INTERNATIONAL STANDARD

ISO/IEC 24723

Second edition
2010-08-01

Information technology — Automatic identification and data capture techniques — GS1 Composite bar code symbology specification

Technologies de l'information — Techniques automatiques d'identification et de capture des données — Spécifications de la symbologie des codes à barres du Composant GS1

Reference number
ISO/IEC 24723:2010(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

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. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 24723 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 31, *Automatic identification and data capture techniques*.

This second edition cancels and replaces the first edition (ISO/IEC 24723:2006), which has been technically revised.

Introduction

Composite symbologies are a class of bar code symbology, the principal distinguishing feature of which is that they comprise two, or more, components, each of which is a distinct symbol, but which contain a set of related data. Typically one component is a linear symbol containing primary data, which can be read on its own in some areas of the application. The other component(s) is a two-dimensional symbol containing supplementary data which qualifies the primary message, and requiring all components to be read to extract the complete message. The GS1 Composite symbology is one such symbology. The use of the symbology is intended to comply with the GS1 General Specifications.

A GS1 Composite symbol consists of a linear component (encoding the item's primary identification) associated with an adjacent 2D component (encoding supplementary data, such as a batch number or expiration date). The GS1 Composite symbol always includes a linear component so that the primary identification is readable by all scanning technologies, and so that 2D imagers can use the linear component as a finder pattern for the adjacent 2D component. The GS1 Composite symbol always includes a multi-row 2D component, for compatibility with linear and 2D imagers, and with linear and rastering laser scanners.

GS1 Composite symbols are intended for encoding identification numbers and data supplementary to the identification in accordance with the GS1 General Specifications. The administration of the numbering system by GS1 ensures that identification codes assigned to particular items are unique world-wide and that they and the associated supplementary data are defined in a consistent way.

Information technology — Automatic identification and data capture techniques — GS1 Composite bar code symbology specification

1 Scope

This International Standard defines the requirements for the GS1 Composite symbology. It specifies the GS1 Composite symbology characteristics, data character encodation, symbol formats, dimensions and print quality requirements, error correction rules, and reference decoding algorithms. For those linear and 2D components of GS1 Composite symbols with published symbology specifications, those published specifications apply, except as specifically noted in this International Standard.

2 Normative references

The following referenced documents are indispensable for the application 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 15415, *Information technology — Automatic identification and data capture techniques — Bar code print quality test specification — Two-dimensional symbols*

ISO/IEC 15416, *Information technology — Automatic identification and data capture techniques — Bar code print quality test specification — Linear symbols*

ISO/IEC 15417, *Information technology — Automatic identification and data capture techniques — Code 128 bar code symbology specification*

ISO/IEC 15420, *Information technology — Automatic identification and data capture techniques — EAN/UPC bar code symbology specification*

ISO/IEC 15438, *Information technology — Automatic identification and data capture techniques — PDF417 bar code symbology specification*

ISO/IEC 19762-1, *Information technology — Automatic identification and data capture (AIDC) techniques — Harmonized vocabulary — Part 1: General terms relating to AIDC*

ISO/IEC 19762-2, *Information technology — Automatic identification and data capture (AIDC) techniques — Harmonized vocabulary — Part 2: Optically readable media (ORM)*

ISO/IEC 24728, *Information technology — Automatic identification and data capture techniques — MicroPDF417 bar code symbology specification*

GS1 General Specifications (GS1, Brussels, Belgium)

3 Terms, definitions, abbreviated terms and mathematical operators

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 19762-1, ISO/IEC 19762-2 and the following apply.

3.1.1
2D component
CC
two-dimensional portion of a GS1 Composite symbol, which encodes supplemental information about an item, such as its lot number or expiration date

3.1.2
AI element string
character string containing an application identifier followed by its associated data field

3.1.3
CC-A
2D component that is a structural variant of MicroPDF417

NOTE 1 CC-A components can be autodiscriminated from MicroPDF417 symbols, because CC-A components have a unique Rotation of 32 between any two adjacent Row Address Patterns.

NOTE 2 CC-A is one of the three choices for the 2D component in a symbol encoded in the GS1 Composite symbology.

3.1.4
CC-B
2D component that is a MicroPDF417 symbol which begins with PDF417 codeword 920, indicating conformance with this International Standard

NOTE CC-B is one of the three choices for the 2D component in a symbol encoded in the GS1 Composite symbology.

3.1.5
CC-C
2D component that is a PDF417 symbol which begins with PDF417 codeword 920, indicating conformance with this International Standard

NOTE CC-C is one of the three choices for the 2D component in a symbol encoded in the GS1 Composite symbology.

3.1.6
linear component
linear portion of a GS1 Composite symbol, which encodes the primary identification of an item

3.1.7
linkage flag
indicator encoded in a GS1 DataBar or GS1-128 component to signal if a 2D component accompanies the linear component

3.1.8
RAP Rotation
difference between the number designating a Center or Right Row Address Pattern and the number designating the nearest Row Address Pattern to the left, in the same row of a MicroPDF417 symbol, or a CC-A or CC-B component

3.1.9
Row Address Pattern
RAP
one of a set of patterns made up of three bars and three spaces occupying ten modules, that serve both as start or stop patterns and as row indicators in a MicroPDF417 symbol, or a CC-A or CC-B component

3.1.10
separator pattern
pattern between the linear and 2D components of a GS1 Composite symbol

3.1.11
symbol separator character
non-data character that is used to break the transmitted data string into separate transmissions, each beginning with the appropriate symbology identifier prefix

3.2 Abbreviated terms

AI Application Identifier

3.3 Mathematical operators and notational conventions

For the purposes of this document, the following mathematical operators apply.

div integer division operator which discards the remainder

mod integer remainder after integer division

The following ISO notational conventions are used.

0,2 a comma between digits separates the integer from the decimal fraction (e.g. 0,2 equals 2/10) except when used as an (n,k) designation

12 345 a space between digits indicates factors of a thousand

4 Symbol description

4.1 Basic characteristics

A GS1 Composite symbol consists of a linear component associated with an adjacent 2D component. The characteristics of the GS1 Composite symbology are:

a) Encodable character set:

   1) Both linear and 2D components encode a subset of ISO/IEC 646, consisting of the upper and lowercase letters, digits, and 21 selected punctuation characters.

   2) The function character FNC1 and a Symbol Separator character.

b) Symbol character structure: various edge-to-similar-edge decodable symbol characters are used, in accordance with the selected Linear and 2D components of the symbol.

c) Code type:

   1) Linear component: continuous, linear bar code symbology.

   2) 2D component: continuous, multi-row bar code symbology.

d) Maximum numeric data capacity (including implied application identifiers and calculated check digits where appropriate, but not including any encoded FNC1 characters):

1) Linear component:

i) GS1-128: up to 48 digits

ii) EAN/UPC: UPC-A, EAN-8, or EAN-13 (12, 8, or 13 digits respectively)

iii) GS1 DataBar Expanded: up to 74 digits, see note

NOTE The GS1 DataBar Expanded data capacity depends on the encodation method. The maximum is 74 digits for (01) + other AI's, the maximum is 70 digits for any AI's, and the maximum is 77 digits for (01) + (392x) + any AI's.

iv) Other GS1 DataBar: 16 digits

2) 2D component:

i) CC-A: up to 56 digits

ii) CC-B: up to 338 digits

iii) CC-C: up to 2 361 digits

e) Error detection and correction:

1) Linear component: one check character for error detection.

2) 2D component: a fixed or variable number of Reed-Solomon error correction codewords, depending upon the specific 2D component.

f) Character self-checking: yes.

g) Bi-directionally decodable: yes.

4.2 Summary of additional features

The following is a summary of additional GS1 Composite symbology features:

a) **Data compaction**: the 2D components utilize a bit-oriented compaction mode, designed to encode efficiently data using application identifiers.

b) **Component linkage**: the 2D component of each GS1 Composite symbol contains a linkage flag, which indicates to the reader that no data shall be transmitted unless the associated linear component is also scanned and decoded. All linear components except EAN/UPC also contain an explicit linkage flag.

c) **GS1-128 emulation**: readers set to the GS1-128 emulation mode transmit the data encoded within the GS1 Composite symbol as if the data were encoded in one or more GS1-128 symbols.

d) **2D component escape mechanism**: a mechanism to support future applications which require data content beyond the ISO/IEC 646 subset encodable in the standard form of the GS1 Composite symbology.

4.3 Symbol structure

Each GS1 Composite symbol consists of a linear component and a multi-row 2D component. The 2D component should nominally be printed with the same X dimension as the linear component. The 2D component is printed above the linear component (as defined in 12.2 and 12.3).

To facilitate printing the two components independently, 12.2 and 12.3 permit relative positional variation between the two components. A bar code reader should not use the relative locations of the components shown in the illustrations of GS1 Composite symbols in this specification to predict the exact location of the 2D component from the location of the linear component.

The linear component is one of:

a) a member of the EAN/UPC symbology (EAN-13, EAN-8, UPC-A, or UPC-E, which may include an add-on symbol, in accordance with ISO/IEC 15420),

b) a member of the GS1 DataBar symbology family,

c) GS1-128.

The choice of linear component determines the name of the GS1 Composite symbol, such as an EAN-13 Composite symbol, or a GS1-128 Composite symbol.

The 2D component (abbreviated as CC) is chosen based on the selected linear component, and on the amount of supplementary data to be encoded. The three 2D components, listed in order of increasing maximum data capacity, are:

a) *CC-A*, a variant of MicroPDF417, designed for efficient encoding of supplemental application identifier data,

b) *CC-B*, a MicroPDF417 symbol with a codeword of 920 in the first data codeword position as a linkage flag, and denoting GS1 data compaction, and

c) *CC-C*, a PDF417 symbol with a codeword of 920 in the first data codeword position as a linkage flag, and denoting GS1 data compaction.

Figure 1 — A GS1 DataBar Limited Composite symbol

Figure 1 illustrates a GS1 DataBar Limited Composite symbol which utilizes a 4-row CC-A component as its 2D component. The GS1 DataBar Limited component in Figure 1 identifies the product as “0113112345678906”, and the CC-A component encodes the expiration date and lot number (as “1701061510A123456”). The human-readable interpretation of the data in the symbols would be shown, if present, as “(01)13112345678906” and “(17)010615(10)A123456” respectively.

Figure 2 — A GS1-128 Composite symbol

Figure 2 illustrates a GS1-128 Composite symbol which utilizes a 5-row CC-C component as its 2D component. The GS1-128 component in Figure 2 identifies the product as “0193812345678901”, and the CC-C component encodes the lot number and deliver to location number (as “10ABCD123456<FNC1>4103898765432108”). The human-readable interpretation of the data in the symbols would be shown, if present, as “(01)93812345678901” and “(10)ABCD123456(410)3898765432108” respectively.

4.4 Supported component combinations

Based upon the width of the linear component, a choice of “best-fit” 2D component is specified. Table 1 lists all of the permissible combinations.

Table 1 — Permissible combinations of linear and 2D components

Linear component	CC-A/CC-B	CC-C
UPC-A and EAN-13	Yes (4-columns)	No
EAN-8	Yes (3-columns)	No
UPC-E	Yes (2-columns)	No
GS1-128	Yes (4-columns)	Yes (variable width)
GS1 DataBar Omnidirectional	Yes (4-columns)	No
GS1 DataBar Stacked	Yes (2-columns)	No
GS1 DataBar Stacked Omnidirectional	Yes (2-columns)	No
GS1 DataBar Limited	Yes (3-columns)	No
GS1 DataBar Expanded	Yes (4-columns)	No
GS1 DataBar Expanded Stacked	Yes (4-columns)	No

In all cases where a CC-A component is shown in the table, the printing software shall automatically use a CC-B component (of the same number of columns) when the data to be encoded exceeds the maximum capacity of the CC-A or other means shall be provided to enable the correct selection of CC-A or CC-B to suit the length of the data. The presence of an add-on symbol shall not affect the choice of 2D component. When the linear component is GS1-128, either CC-A/CC-B or CC-C may be used; see 7.3.

5 Source data encodation into a binary string

5.1 General

The user data to be encoded into a 2D component shall always consist of application identifiers and data fields that comply with the data standard of the GS1 General Specifications. The GS1-128 rules for concatenation of AI element strings, such as the termination of a variable-length string by a FNC1 character, shall be followed when encoding a 2D component.

Before encoding a 2D component, the given data string is encoded into a binary string, where data characters are represented with a variable number of bits. The resulting binary string consists of two or three binary fields. The fields are:

a) Encodation Method (see 5.2),

b) compressed data (see 5.3), and

c) general-purpose data compaction (see 5.4).

The Encodation Method is always encoded as the first field (see 5.2). One or both of the other fields are also encoded in a 2D component. When both are present they are encoded in the order shown above.

The compressed data field efficiently encodes data using specific AIs, such as date and lot number or data using AI 90 and serial numbers (see 5.3).

The general-purpose compaction field can encode data using any combination of AIs (see 5.4).

In the text of this clause, bit fields will be indicated by their binary values enclosed in double quotation marks.

5.2 Encodation Method field

The Encodation Method field consists of one or two bits. It occupies the first bits in the encoded binary data. It defines whether the symbol is a general-purpose symbol or begins with an application-oriented compressed data field (such as for efficient representation of expiration date and lot number AI element strings). The Encodation Method field is defined in Table 2.

Table 2 — Encodation Method field

Encodation Method field bits	Description
0	AI element string data using general-purpose data compaction
10	Date, lot number, and other AI element strings
11	AI 90 (containing alphanumeric data) optionally followed by other AI element strings

An Encodation Method field of “0” is directly followed by AI data encoded using general-purpose data compaction (see 5.4). The encoded data starts with the first application identifier and its element string data, optionally followed by one or more AI element strings.

An Encodation Method field of “10” or “11” is directly followed by a compressed data field, optionally followed by a general-purpose data compaction field (see 5.4).

5.3 Compressed data field

If the Encodation Method field is other than a single “0” bit, this indicates the presence of a field of compressed data. The compression is specified to encode the data of specific AI element strings efficiently. An Encodation Method field of “10” (see 5.3.1) is directly followed by a compressed data field suited for marking pharmaceuticals and many other small items requiring an Expiration or Production Date and/or Lot number. An Encodation Method field of “11” (see 5.3.2) is directly followed by a compressed data field suited for many applications for marking small products with alphanumeric data. It may be necessary to sort the AI element strings into the required sequence in order to benefit from the applicable Encodation Method.

5.3.1 Encodation Method field “10” — date and lot number

This Encodation Method may be used if the element strings of AI 11 followed by AI 10, AI 17 followed by AI 10, or the single AIs 10, 11 or 17 occur at the start of the data message to be encoded. An Encodation Method field of “10” is followed by a compressed date field (which may be empty), an optional lot number, possibly followed by other AI element strings (which if present, are encoded using general-purpose data compaction).

The compressed date field following the “10” Encodation Method field consists of either:

a) A 17-bit field, consisting of:

   1) a 16-bit value encoding 0 to 38 399 for YYMMDD (year, month, and day) according to the equation

      (YY x 384) + ((MM-1) x 32) + (DD)

   2) followed by a single bit indicating the type of date: “0” for Production Date (AI 11) or “1” for Expiration Date (AI 17)

or:

b) A 2-bit date field of “11” which is a flag indicating that no date was encoded.

NOTE 1 The 16-bit date values 0-38 399 will start with “0” or “10” so that “11” is a unique two-bit flag.

NOTE 2 The YYMMDD format for data is required to comply with GS1 application specifications. Separate algorithms are provided to convert the date into its format of CCYYMMDD.

The date is encoded by stripping the two-digit AI 11 or AI 17 from the date element string and encoding the remaining six digits into 16 bits using the equation above. For AI 11, the next bit is “0”; for AI 17 the next bit is “1”. If there was no date, a 2-bit field of “11” is encoded instead of the 17-bit field.

If a lot number directly follows the date element string, the two-digit AI 10 is stripped from the lot number element string, and the remainder of the lot number element string is encoded using general-purpose data compaction, directly following the date field. If more AI element strings follow the lot number, a FNC1 separates the lot number data from the next AI element string to be encoded.

If a lot number does not directly follow the date element string, a FNC1 is encoded following the date element string, even if no more data follows the date element string (this FNC1 shall not be transmitted by the decoder). If more data follows, it is encoded using general-purpose data compaction beginning with the digits of the next AI.

The decoder shall reconstruct the AI element strings from a compressed data field using an Encodation Method field of “10” according to the following procedure:

a) If the bits “11” follow the method “10”, no date is encoded, and the decoder shall insert the two-digit AI 10 before the remaining general-purpose data compaction field is decoded.

b) Otherwise, the 6-digit date shall be extracted according to the above equation. If the seventeenth bit is “0”, an AI 11 for production date is added as a prefix by the decoder. If the seventeenth bit is “1”, an AI 17 for expiration date is added. If the first encoded data character following the date field is FNC1, there is no lot number (this FNC1 shall not be transmitted by the decoder). Otherwise the decoder shall insert the two-digit AI 10 before the remaining general-purpose data compaction field is decoded.

5.3.2 Encodation Method field of “11” — AI 90

This Encodation Method may be used if an element string with an AI 90 occurs at the start of the data message, and if the data field following the two-digit AI 90 starts with an alphanumeric substring which complies with a specific format. The format of the alphanumeric data that can be used in this compaction method is 0, 1, 2, or 3 digits (strings with leading zeros do not comply with the required format) followed by an uppercase alphabetic character.

An Encodation Method field of “11” is followed by a compressed data field which includes the encoded special-format alphanumeric string, followed by the remainder of its data field, optionally additional compressed data fields, and optionally additional AI element strings.

The compressed data field following the “11” Encodation Method consists of the following:

a) One or two bits indicating the starting data encodation scheme used for the remainder of the AI 90 data field. A “0” indicates Alphanumeric encodation, a “10” indicates Numeric encodation and an “11” indicates Alpha encodation (see 5.3.3).

b) One or two bits indicating the absence or presence, respectively, of specific AIs after the first FNC1 (which terminates the AI 90 element string, unless no further data is encoded). “0” indicates that either no more data is encoded, or that the remaining data is encoded according to general-purpose data compaction rules. Otherwise, the AI of the next element string will not be explicitly encoded. “10” indicates that an AI 21 follows, and “11” indicates that an AI 8004 follows.

c) Nine or 20 bits encoding the 1 to 4 characters of the alphanumeric string that followed the AI 90 in the source message, encoded as follows:

   1) Convert the numeric portion of the alphanumeric string to a value. If the string contains no numeric digits, the value is 0.

2) If the numeric value is less than 31, and if the uppercase alphabetic letter is one of those listed in Table 3, encode the numeric value as a five-bit binary string. Then convert the uppercase alphabetic character to four bits of binary data using Table 3 and encode these as a four-bit binary string.

3) Otherwise encode a five-bit string “11111”, followed by a 10-bit string representing the value of the numeric digits, followed by a five-bit string representing the ASCII value of the letter minus 65 (where “A” is encoded as “00000”, and “Z” is encoded as “11001”).

Table 3 — Supported uppercase alphabetic letters

Letter	Binary Data	Letter	Binary Data
B	0000	P	1000
D	0001	Q	1001
H	0010	R	1010
I	0011	S	1011
J	0100	T	1100
K	0101	V	1101
L	0110	W	1110
N	0111	Z	1111

During encoding, the AI 90 and the alphanumeric string are stripped from the first AI element string. The next two to four bits specify the starting encodation scheme for the remainder of the element string, and whether an AI 21 or AI 8004 element string follows the AI 90 element string. Then the compacted alphanumeric string is encoded in the next nine or twenty bits. The remainder of the AI 90 element string is encoded using either general-purpose data compaction starting in Alphanumeric or Numeric schemes, or the Alpha encodation scheme, as specified by the encodation scheme bit field. The AI element string data is terminated by an encoded FNC1, unless it is the last AI element string of the input data.

If an AI 21 or AI 8004 element string follows the AI 90 element string, the 21 or 8004 is stripped from the element string, and the remainder of the AI element string is encoded, starting in the default Numeric encodation scheme of general-purpose data compaction. If the AI 90 element string is encoded using the numeric encodation scheme when the FNC1 is encountered, the first digit of the following AI element string may be encoded with the FNC1 in the same seven bits of the numeric encodation scheme. Any additional data is then encoded.

The decoder shall reconstruct the AI element strings from a “11” compressed data field according to the following procedure.

a) The 2-digit AI 90 is inserted to start the decoded data string.

b) The first two to four bits are decoded to determine the encodation scheme and whether a specific element string of AI 21 or AI 8004 follows the AI 90 element string.

c) The next five bits are decoded to determine the number of digits in the alphanumeric string and their value.

   1) If the value of the five bits is less than 31, these five bits are decoded to determine the number of digits and their value (discarding any leading zeros), and the next four bits are decoded using Table 3 to determine the uppercase alphabetic character in the alphanumeric string.

   2) Otherwise, the ten bits following the “11111” are decoded to determine the number of digits and their value (discarding any leading zeros); and the next five bits are decoded to determine the uppercase letter in the alphanumeric string.

d) Append the alphanumeric string to the decoded data string.

e) The bits up to the first FNC1 are decoded starting with the encodation scheme specified by the first one or two bits.

f) If the special-AI bit field was “0”, proceed to the next step below. Otherwise, the bit field was “10” or “11”, add the two-digit AI 21 or the four-digit AI 8004, respectively, to the decoded data string directly following the last decoded FNC1.

NOTE the leading digit of the data field following AI element string’s data field may have already been decoded if it was paired with the FNC1 in Numeric encodation.

g) Starting from Numeric encodation, continue decoding the general-purpose field following the FNC1 character.

The Symbol Separator character also can terminate an AI element string. It is processed in the same way as FNC1 as described in this clause, except that it cannot be encoded in Numeric or Alpha compaction schemes.

5.3.3 Alpha encodation

Alpha encodation, available only from within the “11” Encodation Method as specified in 5.3.2, encodes uppercase Alphabetic characters into 5 bits each, and encodes digits into 6 bits each. This mode also includes a 5-bit FNC1/numeric latch character. The mode remains in effect until the next FNC1 (encoded as the FNC1/numeric latch) or the end of the data message. Each character is encoded as shown in Table 4.

If, after all of the source message has been encoded in Alpha encodation and any bits remain in the symbol capacity, then a FNC1/numeric latch shall be encoded. This FNC1 is truncated if fewer than five bits remain in the symbol. If necessary, the remaining symbol capacity shall be padded as described in 5.4.4. The reader shall not transmit this trailing FNC1.

Table 4 — Alpha encodation

Character(s)	ASCII value(s)	Encoded value	Encoded binary data
A to Z	65 to 90	ASCII - 65 (5-bit)	00000 to 11001
0 to 9	48 to 57	ASCII + 4 (6 bit)	110100 to 111101
FNC1/numeric latch		31 (5-bit)	11111

The Alpha mode should be used in the AI 90 compaction method if:

a) The remaining data characters up to the next FNC1 (or to the end of the message if there is no FNC1) are either digits or uppercase letters, and

b) The number of letters is greater than the number of digits.

The data is encoded by appending the variable length binary data for each character, up to and including the next FNC1 or the end of the message.

The encoded bit field is decoded by first examining the first five bits in the field following the previously decoded character or initially at the start of the field.

a) If the value of these five bits is less than 26, it is one of the 5-bit characters in the set; decode these 5 bits as an uppercase letter according to Table 4.

b) If the value of the five bits is 31, then it is a 5-bit FNC1/numeric latch character.

c) Otherwise, decode the five bits together with the following bit as a digit according to Table 4.

5.4 General-purpose data compaction field

The general-purpose data compaction field is preceded either by an Encodation Method field of “0”, or by a compressed data field. It encodes AI element string data into a binary field, using a combination of three encodation schemes:

a) Numeric encodation is used for numbers and FNC1 characters, requiring 3,5 bits per encoded character;

b) Alphanumeric encodation is used for a mix of numbers, uppercase letters, and a subset of punctuation characters, requiring 5 bits per digit or FNC1 character, and 6 bits per letter or punctuation character;

c) ISO/IEC 646 encodation is used for a mix of uppercase and lowercase letters, numbers, and most punctuation characters, requiring 5 bits per digit or FNC1, 7 bits per letter, and 8 bits per punctuation character.

General-purpose data compaction is used to encode the AI element strings other than those specifically encoded using one of the compressed data Encodation Methods defined in 5.3. The general-purpose data compaction field, if present, is the last field in the symbol. Once all of the data has been processed using general-purpose data compaction scheme(s), the general-purpose encoding process concludes with the padding procedure described in 5.4.4.

5.4.1 Numeric encodation

Numeric encodation is the default encodation scheme that is in effect at the start of the general-purpose data compaction field. It encodes two digits or a digit and FNC1 character (in either order) in seven bits. These seven bits represent the decimal value V, determined by:

$$V = (11\ D_1) + D_2 + 8$$

where D_1 and D_2 are the first and second digit values or FNC1 value (FNC1 is assigned a decimal value of 10). The decimal value V ranges from 8 to 127 corresponding to the binary strings “0001000” to “1111111” (two FNC1 characters cannot be encoded in seven bits). A binary sequence of “0000” at the start of the field or following the end of a previous Numeric encodation seven-bit value signals a latch, or change, to alphanumeric encoding (see Table 5).

Table 5 — Numeric encodation

Character(s)	Encoded binary data
digit-digit, digit-FNC1, and FNC1-digit pairs	0001000 to 1111111
Alphanumeric latch	0000

A Numeric encodation sequence continues to encode pairs of data characters until one of the following conditions becomes true:

a) If at least two characters remain and Numeric encodation cannot be applied to both of the next two characters, encode an Alphanumeric latch in the data compaction field.

b) If one character remains, which is not a digit, encode an Alphanumeric latch in the data compaction field.

c) If one character remains, which is a digit, first calculate the symbol size needed to encode the current binary string, and then the number of unused bits. (The number of unused bits is calculated by determining the minimum number of rows in the given symbol required to encode the data bit string up to this point. The number of unused bits is the difference between the number of bits in the current encoded data bit string and the data bit capacity of the symbol of that size.)

   1) If seven or more unused bits remain, encode the digit and a FNC1 pad in the next seven bits. This trailing FNC1 will be recognized as a pad and will not be transmitted by the reader.

2) If four to six bits remain, add 1 to the digit value and encode the result in the next four bits. The fifth and sixth bits, if present, shall be “0”s.

3) Otherwise the next larger symbol size will be used, encoding the digit and a FNC1 pad in the next seven bits. This trailing FNC1 will be recognized as a pad and will not be transmitted by the reader. Any remaining bits are encoded according to the padding procedure of 5.4.4.

d) No characters remain: any remaining bits are encoded according to the padding procedure of 5.4.4.

Whenever an Alphanumeric latch is encoded, the encodation scheme is changed to reflect that latch. If the next data character requires ISO/IEC 646 encodation, the Alphanumeric encodation scheme will immediately encode an ISO/IEC 646 latch following the Alphanumeric latch.

During decoding the following special checks shall be made when numeric compaction is in effect at the end of the symbol:

a) If the last seven bits of Numeric encodation immediately before the pad sequence encode a digit followed by a FNC1, the FNC1 is ignored.

b) If Numeric encodation is in effect when only four to six bits remain in the symbol, then the value of the 4-bit string at the start of the remaining bits is converted into its decimal value.

1) If the value is zero, then the data message is complete.

2) Otherwise, it is decoded as a final digit in the data message equal to the decimal value minus one.

5.4.2 Alphanumeric encodation

Alphanumeric encodation encodes the digits and FNC1/numeric latch, the uppercase letters, five punctuation characters, a Symbol Separator/numeric latch character, and two latch characters. The encoded bit stream does not have a fixed bit length per character. Each character is encoded in from three to six bits as shown in Table 6. The Symbol Separator/numeric latch character listed in the table is used to break the data string into separate transmissions, each beginning with the appropriate symbology identifier.

Table 6 — Alphanumeric encodation

Character(s)	ASCII value(s)	Encoded value	Encoded binary data
0 to 9	48 to 57	ASCII value minus 43 (5-bit)	00101 to 01110
FNC1/numeric latch		15 (5-bit)	01111
A to Z	65 to 90	ASCII value minus 33 (6-bit)	100000 to 111001
* (asterisk)	42	58 (6-bit)	111010
, (comma)	44	59 (6-bit)	111011
- (minus or hyphen)	45	60 (6-bit)	111100
. (period or full stop)	46	61 (6-bit)	111101
/ (slash or solidus)	47	62 (6-bit)	111110
Symbol Separator/numeric latch		63 (6-bit)	111111
Numeric latch		0 (3-bit)	000
ISO/IEC 646 latch		4 (5-bit)	00100

The data is encoded by appending the variable length binary data for each character to the general-purpose data compaction field, according to the following rules:

a) If the next data character is a FNC1, encode it in alphanumeric encodation.

b) If the next data character requires ISO/IEC 646 encodation, encode an ISO/IEC 646 latch in the data compaction field.

c) If the next six data characters can be encoded using Numeric encodation, encode a Numeric latch in the data compaction field.

d) If the next four or more data characters can be encoded using Numeric encodation and they terminate the data string, encode a Numeric latch in the data compaction field.

Whenever a latch is encoded, the encodation scheme is changed to reflect that latch.

The encoded bit field is decoded by first examining the first one or three bits in the field following the previously decoded character or initially at the start of the field.

a) If the first bit is “1”, decode the character as a 6-bit character.

b) If the first three bits are “000”, it is a numeric latch.

c) Otherwise, decode the character as a 5-bit character.

5.4.3 ISO/IEC 646 encodation

ISO/IEC 646 encodation encodes the characters in, which includes a subset of those in ISO/IEC 646, together with a FNC1/numeric latch, a Symbol Separator/numeric latch, a Numeric latch, and an Alphanumeric latch. The encoded bit stream does not have a fixed bit length per character. The bits are allocated according to the bit length of each encoded character. Each character is encoded in from three to eight bits as shown in Table 7. The Symbol Separator/numeric latch character listed in the table is used to break the data string into separate transmissions, each beginning with the appropriate symbology identifier.

Table 7 — ISO/IEC 646 encodation

Character(s)	ASCII value(s)	Encoded value	Encoded binary data
0 to 9	48 to 57	ASCII value – 43 (5-bit)	00101 to 01110
FNC1/numeric latch		15 (5-bit)	01111
A to Z	65 to 90	ASCII value – 1 (7-bit)	1000000 to 1011001
a to z	97 to 122	ASCII value – 7 (7-bit)	1011010 to 1110011
! (exclamation mark)	33	232 (8-bit)	11101000
" (quotation mark)	34	233 (8-bit)	11101001
% (percent sign)	37	234 (8-bit)	11101010
& (ampersand)	38	235 (8-bit)	11101011
' (apostrophe)	39	236 (8-bit)	11101100
((left parenthesis)	40	237 (8-bit)	11101101
) (right parenthesis)	41	238 (8-bit)	11101110
* (asterisk)	42	239 (8-bit)	11101111
+ (plus sign)	43	240 (8-bit)	11110000
, (comma)	44	241 (8-bit)	11110001
- (minus or hyphen)	45	242 (8-bit)	11110010
. (period or full stop)	46	243 (8-bit)	11110011
/ (slash or solidus)	47	244 (8-bit)	11110100
: (colon)	58	245 (8-bit)	11110101
; (semicolon)	59	246 (8-bit)	11110110
< (less-than sign)	60	247 (8-bit)	11110111

Character(s)	ASCII value(s)	Encoded value	Encoded binary data
= (equals sign)	61	248 (8-bit)	11111000
> (greater-than sign)	62	249 (8-bit)	11111001
? (question mark)	63	250 (8-bit)	11111010
_ (underline or low line)	95	251 (8-bit)	11111011
space	32	252 (8-bit)	11111100
Symbol Separator/numeric latch		253 (8-bit)	11111101
Numeric latch		0 (3-bit)	000
Alphanumeric latch		4 (5-bit)	00100

The data is encoded by appending the variable length binary data for each character to the data compaction field according to the following rules:

a) If the next data character is a FNC1, encode it in ISO/IEC 646 encodation.

b) If none of the next ten characters requires ISO/IEC 646 encodation and the first four characters can be encoded in Numeric Compaction, encode a Numeric latch in the data compaction field.

c) If none of the next ten characters requires ISO/IEC 646 encodation and the first five characters can be encoded in Alphanumeric Compaction, encode an Alphanumeric latch in the data compaction field.

With either the second or third rule, if the data terminates in less than ten characters, then the ten-character test completes early at the end of the data. If a latch is encoded, the encodation scheme is changed to reflect that latch.

The encoded bit field is decoded by first examining the first three or five bits following the previously decoded character or initially at the start of the field:

a) If the first three bits are “000”, it is a numeric latch.

b) Otherwise, get the decimal value of the first five bits. If the value is:

   1) 15 or less, decode the character as a 5-bit character;

   2) 16 to 28, decode the character as a 7-bit character;

   3) 29 or more, decode the character as an 8-bit character.

5.4.4 Pad bits for the general-purpose data compaction field

The number of rows in the symbol should be the minimum required for encoding the data in that symbol. However, there may be unused bits in the symbol after the data is encoded. These bits shall be filled with a pad bit sequence until the data capacity of the symbol is filled.

NOTE For CC-B and CC-C the standard MicroPDF417 and PDF417 pad codewords are not used.

The padding bit string is created by repeated five-bit pad sequences of “00100”, which is both the ISO/IEC 646 latch in Alphanumeric encodation and the Alphanumeric latch in ISO/IEC 646 encodation, so that the encodation modes and their latches alternate without encoding more data. The last pad sequence may be truncated on the right if there are not enough bits left in the symbol.

If Numeric encodation ends the data encodation, a four-bit alphanumeric latch of “0000” is required before the “00100” alternating latch pad sequence. For example, if the encoding ends in Numeric encodation and there are seven remaining bits, they shall be encoded as “0000001”, which is the alphanumeric latch “0000” followed by the first three bits “001” of the ISO/IEC 646 latch of “00100”. Also the first four-bit latch may be shortened if fewer than four unused bits remain to be padded.

6 Error correction

The 2D components all use the same Reed-Solomon error correction method as defined in ISO/IEC 24728 (for CC-A and CC-B) and ISO/IEC 15438 (for CC-C). The specific number of error correction codewords for each CC-A version is defined in Table 9; for CC-B the number is defined in ISO/IEC 24728; for CC-C the number shall be at least the minimum recommended number defined in ISO/IEC 15438.

Because of the requirement to reserve some error correction codewords for error detection, the maximum number of erasures and substitution errors that can be corrected follow the formulae given in ISO/IEC 15438 and ISO/IEC 24728. CC-A components with four error correction codewords are the exception; they have an error correction capacity of one erasure or one substitution error. This exception can be expressed as the following formulae:

a) error correction capacity when there are no erasures:

$$2f + L \leqslant k - 2$$

b) otherwise the error correction capacity is:

$$2f + L \leqslant k - 3$$

NOTE The L, f and k are defined in ISO/IEC 24728.

7 Linear component requirements

7.1 General

The linear component of a GS1 Composite symbol shall always be either an EAN/UPC symbol, a GS1 DataBar-family symbol, or a GS1-128 symbol. The linear component always encodes the primary identification of the item, as defined by the GS1 specifications. In each case, the specification for the linear symbology referenced in Clause 2 applies, except as specifically noted in the subsections of this clause. Clause 12 defines the relative positioning of the linear and 2D components.

7.2 EAN/UPC linear components

EAN/UPC symbols are defined in ISO/IEC 15420. When used as the linear component of a GS1 Composite symbol, they shall be positioned relative to the 2D component as described in Clause 12, with the addition of an extended bar pattern above the first and last guard bars (see 11.4). The presence of an add-on symbol shall not affect the choice of or positioning of the 2D component.

NOTE There is no linkage flag in the EAN/UPC symbol to indicate the presence of an associated 2D component.

To facilitate autodiscrimination processing, restrictions are placed on which version of CC-A or CC-B may appear above an EAN/UPC component, as follows (see 4.4):

a) UPC-E linear components may only be linked to two-column CC-A or CC-B components;

b) EAN-8 linear components may only be linked to three-column CC-A or CC-B components;

c) EAN-13 and UPC-A linear components may only be linked to four-column CC-A or CC-B components.

CC-C components are never used in a GS1 Composite symbol containing an EAN/UPC linear component.

Figure 3, Figure 4 and Figure 5 illustrate the three types of EAN/UPC components linked to CC-A components.

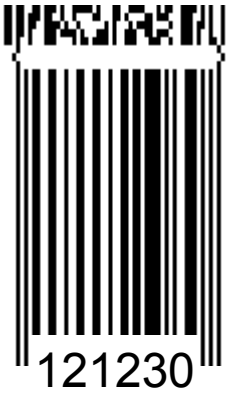


Figure 3 — A UPC-E Composite symbol (with CC-A), encoding 012000001239 and 15021231 respectively

Figure 4 — An EAN-8 Composite symbol (with CC-A), encoding 12345670 and 21A12345678 respectively

Figure 5 — An EAN-13 Composite symbol (with CC-A), encoding 3312345678903 and 991234-abcd respectively

7.3 GS1 DataBar family linear components

GS1 DataBar symbols are defined in the GS1 DataBar specification. When used as the linear component of a GS1 Composite symbol, the encoded value shall include a linkage flag, indicating the presence of an adjacent 2D component.

To facilitate autodiscrimination processing, restrictions are placed on which version of CC-A or CC-B may appear above a GS1 DataBar family component, as follows (see 4.4):

a) GS1 DataBar Stacked or GS1 DataBar Stacked Omnidirectional components may only be linked to two-column CC-A or CC-B components;

b) GS1 DataBar Limited components may only be linked to three-column CC-A or CC-B components;

c) GS1 DataBar Omnidirectional and GS1 DataBar Truncated components may only be linked to four-column CC-A or CC-B components;

d) GS1 DataBar Expanded components may only be linked to four-column CC-A or CC-B components. If linked to a 2D component, a GS1 DataBar Expanded Stacked component shall contain at least four symbol characters within its top row.

CC-C components are never used in a GS1 Composite symbol containing a GS1 DataBar family component.

Figure 6, Figure 7, Figure 8, Figure 9 and Figure 10 illustrate five types of GS1 DataBar component linked to CC-A or CC-B components.

Figure 6 — A GS1 DataBar Stacked Composite symbol (with CC-A), encoding 0103412345678900 and 17010200 respectively

Figure 7 — A GS1 DataBar Limited Composite symbol (with CC-B), encoding 0103512345678907 and 21abcdefghijklmnopqrstuv respectively

Figure 8 — A GS1 DataBar Omnidirectional Composite symbol (with CC-A), encoding 0103612345678904 and 11990102 respectively

Figure 9 — A GS1 DataBar Expanded Composite symbol (with CC-A), encoding 01937123456789043103001234 and 911A2B3C4D5E respectively

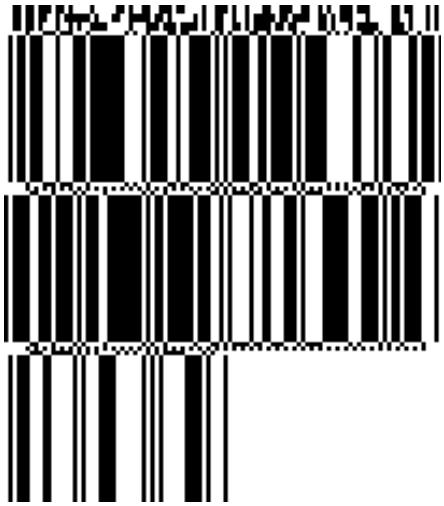


Figure 10 — A GS1 DataBar Expanded Stacked Composite symbol (with CC-A), encoding 0100012345678905 10ABCDEF and 2112345678 respectively

7.4 GS1-128 components

GS1-128 symbols are defined in ISO/IEC 15417 as a subset of Code 128 symbols commencing with the FNC1 character in the first symbol character position after the start character. When used as the linear component of a GS1 Composite symbol, the GS1-128 symbol shall end with a Code Set character encoded in the last symbol character position before the check character. This extra Code Set character serves as a linkage flag, indicating to the decoder the presence of an adjacent 2D component. The choice of Code Set character to be used depends on the code set in use at the end of the linear component's data and whether the 2D component is MicroPDF417-based (CC-A or CC-B) or PDF417-based (CC-C), as shown in Table 8:

Table 8 — Code Set characters to be used for linkage

Active Code Set at end of data	For CC-A or CC-B, use	For CC-C, use
A	Code Set B	Code Set C
B	Code Set C	Code Set A
C	Code Set A	Code Set B

Any of the three 2D components may appear above a GS1-128 component. Only the four-column versions of CC-A or CC-B may be used; CC-C may be any valid number of columns wide.

The choice between CC-A or CC-C is normally made by the user. As an option, this choice may be made implicitly by the encoding software, with the goal of creating the 2D component of the smallest overall height that will fit entirely within the horizontal bounds of the GS1-128 component (including its quiet zones).

Figure 11 and Figure 12 illustrate GS1-128 components linked to CC-A and CC-C components respectively.

Figure 11 — A GS1-128 Composite symbol (with CC-A), encoding 0103212345678906 and 1A1B2C3D4E5F6G7H8 respectively

Figure 12 — A GS1-128 Composite symbol (with CC-C), encoding 00030123456789012340 and 02130123456789093724<FNC1>101234567ABCDEFG respectively

8 CC-A component requirements

8.1 CC-A — General

CC-A is a multi-row symbology component derived from MicroPDF417, for use only in GS1 Composite symbols. CC-A components have two, three, or four data columns, and range in size from three to twelve rows high. Because CC-A components use a Rotation of 32 between adjacent Row Address Patterns, different from any standard MicroPDF417 Rotation, CC-A components are fully compatible with, and can be autodiscriminated from, standard MicroPDF417 symbols. CC-A components are always linked to a linear component.

8.2 Overview of the CC-A component

CC-A has many of the characteristics of MicroPDF417, but differs from MicroPDF417 in several key respects. This clause specifies which aspects of CC-A are identical to MicroPDF417, and then gives an overview of the differences, which are fully defined in subsequent clauses of this specification.

The following characteristics of CC-A are the same as described in ISO/IEC 24728:

a) **Symbol character set**: CC-A uses the same symbol character set as MicroPDF417 (and as PDF417). This symbol character set is arranged in three clusters, alternating from row to row in the same way as in MicroPDF417.

b) **Row Address Patterns**: CC-A utilizes the same two sets of 52 Row Address Patterns (one set for the Left and Right RAPs, and the other set for Center RAPs) as MicroPDF417. These patterns are vertically arranged in the same sequence and are assigned the same RAP numbers as in MicroPDF417.

c) **Error correction**: CC-A utilizes the same Reed-Solomon error correction method as does MicroPDF417, and like MicroPDF417, each defined version has a fixed number of error correction codewords.

The following characteristics of CC-A are different from MicroPDF417:

a) **Rotation**: CC-A uses a Rotation of 32 between any two neighboring RAP patterns on the same row, whereas MicroPDF417 symbols all use rotations of 24 or less.

b) **Linkage**: a CC-A component is always linked to an adjacent linear component, unlike MicroPDF417, which may be printed as stand-alone 2D symbols.

c) **Symbol versions**: CC-A defines a different set of two-, three-, and four-column versions than those defined for MicroPDF417.

d) **Symbol structure**: 3-column CC-A components have the left Row Address Pattern omitted.

e) **High level data encodation**: CC-A treats codewords 0 to 927 as base 928 data codewords; codeword 928 is reserved. CC-A does not use the Text, Byte, and Numeric Compaction modes of MicroPDF417. Instead, CC-A uses the compaction method defined in Clause 5 for translating the user data into a bit stream, which is then encoded into a CC-A component by using a base 928 radix conversion. Unlike

MicroPDF417, the first codeword of a CC-A component is a data codeword, not an ECI Descriptor codeword.

f) **Quiet zones**: No quiet zone is required above the CC-A component, and a separator pattern shall appear below it (see 11.4).

g) **Unsupported MicroPDF417 features**: because CC-A uses base 928 encoding, CC-A does not support any of the MicroPDF417 special features that are invoked in MicroPDF417 by use of codewords above 899. Thus, the following MicroPDF417 features are not supported by CC-A:

   1) Emulation of standard Code 128 symbols, or of Code 128 symbols with FNC1 in the second position;

   2) Macro codeword substitution (for compaction of Format 05 and Format 06 Common Data Syntax messages);

   3) Reader initialization;

   4) Structured Append;

   5) Extended Channel Interpretations (ECIs).

h) **Reference decode algorithm**: CC-A utilizes the MicroPDF417 Reference Decode Algorithm (except that a different table of defined versions is used). However, CC-A adds an algorithm for rejecting false CC-A component fragments that may be found within scans of the associated linear component.

i) **Symbology identifier**: CC-A does not use the PDF417 symbology identifier prefix.

8.3 CC-A component structure

Each CC-A component consists of a stack of vertically-aligned rows (with a minimum of three and a maximum of twelve rows); the allowable numbers of rows are specified separately for each of the two-, three-, and four-column versions.

8.3.1 Row and column combinations

CC-A components shall conform with certain predefined combinations of numbers of rows, columns, and error-correction codewords. These versions are defined in Table 9.

Table 9 — CC-A version characteristics

Number of data columns (c)	Number of rows (r)	Total CWs in data region	Number of EC CWs (k)	% of CWs for EC	Number of CWs for data	Max Capacity, in Bits	Component Width, in X (Note 1)	Component Height, in X (Note 2)
2	5	10	4	40.00 %	6	59	57	10
2	6	12	4	33.33 %	8	78	57	12
2	7	14	5	35.71 %	9	88	57	14
2	8	16	5	31.25 %	11	108	57	16
2	9	18	6	33.33 %	12	118	57	18
2	10	20	6	30.00 %	14	138	57	20
2	12	24	7	29.17 %	17	167	57	24
3	4	12	4	33.33 %	8	78	74	8
3	5	15	5	33.33 %	10	98	74	10
3	6	18	6	33.33 %	12	118	74	12
3	7	21	7	33.33 %	14	138	74	14
3	8	24	7	29.17 %	17	167	74	16
4	3	12	4	33.33 %	8	78	101	6
4	4	16	5	31.25 %	11	108	101	8
4	5	20	6	30.00 %	14	138	101	10
4	6	24	7	29.17 %	17	167	101	12
4	7	28	8	28.57 %	20	197	101	14

CW = Codeword; EC = Error Correction.

NOTE 1 Includes a 1X quiet zone on either side.

NOTE 2 Assumes Y = 2X; does not include separator pattern.

8.3.2 Row parameters

Each row within a CC-A component has the following structure:

a) A leading quiet zone;

b) A Left Row Address Pattern (omitted in three-column versions);

c) One of the following:

   1) For the two-column version, two PDF417 codewords, or

   2) For the three-column version, one PDF417 codeword followed by a Center Row Address Pattern and two more PDF417 codewords, or

   3) For the four-column version, two PDF417 codewords followed by a Center Row Address Pattern and two more PDF417 codewords;

d) A Right Row Address Pattern;

e) A one-module stop bar;

f) A trailing quiet zone.

All CC-A components contain at least two Row Address Pattern columns (separated from each other by two data columns), and every data column is adjacent to one Row Address Pattern column. Because the three- and four-column CC-A versions use two completely disjoint sets of Center and Right RAPs, a scan of a three-column CC-A (which has no Left RAP) can always be distinguished from a partial scan of a four-column CC-A.

8.3.3 Row Address Pattern assignments

The Row Address Pattern numbers assigned to each row of each CC-A 2-column version are defined in Table 10. Those assigned to the 3- and 4-column versions are defined in Table 11.

In every 2-column version, the trailing edge of the second bar in the left RAP always moves to the left or right between the lowest row and the row above it, as shown by the "<" and ">" marks in the graphical representation in Table 10. This transition can be used by imager decoders to estimate the row height. In every 3- and 4-column version, a similar transition occurs in the third bar of the centre or left RAP respectively, as shown by the "<" and ">" marks in the graphical representation in Table 11.

8.3.4 Codeword sequence

A CC-A component may contain up to 28 symbol characters. The first symbol character is placed at the upper left corner of the symbol matrix. The remaining data codewords, followed by the error-correction codewords, are encoded from left to right, top row to bottom. In the three- and four-column versions, the Center Row Address Pattern does not affect the codeword sequence. There are no pad codewords (the encoded bit stream includes pad bits as needed).

8.3.5 High level data encodation

CC-A does not use the Text, Byte, or Numeric Compaction modes of standard MicroPDF417. Instead, the user's source data is compressed into a binary string, and the resulting binary string is then encoded into the CC-A component as a sequence of base 928 codewords (see 8.5). The binary string is created using an Encodation Method tailored for typical supplementary AI element strings (see Clause 5).

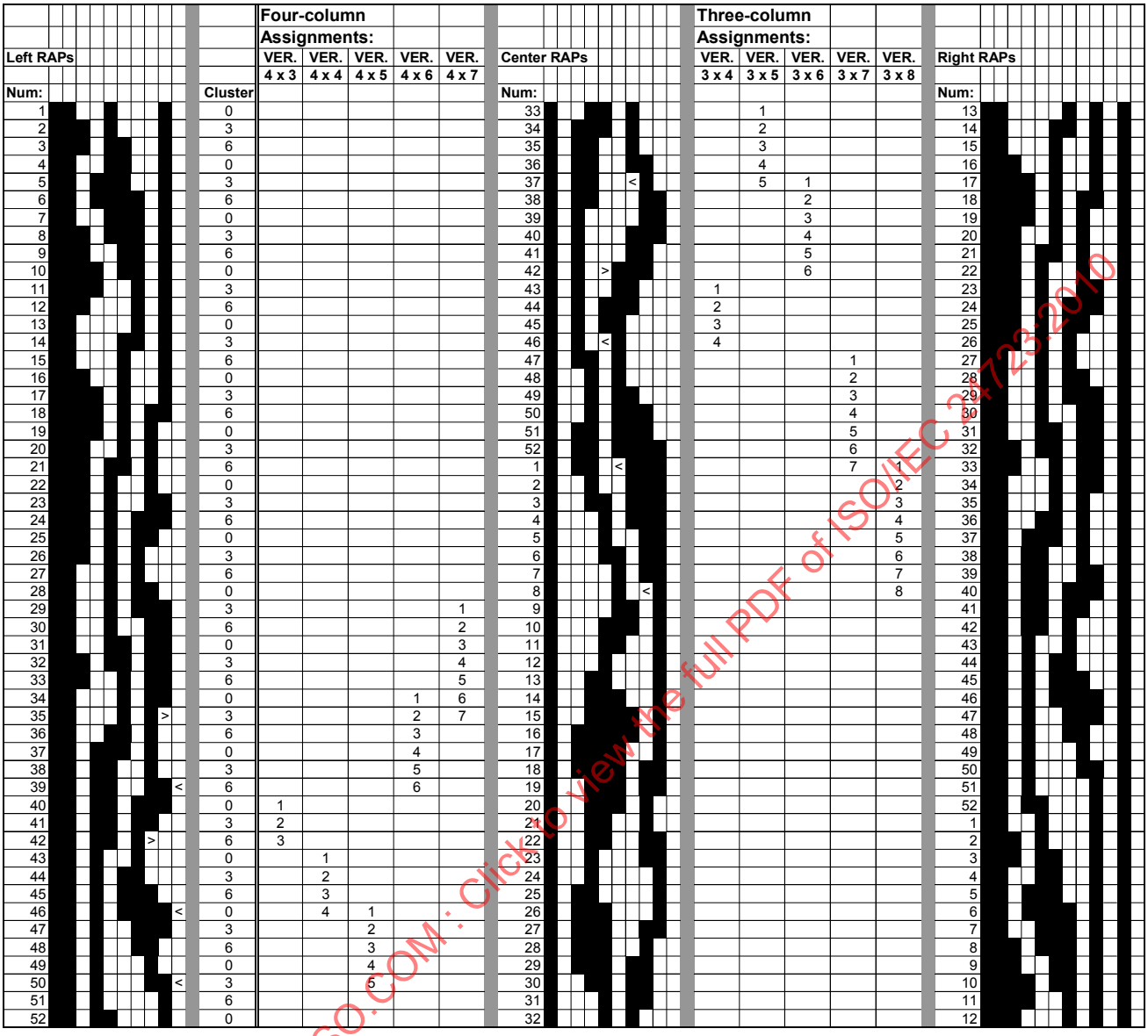
8.4 Symbol character structure

CC-A uses exactly the same symbol characters as MicroPDF417 and PDF417, arranged in the same mutually-exclusive symbol character sets or clusters. The symbol characters are assigned the same symbol character values as in MicroPDF417 and PDF417.

Table 10 — CC-A two-column Row Address Pattern assignments

Left Row Address Patterns		Cluster	VER. 2 x 5	VER. 2 x 6	VER. 2 x 7	VER. 2 x 8	VER. 2 x9	VER. 2 x10	VER. 2 x 12	Right Row Address Patterns
Pattern #										Pattern #
1		0		1						33
2		3		2						34
3		6		3						35
4		0		4						36
5		3		5						37
6	<	6		6						38
7		0								39
8		3				1				40
9		6				2				41
10		0				3				42
11		3				4				43
12		6				5				44
13		0				6				45
14		3				7	1			46
15	>	6				8	2			47
16		0					3			48
17		3					4			49
18		6					5			50
19		0					6			51
20		3					7		1	52
21		6					8		2	1
22	>	0					9		3	2
23		3							4	3
24		6							5	4
25		0							6	5
26		3							7	6
27		6							8	7
28		0							9	8
29		3							10	9
30		6							11	10
31	<	0							12	11
32		3			1					12
33		6			2					13
34		0			3					14
35		3			4					15
36		6			5					16
37		0			6					17
38	>	3			7					18
39		6	1							19
40		0	2							20
41		3	3							21
42		6	4							22
43	>	0	5					1		23
44		3						2		24
45		6						3		25
46		0						4		26
47		3						5		27
48		6						6		28
49		0						7		29
50		3						8		30
51		6						9		31
52	<	0						10		32

Table 11 — CC-A three- and four-column Row Address Pattern assignments



Left RAPs Num:	Cluster	Four-column Assignments: VER. 4 x 3	VER. 4 x 4	VER. 4 x 5	VER. 4 x 6	VER. 4 x 7	Center RAPs Num:	Three-column Assignments: VER. 3 x 4	VER. 3 x 5	VER. 3 x 6	VER. 3 x 7	VER. 3 x 8	Right RAPs Num:
1	0						33		1				13
2	3						34		2				14
3	6						35		3				15
4	0						36		4				16
5	3						37 <		5	1			17
6	6						38			2			18
7	0						39			3			19
8	3						40			4			20
9	6						41			5			21
10	0						42 >			6			22
11	3						43	1					23
12	6						44	2					24
13	0						45	3					25
14	3						46 <	4					26
15	6						47				1		27
16	0						48				2		28
17	3						49				3		29
18	6						50				4		30
19	0						51				5		31
20	3						52				6		32
21	6						1 <				7		33
22	0						2					2	34
23	3						3					3	35
24	6						4					4	36
25	0						5					5	37
26	3						6					6	38
27	6						7					7	39
28	0						8 <					8	40
29	3					1	9						41
30	6					2	10						42
31	0					3	11						43
32	3					4	12						44
33	6					5	13						45
34	0				1	6	14						46
35 >	3				2	7	15						47
36	6				3		16						48
37	0				4		17						49
38	3				5		18						50
39 <	6				6		19						51
40	0	1					20						52
41	3	2					21						1
42 >	6	3					22						2
43	0		1				23						3
44	3		2				24						4
45	6		3				25						5
46 <	0		4	1			26						6
47	3			2			27						7
48	6			3			28						8
49	0			4			29						9
50 <	3			5			30						10
51	6						31						11
52	0						32						12

8.5 Base 928 compaction mode

The source data is compressed into a binary string (see Clause 5) and padded according to the procedure in 5.4.4 to fill exactly the number of bits in the appropriate symbol (see Table 9). The bit string is then encoded into a CC-A component using a base 928 radix conversion. Therefore, all of the CC-A symbol character values from zero through 927 are used for data, and only 928 is reserved.

NOTE CC-B and CC-C components use the Byte Compaction mode encodation of PDF417, and do not use this base 928 encodation.

When encoding the compressed binary string, groups of 69 bits are converted to groups of seven codewords using a base 928 radix conversion. The last group of bits in the compressed binary string (which may contain fewer than 69 bits) is converted to a number of codewords as shown in Table 12.

Table 12 — Base 928 codeword capacities

Remaining bits to encode	Number of codewords needed
9	1
19	2
29	3
39	4
49	5
59	6
69	7

A demonstration program for base 928 radix conversion is shown in Annex E.

8.6 Reference decode algorithm

CC-A utilizes the MicroPDF417 reference decode algorithm as defined in the MicroPDF417 specification, except that the tables of valid symbol versions from this specification, rather than those in the MicroPDF417 specification, are used during the decode process. In addition to the steps outlined in the MicroPDF417 decode algorithm, a CC-A decoder should perform two pieces of additional processing, to take into account the presence of an adjacent linear component, as described in 8.6.1 and 8.6.2. Also, a 2D imager can take advantage of the specific alignments and arrangements of the CC-A component, as described in 8.6.3.

8.6.1 Rejecting false 2D component candidates based on the linear component

Once the linear component has been decoded, the symbol type can be used to reject some false 2D component candidates *a priori*, based on the restricted linear and 2D component combinations defined in Table 1. As two examples:

a) After decoding any linear component, the reader can discard any candidate MicroPDF417 segment that defines a single-column symbol.

b) Any candidate segment that defines the left half of a three-column CC-A or CC-B component can be discarded, unless the linear symbology is EAN-8 or GS1 DataBar Limited.

8.6.2 Rejecting false 2D component candidates within the linear component

CC-A requires additional decoding logic beyond PDF417 or MicroPDF417, to reject false candidates that can appear in a small percentage of the linear components that will be printed adjacent to the CC-A component.

After the linear component has first been decoded, the appropriate MicroPDF417 decode algorithm (which at this point may be either in Phase 1 or Phase 2) is executed over the bar and space measurements of that same scan line, in order to detect false MicroPDF417 fragments that may occasionally be found within the bar space pattern of a linear component. If one or more CC-A candidates survive all of the checks for that stage of the algorithm, then:

a) set its scan occurrence count to zero (if in Phase 2, zero out the votes for each of the candidate's codewords in the symbol matrix), and

b) check to see if the MicroPDF417 algorithm can now proceed (to Phase 2, if it was in Phase 1, or to error correction, if it was in Phase 2).

The longest-candidate rule is suspended when following this procedure, so that false candidates that could result from a partial (rather than complete) scan of the linear component are also detected.

8.6.3 Aids to image-processing software

When scanned with a 2D imager, if the linear component is found first, the 2D component may be found by:

a) Calculating a scan line perpendicular to the bars of the linear component and within the linear component or upper-most stacked row for a stacked GS1 DataBar symbol.

b) Looking at successive parallel scan lines stepped in the direction of the top of the linear component in 1Z steps until one of the following occurs:

   1) A phase 1 decodable scan line of a two, three or four column symbol (as defined in ISO/IEC 24728) is found, then the CC-A/B decoding may proceed from this scan line.

   2) A decodable scan line of a CC-C (as defined in ISO/IEC 15438) is found, then the CC-C decoding may proceed from this scan line.

   3) 16 consecutive scan lines do not decode as the linear component or as steps 1 or 2 above, then the 2D component is missing.

When scanned with a 2D imager, if the 2D component is found first, the linear component may be found by:

a) Calculating a scan line perpendicular to the outside bars of the 2D component and within the bottom-most decoded row of the 2D component.

b) Looking at successive parallel scan lines stepped in the direction of the bottom of the 2D component in 1Z steps until one of the following occurs:

   1) A decodable scan of a valid linear component or GS1 DataBar stacked row occurs, then the linear component decoding can proceed from this scan line.

   2) 16 consecutive scan lines do not decode as a decodable scan line of the 2D component or as a valid linear component, then the linear component is missing.

9 CC-B component requirements

CC-B components follow ISO/IEC 24728, with the following restrictions and exceptions:

a) **Reserved codeword**: the CC-B component shall have codeword 920 in the first symbol character position (in place of the ECI Descriptor). MicroPDF417 features that require any other codeword in the first position (such as 916 and 917 for Macro strings) are not supported.

b) **High level data encodation**: immediately following the first codeword of 920, the second codeword shall be a latch to Byte Compaction mode. The user data, compatible with the GS1 system, shall be encoded into a bit stream using the procedures of Clause 5, and the resulting bit stream shall be encoded into the CC-B component, using Byte Compaction mode.

c) **Structured Append**: Structured Append shall not be used in a CC-B component.

d) **Reader Initialization**: the Reader Initialization codeword 921 shall not appear in a CC-B component.

e) **Quiet zones**: no quiet zone is required above the CC-B component, and a separator pattern shall appear below it (see 11.4).

f) **Reference decode algorithm**: the additional steps specified in 8.6 apply to the reference decode algorithm.

g) **Symbology identifiers**: the symbology identifiers for GS1 Composite symbols are as defined in Annex A, rather than as defined for PDF417.

h) **2D component escape mechanism**: after the first Byte mode codeword, 901 or 924, in the 2D component, if another codeword greater than 899 occurs, the GS1 Composite symbol is logically terminated at that point and no further data is processed. The remaining codewords are encoded and decoded according to the rules specified in Annex C and not according to those specified in the body of this specification.

10 CC-C component requirements

CC-C components follow ISO/IEC 15438, with the following restrictions and exceptions:

a) **Reserved codeword**: the CC-C component shall have codeword 920 in the second symbol character position (immediately following the Symbol Length Descriptor).

b) **High level data encodation**: immediately following the first data codeword of 920, the second data codeword shall be a latch to Byte Compaction mode. The user data, compatible with the GS1 system, shall be encoded into a bit stream using the procedures of Clause 5, and the resulting bit stream shall be encoded into the CC-C component, using Byte Compaction mode.

c) **Structured Append**: Structured Append (also known as “MacroPDF417”) shall not be used in a CC-C component.

d) **Reader Initialization**: the Reader Initialization codeword 921 shall not appear in a CC-C component.

e) **Quiet zones**: No quiet zone is required above the CC-C component, and a separator pattern shall appear below it (see 11.4).

f) **Reference decode algorithm**: the additional step specified in 8.6.1 applies to the reference decode algorithm.

g) **Error correction**: all CC-C components shall meet or exceed the minimum error correction level recommended in the PDF417 specification.

h) **Symbology identifiers**: the symbology identifiers for GS1 Composite symbols are as defined in Annex A, rather than as defined for PDF417.

i) **2D component escape mechanism**: after the first Byte mode codeword, 901 or 924, in the 2D component, if another codeword greater than 899 occurs, the GS1 Composite symbol is logically terminated at that point and no further data is processed. The remaining codewords are encoded and decoded according to the rules specified in Annex C and not according to those specified in the body of this specification.

If a choice of CC-C has been made, then the encoder will normally choose the largest number of CC-C columns (including those for the Start and Stop patterns and Row Indicators) that enable the component to fit within the lateral boundaries of the linear component (including its quiet zones). However, to minimize the height of the 2D component, printers may support user input to create a wider CC-C that exceeds the right bound of the trailing quiet zone of the linear component. In either case, in order to maintain angular tolerance for single-line scanners, very large width to height ratios are avoided, by adding a requirement that the total number of CC-C columns shall be less than four times the number of rows in the component. In practical terms, this restriction only affects low-data-content CC-C components, and only when linked to GS1-128 components longer than an SSCC-18.

11 Symbol dimensions

GS1 Composite symbols shall conform to the following dimensions:

11.1 Minimum width of a module (X)

This should be as defined by the relevant application specification for the selected linear component.

The X dimension shall be constant throughout a given component. The X dimension of the 2D component is nominally the same as that of the linear component within the tolerances required to meet the horizontal positioning restrictions defined in 12.1.

11.2 Linear component height

The minimum height of a linear component is specified in the EAN/UPC specification, in the GS1 specification for GS1-128, and in the GS1 DataBar specification.

11.3 2D component row height (Y)

For a CC-A or CC-B component, the row height shall be a minimum of 2X. For a CC-C component, the row height shall be a minimum of 3X.

11.4 Separator pattern and vertical separator bars

When the GS1 Composite symbol includes a GS1 DataBar family or GS1-128 component, the 2D component shall be printed with a 1X high (minimum) high separator pattern above the linear component.

When the GS1 Composite symbol includes a GS1 DataBar Limited component, in the separator pattern, the leftmost four modules and the rightmost four modules above the GS1 DataBar Limited symbol shall be light. The remaining modules in the separator pattern shall be the complement of the GS1 DataBar Limited symbol below with dark modules above the spaces and light modules above the bars.

When the GS1 Composite symbol includes a GS1-128 component, the separator pattern shall be the complement of the linear symbol elements below with dark modules above the spaces and light modules above the bars. However, the separator pattern shall contain light modules above the GS1-128 component's quiet zones.

When the GS1 Composite symbol includes a GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional or a GS1 DataBar Expanded component, the separator pattern shall be the complement of the linear symbol row below, except for the leftmost four and rightmost four modules and the 13 modules above the finder pattern. For GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional, these are the 13 elements over the finder pattern elements 1, 2 and 3. For GS1 DataBar Expanded, these are the 13 elements over the finder pattern elements 1, 2 and 3 of a version 1 finder pattern and elements 3, 4, and 5 of a version 2 finder pattern. These 13 modules are light over the adjacent finder bars and alternating dark, light, dark, etc. over the adjacent finder spaces. The single dark module that occurs in the 13 modules over GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Stacked or GS1 DataBar Stacked Omnidirectional finder value 3 is shifted one module to the right so that it is over the start of the three module-wide finder bar.

The first four and last four modules of the separator pattern are always light, overriding the complementary pattern.

When the GS1 Composite symbol includes an EAN/UPC component, the 2D component shall be printed with a 6X high light separator pattern (a separator space) above the EAN/UPC component, with two 1X wide vertical separator bars (aligned so as to extend the two outermost guard bars of the EAN/UPC symbol vertically) enclosing the separator space. The vertical separator bars shall be offset outwards by 1X, after 2X of extended height, and inwards again by 1X, after 4X of total extended height; each vertical separator bar thus consists of three 2X high segments. The separator space shall extend to the vertical separator bars.

11.5 Quiet zones

For all linear components, the minimum left and right quiet zones shall be as specified in the governing symbology specifications. For CC-A and CC-B components, the minimum left and right quiet zones shall be 1X. For CC-C components, the minimum left and right quiet zones shall be 2X. No quiet zones are required above or below any GS1 Composite symbol.

12 Graphical requirements

12.1 General

To facilitate rapid image processing of GS1 Composite symbols, several constraints have been placed upon the allowed combinations of linear component and 2D components, and on their relative alignments. The valid combinations are defined, in relation to each linear component, in Clause 7. The relative alignment requirements are defined in this clause. The two components shall either be printed simultaneously, or an alternative mechanism utilized to ensure proper alignment of the two components.

12.2 Vertical alignment requirements

The linear component and the separator pattern shall be aligned as specified. The lower two corners of the 2D component shall not rise more than 3X vertically from their nominal positions and shall not fall below their nominal positions. The relative orientation of the components shall not vary by more than plus or minus one degree from the parallel orientation.

12.3 Horizontal alignment requirements

The lower two corners of the 2D component shall not vary more than 3X horizontally from their nominal positions.

The nominal alignments are defined in the remainder of this paragraph as follows:

a) UPC-A, UPC-E, EAN-8, or EAN-13: The last space module before the CC-A or CC-B Stop bar is aligned with the last bar of the linear component (the main symbol in the case of a symbol including an add-on symbol).

b) GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional: The CC-A or CC-B Stop bar is aligned with the last interior space of the linear component.

c) GS1 DataBar Limited: The CC-A or CC-B Stop bar is aligned with the last interior space of the linear component.

d) GS1 DataBar Stacked: The first bar module of the CC-A or CC-B component is aligned with the first bar of the linear component.

e) GS1 DataBar Expanded: The first bar module of the CC-A or CC-B component is aligned with the first space of the first symbol character of the linear component.

f) GS1-128 components linked to CC-C: The first interior space module of the CC-C component is aligned with the second module the Start character of the linear component.

g) GS1-128 components linked to the right quiet zone of the CC-A or CC-B: the CC-A or CC-B component is aligned with the last space module of one of the rightmost symbol characters of the linear component. To calculate the target Code 128 symbol character position for alignment, number the positions from right to left (0 is the Stop character, 1 is the Check character, etc.), and then

Position = (total number of Code 128 symbol characters – 9) div 2

NOTE linked GS1-128 components with fewer than nine symbol characters (including all symbol characters from Start to Stop inclusive) would not be able to contain a GS1 primary identification data item, and thus these are not supported.

12.4 Human readable interpretation

When printed, the human readable data shall appear below the symbol in a legible font. The GS1 General Specifications provides additional requirements for human readable interpretation.

13 Symbol quality

13.1 Linear component

The linear component of the GS1 Composite symbol shall be evaluated in accordance with the appropriate bar code symbology specification and ISO/IEC 15416, which defines a standardised methodology for measuring and grading linear bar code symbols. All elements in the separator pattern above the linear component should be visually distinguishable. For the purposes of assessing symbol quality the separator patterns are not graded.

13.2 2D component

The 2D component shall be evaluated in accordance with the appropriate bar code symbology specification and the methodology defined in ISO/IEC 15415, which defines a standardised methodology for measuring and grading multi-row symbols with cross-row scanning ability. The appropriate symbology specification for CC-A and CC-B is ISO/IEC 24728. The appropriate symbology specification for CC-C is ISO/IEC 15438.

13.3 Overall composite symbol grade

The quality grade of the GS1 Composite symbol is based on a combination of the grading of the linear component and 2D component. The overall composite symbol grade shall be the lower of the overall symbol grade for the linear component and the overall symbol grade for the 2D component. The overall composite symbol grade as well as the overall symbol grade for the linear component and the overall symbol grade for the 2D component shall be reported.

13.4 Additional pass/fail criteria

ISO/IEC 15415 and ISO/IEC 15416 allow for additional pass/fail criteria to be stipulated by a symbology specification. For GS1 Composite symbols, there are two additional criteria.

a) The GS1 Composite symbol shall be given an overall composite symbol grade of 0 if all of the specified linkage flags are not set correctly for the specific combination of linear and 2D components.

b) The GS1 Composite symbol shall be given an overall composite symbol grade of 0 if the linear component and the 2D component do not constitute a permissible combination (see Table 1).

14 Transmitted data

14.1 General data transmission techniques

The GS1 system requires the use of symbology identifiers as described in ISO/IEC 15424 and summarised in Annex A. Applications that do not use symbology identifiers may not recognize the application identifiers in GS1 Composite symbols or conversely may misinterpret the data from other symbols as application identifier data.

In the default transmission mode, GS1 Composite symbols are transmitted using a symbology identifier prefix of “]e0”. The data from the 2D component immediately follows the data from the linear component. FNC1 characters used as data field separators shall be transmitted as $^{G}_{S}$ (ASCII 29), except that FNC1 encoded at the end of the data is not transmitted.

If the GS1 Composite symbol encodes a symbol separator character and/or a 2D component escape character, the transmission shall be split up into multiple messages. The data following the symbol separator character is transmitted as a separate message with the symbology identifier prefix “]e1”, see 14.5. The data following the 2D component escape mechanism character is transmitted as a separate message with the symbology identifier prefix “]e2” or “]e3”, see 14.6.

The reader shall support an option for linear-only transmission mode, see 14.7.

The reader shall support an option for GS1-128 emulation mode, see 14.8.

GS1 Composite symbols encode data from a subset of ISO/IEC 646, which is a 7-bit character set. Readers typically output 8-bit bytes and the default character set for GS1 Composite symbols is ISO/IEC 8859-1, of which ISO/IEC 646 is a proper subset. As ISO/IEC 646 is a proper subset of a number of character sets, including all the parts of ISO/IEC 8859, ASCII, and other typical computer code tables, transmission of the data should be compatible with most receiving systems. The default code page assignment of ISO/IEC 8859-1 also applies to symbol separator message and escape message transmissions.

14.2 GS1-128 Composite symbols

The data encoded in the GS1-128 component shall be transmitted according to ISO/IEC 15417 except that the symbology identifier in the default transmission mode shall be “]e0”.

If the data in the linear component ends with a predefined fixed-length AI element string, see Annex B, the data from the two messages shall be concatenated as if all the data were encoded in a single GS1-128 symbol. If the GS1-128 symbol ends with a variable length AI element string, an implied FNC1, a $^{G}_{S}$ control character, shall be inserted by the decoder in the data string after the end of the data from the linear component. See Annex B for amplification of this rule.

14.3 GS1 DataBar Composite symbols

14.3.1 GS1 DataBar Expanded component

Transmission of the data encoded in a GS1 Composite symbol with a GS1 DataBar Expanded component shall follow the rules defined in 14.2.

14.3.2 GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Stacked, GS1 DataBar Stacked Omnidirectional and GS1 DataBar Limited components

Transmission of the data encoded in a GS1 Composite symbol with any of these linear components shall follow the rules defined in 14.2, with the exception that the AI element string of the linear component is always AI 01 which is a predefined fixed-length AI element string.

14.4 EAN/UPC Composite symbols

14.4.1 EAN/UPC symbols in general

The data from the two components shall be transmitted as separate messages. The data in the EAN/UPC component should be prefixed with the appropriate “]Em” symbology identifier, and that in the 2D component with symbology identifier prefix “]e0”.

Because in an EAN/UPC Composite symbol there is no linkage flag in the linear component indicating the presence of a 2D component, readers shall support the following two selectable modes of operation. In the

first mode, the reader shall transmit data from an EAN/UPC symbol regardless of whether it is the linear component of an EAN/UPC Composite symbol or a stand-alone symbol. In the second mode, the reader shall transmit data from an EAN/UPC symbol only when a 2D component was also decoded.

14.4.2 EAN/UPC Composite symbols with add-ons

EAN/UPC components including add-on symbols should be considered as two separate symbols. The first symbol is the main data packet, and the second symbol is the 2 or 5 digit add-on. The data from these two symbols should be transmitted separately, each with its own symbology identifier. Provision is, however, made for the option of transmission of the data from both symbols as a single data packet with symbology identifier prefix "]E3". Regardless of which transmission option is enabled, the order of transmission shall always be as follows: the data from the main EAN/UPC symbol is transmitted first, then the data from the 2- or 5-digit add-on symbol, and finally the data from the 2D component (if enabled as described in 14.4.1).

14.5 Symbol separator character

A symbol separator character signals the reader to break the data string at that point into separate transmissions; the reader shall begin the transmission of the data following the symbol separator character with the "]e1" symbology identifier prefix. More than one symbol separator character may be encoded in a symbol as defined in ISO/IEC 15438 and ISO/IEC 24728.

14.6 2D component escape mechanism character

CC-B and CC-C may encode a codeword with value greater than 899. This 2D component escape mechanism character signals the reader to break the data string at that point into separate transmissions; the reader shall begin the transmission of the data following the 2D component escape mechanism character with the "]e2" or "]e3" symbology identifier prefix, see Annex C. The data following a 2D component escape mechanism character is encoded using the rules for standard PDF417 and MicroPDF417.

14.7 Linear-only transmission mode

Normally, a reader decodes both components if the linkage flag of the linear component is set. However, a reader shall also support a mode where only the linear component is decoded and transmitted regardless of the state of the linkage flag. The transmission shall follow the rules in the appropriate linear symbology specification.

14.8 GS1-128 emulation

For backward compatibility with existing GS1 AI applications, the reader shall support a user-selectable mode for GS1-128 emulation. In this mode, the symbology identifier prefix "]em" is not used.

If the linear component is an EAN/UPC symbol, its data shall be prefixed with the appropriate "]Em" symbology identifier (the same as described in 14.4). The data of the 2D component is then transmitted as one or more messages, each not exceeding 48 characters, and each with a symbology identifier prefix "]C1".

If the linear component is a GS1 DataBar or GS1-128 symbol, the data of the entire GS1 Composite symbol is transmitted as one or more messages, each not exceeding 48 characters, and each with a symbology identifier prefix "]C1".

When the data in the "]C1" message would exceed 48 characters, the message shall be split by the reader into multiple messages. The split shall be made at the start of the AI element string that would cause the message to exceed the 48 character limit, thus preserving the integrity of the data. Annex B defines the method to be used.

The symbol separator character and Composite component escape mechanism characters shall not have their normal behaviour, as defined in 14.4 and 14.5, in this mode. Only the portion of the data encoded in the 2D component prior to the first symbol separator or escape mechanism character shall be transmitted, and the remaining data shall be discarded by the reader.

When GS1-128 emulation and linear-only transmission mode are both enabled, the EAN/UPC symbols have a symbology identifier prefix “]Em”; the GS1-128 and GS1 DataBar symbols have a symbology identifier prefix “]C1”.

14.9 Examples of transmitted data

14.9.1 GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Stacked and GS1 DataBar Stacked Omnidirectional Composite symbol

The data string 1197061610A123C0001 (encoding a production date of 97/06/16 and a lot number A123C0001) supplementing a GS1 DataBar Omnidirectional, GS1 DataBar Truncated, GS1 DataBar Stacked or GS1 DataBar Stacked Omnidirectional encoding 120012345678909 (note the linkage flag, the leading 1) would be transmitted as “]e0012001234567890911970616 10A123C0001”.

14.9.2 EAN/UPC Composite symbol

The data string 991234-abcd supplementing an EAN-13 symbol encoding 3312345678903 would be transmitted as two messages “]E03312345678903” and “]e0991234-abcd”.

14.9.3 GS1 Composite symbol with variable length AI field

The data string 02130123456789093724<FNC1>10123 (encoding a contents article number of 13012345678909, a quantity of 24 and a batch number of 123) supplementing a GS1-128 symbol (encoding Serial Shipping Container Code 00030123456789012340 followed by a serial number AI element string 21ABC) would be transmitted as “]e0000301234567890123402 1ABC^{GS}02130123456789093724^{GS}10123”.

14.9.4 EAN/UPC Composite symbol in GS1-128 emulation mode

The data string 10ABC1234 supplementing an UPC-A symbol encoding 0012345678905 would be transmitted as two messages “]E00012345678905” and “]C110ABC1234”.

15 Application-defined parameters

The only application defined for GS1 Composite symbols is specified by GS1 in accordance with the GS1 General Specifications. The specification defines the parameters of data content, X dimension, minimum symbol quality grade, symbol type, and symbol placement.

Annex A
(normative)

Symbology identifiers

A.1 General

ISO/IEC 15424 provides a uniform methodology for reporting the symbology read, options set in the reader and any special features of the symbology encountered. GS1 Composite symbols are transmitted according to the rules of Clause 14. The symbology identifiers and their modifier characters are defined below.

In all cases, the symbology identifier information is not encoded in the GS1 Composite symbol, but should be generated by the reader after decoding, and should be transmitted as a preamble to the data message.

A.2 Default symbology identifiers for GS1 Composite symbols

The data from GS1 Composite symbols shall be prefixed with a symbology identifier of:

]em

where

] is the symbology identifier flag character (ASCII 93);

e is the code character assigned to the GS1 Composite symbology;

m is a modifier character with one of the values defined in Table A.1.

Table A.1 — Symbology identifier modifier values for EAN/UPC components

m	Option
0	Standard data packet.
1	Data packet containing the data following an encoded symbol separator character.
2	Data packet containing the data following an escape mechanism character. The data packet does not support the ECI protocol.
3	Data packet containing the data following an escape mechanism character. The data packet supports the ECI protocol.

NOTE The protocol for “]e2” corresponds to the protocol defined for PDF417 using symbology identifier prefix “]L2”, and the protocol for “]e3” corresponds to the protocol defined for PDF417 using symbology identifier prefix “]L1”.

A.3 Symbology identifiers for EAN/UPC components

When transmitting data from EAN/UPC Composite symbols, two separate transmissions from the reader are required. The data from an EAN/UPC component shall be prefixed with a symbology identifier of:

]Em

where

] is the symbology identifier flag character (ASCII 93);

E is the code character assigned to the EAN/UPC symbology;

m is a modifier character with one of the values defined in Table A.2.

Table A.2 — Symbology identifier modifier values for EAN/UPC components

m	Option
0	Standard data packet in full EAN format, i.e. 13 digits for EAN-13, UPC-A and UPC-E (does not include add-on data).
1	Two digit add-on data only.
2	Five digit add-on data only.
3	Combined data packet comprising 13 digits from EAN-13, UPC-A or UPC-E symbol and 2 or 5 digits from add-on symbol.
4	EAN-8 data packet.

A.4 Symbology identifier for GS1-128 emulation

When GS1-128 emulation option is enabled in the reader, each data packet (except the data from an EAN/UPC component) shall be prefixed with a symbology identifier of:

]C1

where

] is the symbology identifier flag character (ASCII 93);

C is the code character assigned to the Code 128 symbology;

1 is a modifier character indicating a GS1 data packet.