	29,6	24/45
44	44	20 x 20	4	40 x 40	144	56	144	56	1	288	214	142	28	28/53
48	48	22 x 22	4	44 x 44	174	68	174	68	1	348	259	172	28,1	34/65
52	52	24 x 24	4	48 x 48	204	84	102	42	2	408	304	202	29,2	42/78
64	64	14 x 14	16	56 x 56	280	112	140	56	2	560	418	277	28,6	56/106
72	72	16 x 16	16	64 x 64	368	144	92	36	4	736	550	365	28,1	72/132
80	80	18 x 18	16	72 x 72	456	192	114	48	4	912	682	453	29,6	96/180
88	88	20 x 20	16	80 x 80	576	224	144	56	4	1 152	862	573	28	112/212
96	96	22 x 22	16	88 x 88	696	272	174	68	4	1 392	1 042	693	28,1	136/260
104	104	24 x 24	16	96 x 96	816	336	136	56	6	1 632	1 222	813	29,2	168/318
120	120	18 x 18	36	108 x 108	1 050	408	175	68	6	2 100	1 573	1 047	28	204/390
132	132	20 x 20	36	120 x 120	1 304	496	163	62	8	2 608	1 954	1 301	27,6	248/472
144	144	22 x 22	36	132 x 132	1 558	620	156	62	8[c]	3 116	2 335	1 555	28,5	310/590
							155	62	2[c]					
Rectangular Symbols														
8	18	6 x 16	1	6 x 16	5	7	5	7	1	10	6	3	58,3	3/0
8	32	6 x 14	2	6 x 28	10	11	10	11	1	20	13	8	52,4	5/0
12	26	10 x 24	1	10 x 24	16	14	16	14	1	32	22	14	46,7	7/11
12	36	10 x 16	2	10 x 32	22	18	22	18	1	44	31	20	45,0	9/15
16	36	14 x 16	2	14 x 32	32	24	32	24	1	64	46	30	42,9	12/21
16	48	14 x 22	2	14 x 44	49	28	49	28	1	98	72	47	36,4	14/25

[a] Symbol size does not include quiet zones.

[b] See 7.6.3.

[c] In the largest symbol (144 x 144), the first eight Reed-Solomon blocks are 218 codewords long encoding 156 data codewords, and the last two blocks encode 217 codewords (155 data codewords). All the blocks have 62 error correction codewords.

[d] Based on text or C40 encoding without switching or shifting; for other encoding schemes, this value may vary depending on the mix and grouping of character sets.

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

7.4.2 Insertion of Alignment Patterns into larger symbols

As shown in Table 10, square symbols 32 x 32 and larger and four rectangular symbols (8 x 32, 12 x 36, 16 x 36, and 16 x 48) have two or more data regions. These data regions shall be bound by alignment patterns in accordance with Annex D. The square symbols are divided into 4, 16 or 36 data regions (as illustrated in Figures D.1, D.2 and D.3). The rectangular symbols are divided into two data regions (as illustrated in Figure D.4). The alternating dark modules of the alignment pattern shall be to the top and right of a data region and identify the even columns and rows.

7.5 Structured Append

7.5.1 Basic principles

Up to 16 Data Matrix symbols may be appended in a structured format. If a symbol is part of a Structured Append, this is indicated by codeword 233 in the first symbol character position. This is immediately followed by three structured append codewords. The first codeword is the symbol sequence indicator. The second and third codewords are the file identification.

7.5.2 Symbol sequence indicator

This codeword indicates the position of the symbol within the set (up to 16) of Data Matrix symbols in the Structured Append format in the form *m* of *n* symbols. The first 4 bits of this codeword identify the position of the particular symbol as the binary value of (*m* – 1). The last 4 bits identify the total number of the symbols to be concatenated in the Structured Append format as the binary value of (17 – *n*). The 4-bit patterns shall conform with those defined in Table 11.

Table 11 — Structured Append symbol position bits

Symbol position	Bits 1234	Total number of symbols	Bits 5678
1	0000		
2	0001	2	1111
3	0010	3	1110
4	0011	4	1101
5	0100	5	1100
6	0101	6	1011
7	0110	7	1010
8	0111	8	1001
9	1000	9	1000
10	1001	10	0111
11	1010	11	0110
12	1011	12	0101
13	1100	13	0100
14	1101	14	0011
15	1110	15	0010
16	1111	16	0001

For example, the 3rd symbol of a set of 7 shall be encoded thus:

3rd position: 0010

Total 7 symbols: 1010

Bit pattern: 00101010

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

Codeword value: 42

7.5.3 File identification

The file identification is defined by the value of its two codewords. Each file identification codeword may have a value 1 to 254, allowing 64 516 different file identifications. The purpose of the file identification is to increase the probability that only logically linked symbols are processed as part of the same message.

7.5.4 FNC1 and Structured Append

If Structured Append is used in conjunction with FNC1 (see 7.2.4.7), the first four codewords shall be used for Structured Append and the fifth and sixth codewords are available for FNC1 usage. FNC1 shall not be repeated in these positions in the second and subsequent symbols, except when used as a field separator.

7.5.5 Buffered and unbuffered operation

The message within a Structured Append sequence can be buffered in the reader in its entirety and transmitted after all of the symbols have been read. Alternatively, the reader may transmit the decoded data in each symbol as it is read. In this unbuffered operation, the ECI protocol for structured append (specified in AIM ITS/04-001, Part 1) defines a control block that shall be prefixed to the beginning of the data transmitted for each symbol.

7.6 Error detection and correction

7.6.1 Reed-Solomon error correction

Data Matrix symbols employ Reed-Solomon error correction. For Data Matrix symbols with less than 255 total codewords, the error correction codewords are calculated from data codewords with no interleaving. For Data Matrix symbols with more than 255 total codewords, the error correction codewords shall be calculated from data codewords with the interleaving procedure described in Annex A. Each Data Matrix symbol has a specific number of data and error correction codewords which are divided into a specific number of blocks, as defined in Table 10, and to which the interleaving procedure defined in Annex A is applied.

The polynomial arithmetic for Data Matrix shall be calculated using bit-wise modulo 2 arithmetic and byte-wise modulo 100101101 (decimal 301) arithmetic. This is a Galois field of 2^8 with 100101101 representing the field's prime modulus polynomial: $x^8 + x^5 + x^3 + x^2 + 1$. Sixteen different generator polynomials are used for generating the appropriate error correction codewords. These are given in Clause E.1.

NOTE An example implementation of the error correction algorithm is available in Reference [2].

7.6.2 Generating the error correction codewords

The error correction codewords are the remainder after dividing the data codewords by a polynomial $g(x)$ used for Reed-Solomon codes (see Clause E.1).

If this calculation is performed by "long division", the symbol data polynomial shall first be multiplied by x^k.

The data codewords are the coefficients of the terms of a polynomial with the coefficient of the highest term being the first data codeword and the lowest power term being the last data codeword before the first error correction codeword. The highest order coefficient of the remainder is the first error correction codeword and the zero power coefficient is the last error correction codeword and the last codeword. This can be implemented by using the division circuit as shown in Figure 2. The registers b_0 through b_{k-1} are initialised as zeros. There are two phases to generate the encoding. In the first phase, with the switch in the down position the data codewords are passed both to the output and the circuit. The first phase is complete after n clock pulses. In the second phase ($n + 1 \ldots n + k$ clock pulses), with the switch in the up position, the error

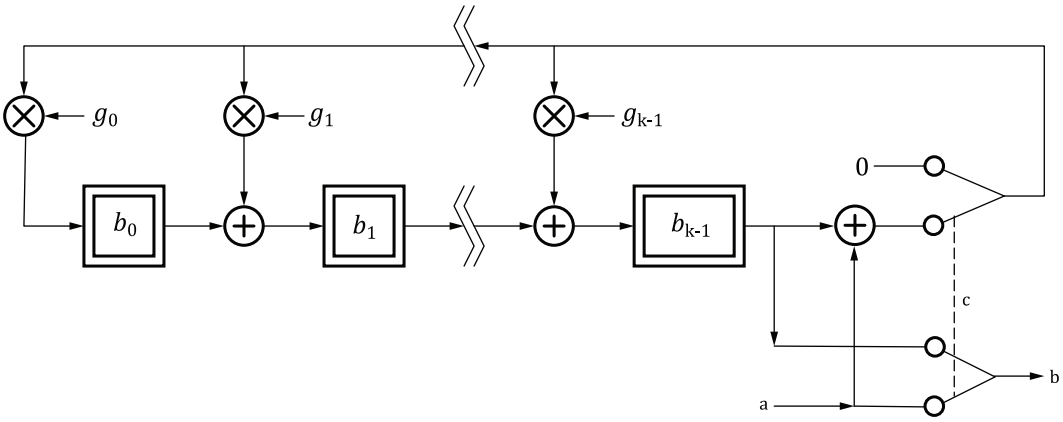
© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

correction codewords $\varepsilon_{k-1}, \ldots, \varepsilon_0$ are generated by flushing the registers in order while keeping the data input at 0. The codewords output from the shift register are in the order that they are to be placed in the symbol.

NOTE 1 If interleaving is used, the codewords will not be placed in consecutive symbol characters. (See Annex A).

NOTE 2 n and k are defined in Clause 4 as the number of data codewords and the number of error correction codewords, respectively.



Key

⊕ GF(256) addition

⊗ GF(256) multiplication

a Input.

b Output.

c Switch.

Figure 2 — Error correction codeword encoding circuit

7.6.3 Error correction capacity

The error correction codewords can correct two types of erroneous codewords: erasures (erroneous codewords at known locations) and errors (erroneous codewords at unknown locations). An erasure is an unscanned or undecodable symbol character. An error is a misdecoded symbol character. The number of erasures and errors that can be corrected is given by Formula (1):

$$e + 2t \leq d - p \tag{1}$$

where

e is the number of erasures;

t is the number of errors;

d is the number of error correction codewords;

p is the number of codewords reserved for error detection.

In the general case, $p = 0$. However, if most of the error correction capacity is used to correct erasures, then the possibility of an undetected error is increased. Whenever the number of erasures is more than half the number of error correction codewords, $p = 3$. For small symbols (10 x 10, 12 x 12, 8 x 18, and 8 x 32), erasure correction should not be used ($e = 0$ and $p = 1$).

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

7.7 Symbol construction

7.7.1 General

Given the codeword sequence obtained in the previous sections, a Data Matrix symbol is constructed using the following steps.

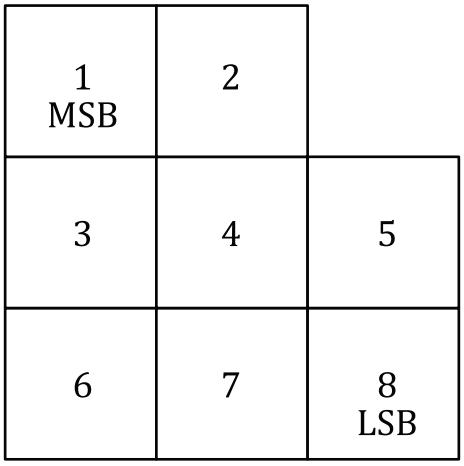
a) Place codeword modules in a mapping matrix.

b) Insert alignment pattern modules, if any.

c) Place finder modules along the perimeter.

7.7.2 Symbol character placement

Each symbol character shall be represented by eight modules which are nominally square in shape; each module represents a binary bit. A dark module is a one and a light module is a zero. The eight modules are in order from left to right and top to bottom to form a symbol character as shown in Figure 3. Because the symbol character shape defined in Figure 3 cannot be perfectly nested at the symbol boundary, some symbol characters are split into portions. Symbol character placement is defined in the C language program in Clause F.1, described in Clause F.2 and illustrated in Clause F.3 in Annex F. Symbol character placement as defined in Annex F shall be used.



Key

LSB least significant bit

MSB most significant bit

Figure 3 — Representation of a codeword in a symbol character

7.7.3 Alignment Pattern module placement

This step is only needed for larger matrices:

— square: 32 x 32, and

— larger rectangular: 8 x 32, 12 x 36 and larger.

The mapping matrix is sub-divided into data regions, of the sizes defined in Table 10, for the chosen symbol format. The data regions are separated from each other by two-module-wide alignment patterns. This will result in some of the symbol characters being split between two adjacent data regions. For square matrices, the alignment patterns are placed between the data regions horizontally and vertically in pairs with a total alignment pattern count of 2, 6 or 10 as shown in Figures D.1 to D.3. For rectangular matrices, only a single vertical alignment pattern is placed between the data regions as shown in Figure D.4.

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

7.7.4 Finder Pattern module placement

Modules are placed along the perimeter of the matrix to construct the finder pattern as described in 6.3.2.

8 Symbol dimensions

Data Matrix symbols shall conform to the following dimensions.

— *X* dimension: the width of a module shall be specified by the application, taking into account the scanning technology to be used, and the technology to produce the symbol.

— Finder pattern: the width of the finder pattern shall be equal to *X.*

— Alignment pattern: the width of the alignment pattern shall be equal to 2*X.*

— Quiet zone: the minimum quiet zone is equal to *X* on all four sides. For applications with moderate to excessive reflectance noise in close proximity to the symbol, a quiet zone of 2*X* to 4*X* is recommended.

9 Symbol quality

9.1 General

Data Matrix symbols shall be assessed for quality using the 2D matrix bar code symbol print quality specification defined in ISO/IEC 15415 or ISO/IEC 29158 in accordance with an applicable application specification, as augmented and modified below.

Some marking technologies can be unable to produce symbols conforming to this document without taking special precautions.

9.2 Symbol quality parameters

9.2.1 Fixed pattern damage

Annex G defines the measurement and grading basis for Fixed Pattern Damage.

9.2.2 Overall symbol grade

The grading method of ISO/IEC 15415 or ISO/IEC 29158 shall be used in accordance with an application specification.

9.2.3 Decode

The reference decode algorithm specified in this document shall be applied to determine the grade for Decode. A failure of the reference decode algorithm to successfully decode the symbol shall result in a grade of 0 for Decode. Reference decode constructs the actual sampling grid which is used to sample the modules of the symbol for the purpose of decoding. In case of an invalid encodation (e.g. 7.1, 7.2 and 7.3 used incorrectly) the symbol shall also result in a grade of 0 for Decode.

9.2.4 Grid non-uniformity

The ideal grid is calculated by using the four corner points in the surrounding fixed pattern /solid area of the sampling grid (see Figure 4) for each data region and subdividing it equally in both axes.

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

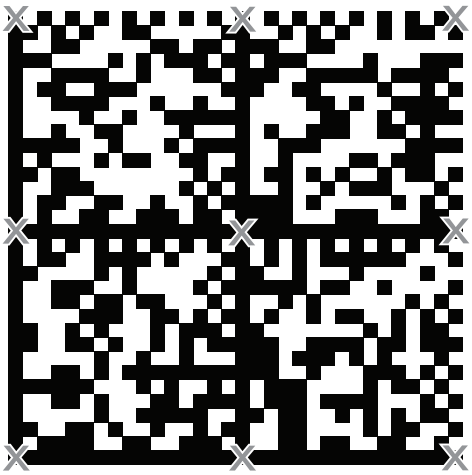


Figure 4 — Corner points of the ideal grid

9.3 Process control measurements

A variety of tools and methods can be used to perform useful measurements for monitoring and controlling the process of creating Data Matrix symbols. These techniques do not constitute a print quality check of the produced symbols. The method specified in 9.2 and in Annex G is the required method for assessing symbol print quality. However, other techniques individually and collectively yield good indications of whether the symbol print process is creating workable symbols.

10 Reference decode algorithm for Data Matrix

This reference decode algorithm finds a Data Matrix symbol in an image and decodes it.

a) Define measurement parameters and form a digitised image.

   1) Define a distance $d_{\min}$ which is 7,5 times the aperture diameter defined by the application. This will be the minimum length of the "L" pattern's side.

   2) Define a distance $g_{\max}$ which is 7,5 times the aperture diameter. This is the largest gap in the "L" finder that will be tolerated by the finder algorithm in step b).

   3) Define a distance $m_{\min}$ which is 1,25 times the aperture diameter. This would be the nominal minimum module size when the aperture size is 80 % of the symbol's X dimension.

   4) Form a black/white image using a threshold determined according to the method defined in ISO/IEC 15415.

b) Search horizontal and vertical scan lines for the two outside edges of the Data Matrix "L".

   1) Extend a scan line horizontally in both directions from the centre point of the image. Sample along the scan line. For each white/black or black/white transition found along the scan line resolved to the pixel boundary perform the following steps.

      i) Follow the edge upward sampling pixel by pixel until either it reaches a point 3,5 $m_{\min}$ distant from the intersection of the scan line and the edge starting point, or the edge turns back toward the intersection of the scan line and the edge – the starting point.

      ii) Follow the edge downward pixel by pixel until either it reaches a point 3,5 $m_{\min}$ distant from the intersection of the scan line and the edge starting point, or the edge turns back toward the intersection of the scan line and the edge – the starting point.

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

iii) If the upward edge reaches a point 3,5 m_{min} from the starting point, perform the following steps.

I) Plot a line A connecting the end points of the upward edge.

II) Test whether the intermediate edge points lie within 0,5 m_{min} from line A. If so, continue to step III. Otherwise proceed to step 1)iv) to follow the edge in the opposite direction.

III) Continue following the edge upward until the edge departs 0,5 m_{min} from line A or the edge turns back for a distance of at least 0,5 m_{min} . Back up along the edge to the closest edge point with an Euclidian distance greater than or equal to m_{min} from the last edge point along the edge before the departing point and save this as the edge end point. This edge point should be along the "L" candidate outside edge.

IV) Continue following the edge downward until the edge departs 0,5 m_{min} from line A or the edge turns back for a distance of at least 0,5 m_{min}. Back up along the edge to the closest edge point with a Euclidian distance greater than or equal to m_{min} from the last edge point along the edge before the departing point and save this as the edge end point. This edge point should be along the "L" candidate outside edge.

V) Calculate a new adjusted line A1 that is a "best fit" line to the edge in the two previous steps. The "best fit" line uses the linear regression algorithm (using the end points to select the proper dependent axis, i.e. if closer to horizontal, the dependent axis is x) applied to each point. The "best fit" line terminates lines at points p1 and p2 that are the points on the "best fit" line closest to the endpoints of the edge.

VI) Save the line A1 segment two end points, p1 and p2. Also save the colour of the left side of the edge viewed from p1 to p2.

iv) If step iii) failed or did not extend upward by 3,5 m_{min} in step iii)IV), test if the downward edge reaches a point 3,5 m_{min} from the starting point. If so, repeat the steps in iii) but with the downward edge.

v) If neither steps iii) or iv) were successful, test if both the upward and downward edges terminated at least 2 m_{min} from the starting point. If so, form an edge comprised of the appended 2 m_{min} length upward and downward edge segments and repeat the steps in iii) but with the appended edge.

vi) Proceed to and process the next transitions on the scan line, repeating from step i), until the boundary of the image is reached.

2) Extend a scan line vertically in both directions from the centre point of the image. Look for line segments using the same logic in step 1) above but following each edge transition first left and then right.

3) Search among the saved line A1 segments for pairs of line segments that meet the following four criteria.

i) If the two lines have the same p1 to p2 directions, verify that the closer of the interline p1 to p2 distances is less than g_{max}. If the two lines have opposite p1 to p2 directions, verify that the closer of the interline p1 to p1 or p2 to p2 distances is less than g_{max}.

ii) Verify that the two lines are co-linear within 5 degrees.

iii) Verify that the two lines have the same saved colour if their p1 to p2 directions are the same or that the saved colours are opposite if their p1 to p2 directions are opposite to each other.

iv) Form two temporary lines by extending each line to reach the point on the extension that is closest to the furthest end point of the other line segment. Verify that the two extended lines are separated by less than 0,5 m_{min} at any point between the two extended lines.

4) For each pair of lines meeting the criteria of step 3) above, replace the pair of line segments with a longer A1 line segment that is a "best fit" line to the four end points of the pair of shorter line

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024

segments. Also save the colour of the left side of the edge of the new longer line viewed from its p1 endpoint to its p2 endpoint.

5) Repeat steps 3) and 4) until no more A1 line pairs can be combined.

6) Select line segments that are at least as long as $d_{\min}$. Flag them as "L" side candidates.

7) Look for pairs of "L" side candidates that meet the following three criteria.

   i) Verify that the closest points on each line are separated by less than 1,5 $g_{\max}$.

   ii) Verify that they are perpendicular within 5 degrees.

   iii) Verify that the same saved colour is on the inside of the "L" formed by the two lines. Note that if one or both lines extend past their intersection, then the two or four "L" patterns formed will need to be tested for matching colour and maintaining a minimum length of $d_{\min}$ for the truncated side or sides before they can become "L" candidates.

8) For each candidate "L" pair found in step 7) form an "L" candidate by extending the segments to their intersection point.

9) If the "L" candidate was formed from line segments with the colour white on the inside of the "L", form a colour inverted image to decode. Attempt to decode the symbol starting with the appropriate normal or inverted image starting from step d) below using each of the "L" candidates from step 8) as the "L" shaped finder. If none decode, proceed to step c).

c) Maintain the line A1 line segments and "L" side candidates from the previous steps. Continue searching for "L" candidates using horizontal and vertical scan lines offset from previous scan lines.

1) Using a new horizontal scan line 3 $m_{\min}$ above the centre horizontal scan line, repeat the process in step b)1), except starting from the offset from the centre point, and then b)3) through b)9). If there is no decode, proceed to the next step.

2) Using a new vertical scan line 3 $m_{\min}$ left of the centre vertical scan line, repeat the process in step b)2), except starting from the offset from the centre point, and then steps b)3) through b)9). If there is no decode, proceed to the next step.

3) Repeat step 1) above except using a new horizontal scan line 3 $m_{\min}$ below the centre horizontal scan line. If there is no decode, repeat step 2) above except using a new vertical scan line 3 $m_{\min}$ right of the centre vertical scan line. If there is no decode, proceed to step 4) below.

4) Continue processing horizontal and vertical scan lines as in steps 1) through 3) that are 3 $m_{\min}$ above, then left, then below, then right of the previously processed scan lines until either a symbol is decoded or the boundary of the image is reached.

d) First assume that the candidate area contains a square symbol. If the area fails to decode as a square symbol, then try to find and decode a rectangular symbol starting from procedure j). For a square symbol, first plot a normalised graph of transitions for the equal sides of the candidate area in order to find the alternating module finder pattern.

1) Project a line through the candidate area bisecting the interior angle of the two sides of the "L" found above as shown in Figure 5. Define the two equal areas formed by the bisecting line as the right side and the left side as viewed from the corner of the "L".

2) For each side, form a line called a “search line” between a point $d_{\min}$ distance from the corner along the “L” line, parallel to the other “L” side line, and extending to the bisecting line as shown in Figure 5.

3) Move each search line away from the corner of the “L” as shown in Figure 5, lengthening each line as it expands to span its two bounding lines, the “L” line and the bisecting line. Keep each search line parallel to the other “L” side line. As each side is moved by the size of an image pixel, count the number of black/white and white/black transitions, beginning and ending the count with transitions from the colour of the “L” side to the opposite colour. A transition from one colour to

© ISO/IEC 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 16022:2024