

AIM Inc.

AIM Inc. TSC/04-018
Secretariat: AIM, Inc.

***International
Symbology Specification –***

Ultracode

Version 0.98

**Executive Summary
For Committee Use Only**

Not for Public Distribution



Document type: AIM International Technical Standard

Introduction

The Ultracode symbology is a family of multi-row, variable length two-dimensional matrix symbols incorporating both structural rules for symbol damage detection and Reed-Solomon error control for error detection and data correction. The Ultracode symbol is designed to read with any standard sRGB-compliant scanner, color digital camera or mobile phone.

The Ultracode symbology is designed to be an alternative to conventional linear bar codes when any or all of the following conditions apply:

- ASCII or ISO 8859/x family 8-bit characters are to be encoded in the symbol
- Efficient encoding of URLs is desired
- CMYK printing or direct marking techniques are used for symbol creation
- sRGB displays are used for carrying the symbol
- The use of Extended Channel Interpretation (ECI) is desired
- Space is limited
- Aesthetic considerations apply, such as on consumer packaging

Both the 7-bit ASCII (equivalent to ISO/IEC 646:1991) and the ISO/IEC 8859 family of 8-bit character sets are used in Ultracode symbols.

The Ultracode symbology also fully implements the AIM International Technical Standard *Extended Channel Interpretations: Part 1: Identification Schemes and Protocol*.

Existing 7-bit and 8-bit linear or stacked non-error correcting symbologies may be emulated using Ultracode Bar Code Emulation Mode symbols. This includes acceptance of existing bar code character sets, data structures, Symbology Identifiers, and options. Emulation provides an upgrade path for legacy applications to the use of an error-correcting symbology with minimum impact on the surrounding applications software. To further support use of the Ultracode symbology with legacy systems and applications, a compliant 7-bit transmission mode scanner type is defined for use with 7-bit ASCII data, which requires neither the use of Symbology Identifiers nor the AIM Extended Channel Interpretation (ECI) Protocol.

Ultracode symbols are designed to be printed using any cyan, magenta, yellow and black ink process color printing, inkjet or toner based printing process, or displayed on any sRGB compliant electronic display, such as an LCD screen or a mobile phone display. Restriction of the color set used is to enhance decodability under poor lighting conditions, at some sacrifice of data density.

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

1 Specification scope and structure

1.1 *Scope*

This specification defines the requirements for both the Ultracode symbology. It specifies the complete symbol characteristics including the data character encodation process, rules for error control encoding, the graphical symbol formats, symbol dimensions and print quality requirements, reference decoding algorithms, and user-selectable application parameters.

This specification will first completely explain the symbol structure, graphical encoding, data encoding, error correction, reference decode algorithm, and print quality measurement for the Ultracode graphical encoding method. A reference decode algorithm will be given, and specific print quality considerations for Ultracode will be discussed within the methods of ISO/IEC 15415, which is a standard for monochrome 2-dimensional symbols. Additional considerations for the measurement of color print quality in Ultracode outside the scope of ISO/IEC 15415 will be discussed.

4 Ultracode symbol description

4.1 *Symbology characteristics*

Ultracode symbols are two-dimensional matrix symbols with a data dependent number of rows and a data dependent length. They are designed to be created using CMYK printing processes or displayed using RGB color displays. They may be read using any standard color digital imager, scanner or camera. The Reed-Solomon error control process is used for error detection and correction.

The principal characteristics of the Ultracode symbology are:

1. **Code type:** Variable-size multi-row matrix code. Height and length are dependent on data length and error correction level. The nominal square module width and height are both X.
2. **Encodable character sets:** Includes the family of ISO 8859 8-bit character sets with defaults of ISO/IEC 8859/1 and 7-bit ASCII (equivalent to ISO/IEC 646:1991). Other languages and character sets may be supported through the use of the AIM ECI protocol.
3. **Representation of data:** Utilizes data encodation tiles of 1-wide by 5-high mixed color modules to encode each data codeword and similarly for special pattern tiles to carry structural information. These tiles are enclosed in a black and white frame called the Guard Pattern (see Figure 1). All tiles utilize cyan, magenta, yellow and, green (CMYG) modules for encoding both codeword and special pattern tiles.
4. **Selectable error control levels:** Encoded using Reed-Solomon Error Control in a prime Galois Field GF(283). Six levels of Reed-Solomon error control, including one for error detection only.
5. **Symbol character self-checking:** Strong tile design rules facilitates detection of damaged Data Encodation tiles as erasures, enhancing the power of RSEC(see [D.3.3](#)).
6. **Symbol size:** 15X high by 13X wide minimum, to 33X by 67X maximum including a 1X Quiet Zone all around.
7. **Symbol overhead:** (see [11.2](#)) Six codeword tiles plus 3 graphical pattern tiles
8. **Finder Pattern:** Like QR Code and Aztec Code, it has a unique interior Finder Pattern.
9. **Guard Pattern:** A 1X black Guard Pattern bordering the symbol tile area. In case of white Quiet Zone encroachment it functions as a secondary black quiet zone in addition to the required 1X white Quiet Zone.
10. **Maximum data characters per symbol:** Assuming a 5-row symbol with default error control level EC2 (11% RSEC/data codewords): 247 Message Codewords which may be all allocated as 369 uppercase English alphabet characters; all as 494 digits; or all as 247 non-character bytes. See [11.3](#) for the calculation. (Typical symbols are a mix of alphanumeric characters)
11. **Orientation independence:** Yes.
12. **Color Reversal:** Neither applicable nor supported
13. **Extended Channel Interpretation:** Intrinsic ECI support in conformance with the ANSI/AIM *International Technical Specification: Extended Channel Interpretation (ECI) – Identification Schemes and Protocol*. This mechanism enables identification and use of multiple character sets and other data interpretations or industry-specific requirements to be represented.

4.2 *Summary of additional features*

The following is a summary of additional features in Ultracode symbols:

1. **Color fade resistance:** The colored tiles were designed to permit decoding under the most common printed color fading problem: Loss of the magenta color due to UV breakdown or oxidation of the dye. Also self-compensates for color hue and intensity variation of all colors.
2. **Structured Append:** Allows files of data to be represented logically and contiguously in up to nine Ultracode symbols. Original data can be correctly reconstructed regardless of the order of symbol scanning.
3. **Associated Symbology indication:** Scanner associated data carrier indication flags are provided for linear and non-Ultracode 2-D bar code symbols, RFID tags, CDs, etc., logically linked to the Ultracode symbol.
4. **Function codes:** FNC1 and FNC3 are available for use.

5. **Multilingual encoding:** Native use of the ISO 8859 family of character sets permits the encoding of most natural languages with the Ultracode symbology, including multiple languages within a single Ultracode symbol.
6. **Linear Bar Code Emulation Mode:** Provides for error-correcting Ultracode symbol emulation of the most popular linear or stacked non-error correcting bar code symbologies using 7-bit ASCII (ISO/IEC 646:1991) or ISO/IEC 8859/1 characters, including Code 39, Code 49, Code 93, Code 128, Codabar, and Interleaved 2 of 5.

4.3 Symbol overview

Each Ultracode symbol consists of a 2 to 5 variable but equal length rows each containing a sequence of colored symbol tiles, with a black and white interior finder pattern at the nominal right edge of the symbol. The tile row area is completely enclosed in a 1X black border.

Each tile is comprised of 5 nominally 1X square modules arranged 1X wide by 5X high. Tiles are placed adjacent to each other in column order from top to bottom then left to right. Each tile module is colored nominally cyan, magenta, yellow, or green. There is an additional tile to the left of the interior black line called the symbol descriptor which tells the number and more importantly tile columns on the right side of the symbol. It is used as an aid in decoding, particularly when the right edge of the symbol is missing or damaged. Details of the Ultracode symbol structure and composition are given in the following clauses.

An 2-row Ultracode symbol encoding the ASCII data, "ULTRACODE_123456789!" using default error correction level EC2 (see [6.7.2](#)) is illustrated in Figure 1. The complete details of this symbol's encoding are given in [Annex G.1](#). Addition symbol examples appear in Annex G.

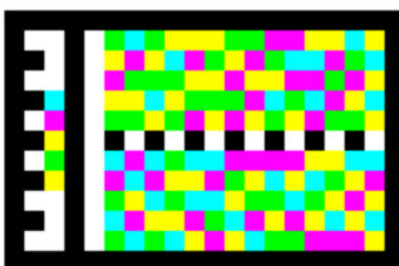


Figure 1 – Ultracode symbol encoding "ULTRACODE_123456789!" using EC2

4.3.1 Symbol tiles

The layout of a Ultracode symbol tile is shown in Figure 2. Each tile has 5 square modules, named z_1 through z_5 , which are of mixed colors nominally cyan(C), magenta(M), yellow(Y) or green(G). The Reference Decode Algorithm in clause 10 is designed to accept a range of hues for each nominal color, and to compensate for magenta fading (see [Annex D.1.4](#)). In the each row of the symbol, modules are printed from top to bottom.

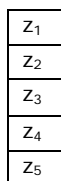


Figure 2 – Ultracode symbol tile structure

The module pattern in each tile is unique and may be described by an 5 octal digit number $z_1z_2...z_5$. Octal values are assigned to cyan(1), magenta(3), yellow(5) and green(6) modules starting with the top module, z_1 . For example, in Figure 1 the GYMYG symbol tile in the upper left corner is denoted 65,326₈ and the CMYCM symbol tile in the lower right corner is denoted 13,513₈.

Denoting the tile names by their octal number representations makes them easier to visualize and remember, since there is a direct bit correspondence between the tile modules colors and the tile's octal number.

The complete set of data encodation tiles used for each codeword or pattern value 0-282 is given in Table D-5 in [Annex D.2.3](#).

Tiles are designed to rules which make them largely self-checking, so that coloring errors can be detected during the tile decoding process, allowing them to be treated as erasures rather than errors in the RSEC process.

4.3.2 Symbol versions

There are 4 versions of the Ultracode symbol, as shown in Table 1, having between 2 and 5 rows of tiles. The length of the symbol also varies the amount of data encoded. The codewords capacity includes both the T data codewords and the ACC codewords available for error correction, depending on the error correction level used.

The design of the versions was to keep the recommended minimum and maximum capacity so that the versions had some small overlap. Also, the recommendations for the printed height:width form factor is kept between 1.2:1 and 1:2 to match mobile phone cameras and displays, and the form factor on common camera CMOS color imagers (3:4 or 9:16). In Table 1, form factors are given in H:1, H:4 and H:16 format for comparison with common display and imager formats. The smallest symbols in each version are approximately square. The largest symbols in each version are approximately 8.5:16 and fit very well on 9:16 HD imagers. However, there is no restriction on printing or scanning the symbol rotated if the image better fits the imager when rotated 90°.

Table 1- Ultracode symbol versions

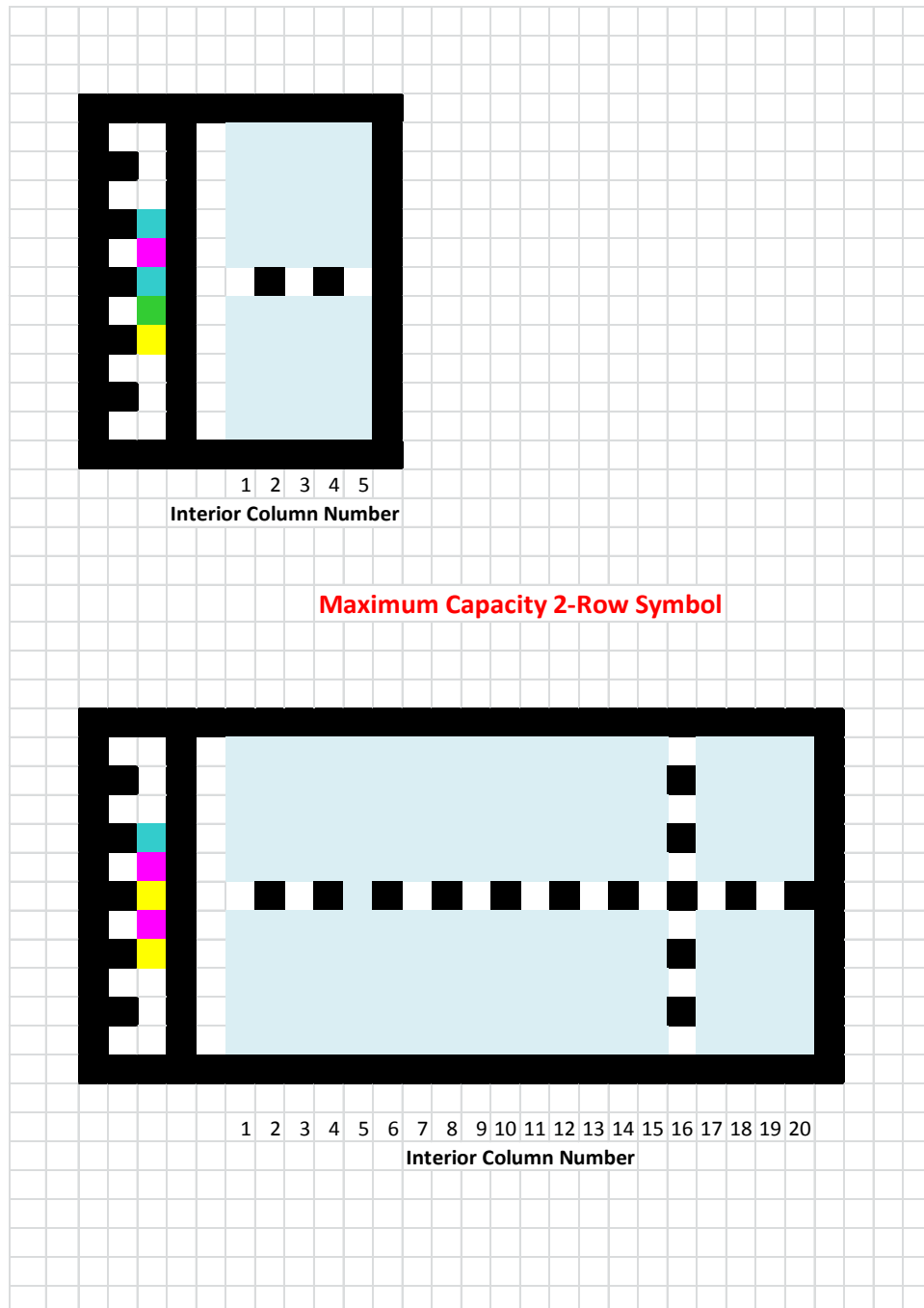
	<u>2 Row</u>		<u>3 Row</u>		<u>4 Row</u>		<u>5 Row</u>	
Recommended:	Min	Max	Min	Max	Min	Max	Min	Max
Height, in X Including 1X QZ	15	15	21	21	27	27	33	33
Width, in X Including 1X QZ	13	28	21	39	31	51	39	68
Form Factor Height: Width (H:1)	1.2:1	0.54:1	1:1	0.54:1	0.87:1	0.53:1	0.85:1	0.49:1
Form Factor Height: Width (H:4)	4.6:4	2.1:4	4:4	2.2:4	3.5:4	2.1:4	3.4:4	1.9:4
Form Factor Height: Width (H:16)	18.5:16	8.6:16	16:16	8.6:16	13.9:16	8.5:16	13.5:16	7.8:16
# Interior Columns	5	20	13	31	23	43	31	60
# Secondary Vertical Clock Tracks	0	0	0	2	1	2	2	3
Available Tile positions	10	40	39	87	88	164	145	285
Available (T+ACC) Codewords	1	31	30	78	79	155	136	276
Symbol Descriptor Base Number	0	0	61	61	127	127	191	191
Symbol Descriptor Tile Value	5	20	74	92	150	170	222	251

For example, if there were T=26 data codewords to be encoded at error correction level EC2, then there would be 9 total RSEC codewords with ACC =6 available for error correction. The total number of (T + ACC) codewords would be 32. This would require a 3 row symbol with 14 interior columns and 1 pad codeword. The size, including a 1X Quiet Zone would 21X height by 22X wide.

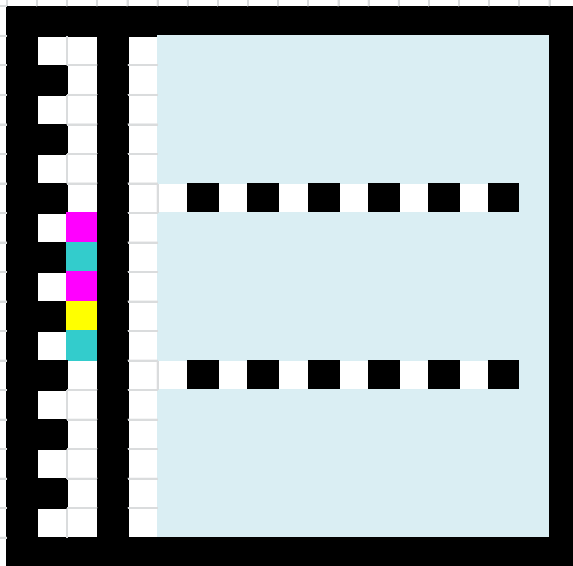
Figure 3 shows the layout of the recommended minimum and maximum capacity symbols for all 4 versions. Note that the 4- and 5-row versions are shown at a smaller scale than the 2- and 3-row versions in order to fit on the page. The light blue areas in each symbol are placeholders for data encodation tiles. Both horizontal and vertical black and white clock tracks are shown, including s secondary vertical clock track every 16 interior columns.

A single pattern tile called the symbol descriptor tile is encoded at the center of the white bar next to the vertical clock track. See Figures 3. It consists of a symbol descriptor base number from Table 1, which depends on the number of tile rows in the symbol to which is added the total number of interior columns, and uniquely identifies the symbol row and column count.

Figure 3 – 2-, 3-, 4- and 5-row minimum and maximum capacity Ultracode symbols

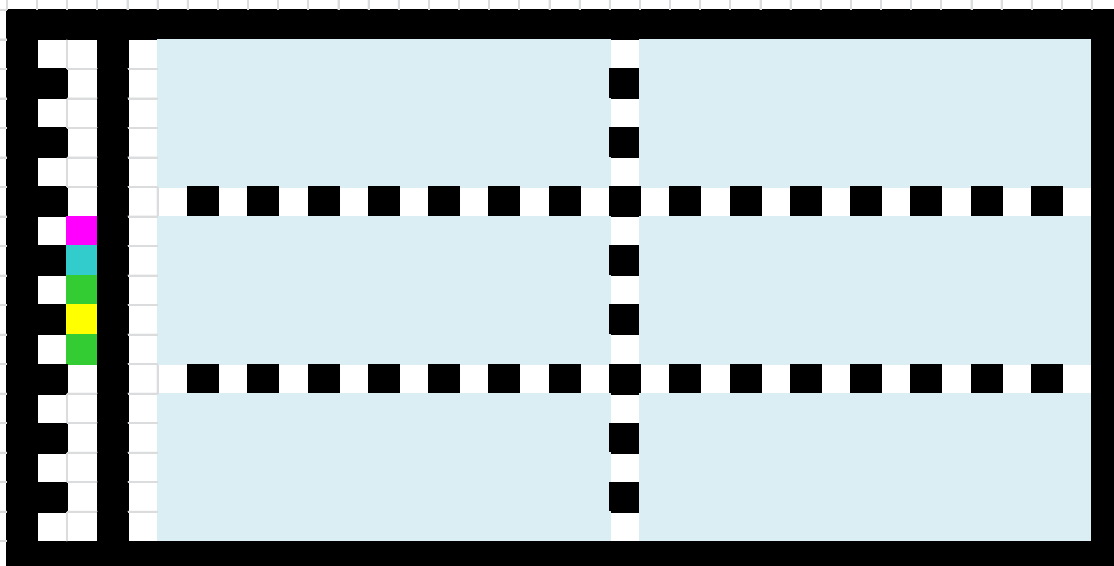


Minimum Capacity 3-Row Symbol



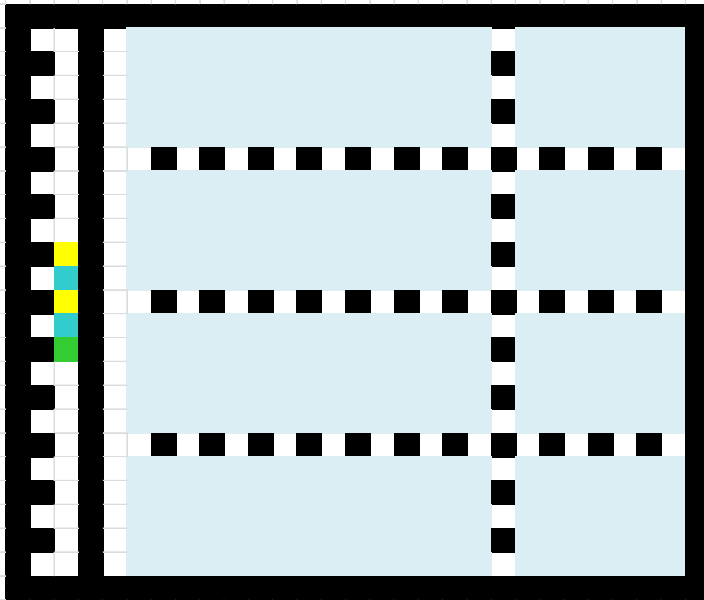
1 2 3 4 5 6 7 8 9 10 11 12 13
Interior Column Number

Maximum Capacity 3-Row Symbol



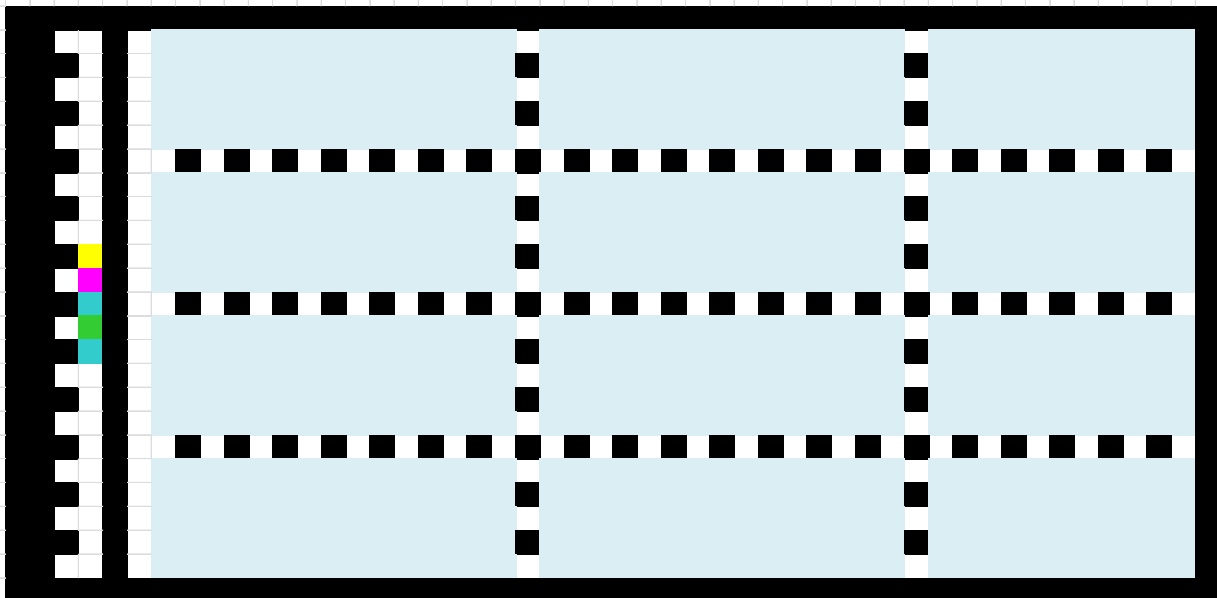
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
Interior Column Number

Minimum Capacity 4-Row Symbol

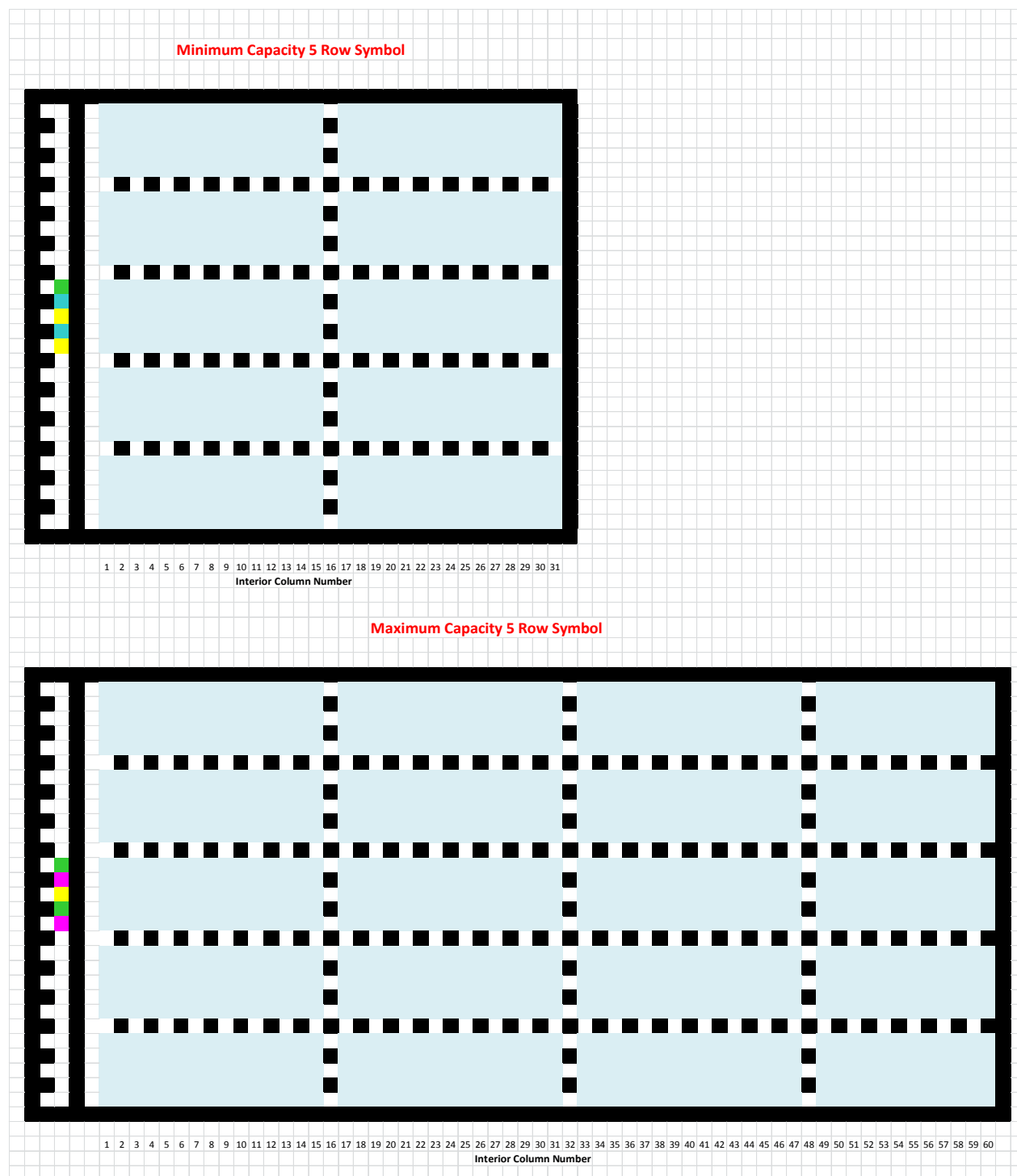


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
Interior Column Number

Maximum Capacity 4-Row Symbol



1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43
Interior Column Number



4.3.3 Quiet Zone

The Quiet Zones is the area of white-colored modules that surround the symbol. The presence of a Quiet Zone enables the symbol to be more easily discriminated from its environment by the reader. There is a mandatory 1X minimum white Quiet Zone around the symbol. Generally, a larger Quiet Zone enables easier detection of the presence of an Ultracode symbol. Neighboring Ultracode symbols may share any common Quiet Zone.

4.3.4 Guard Pattern

The Guard Pattern (see Figure 4) is a fundamental symbol feature used in the symbol scanning or imaging process. It is a black 4-side frame forming an box which encloses the color modules and the Finder Pattern. The Guard Pattern both defines the physical size and orientation of the symbol and, if incomplete, The Guard Pattern is one module tall or wide in thickness. The Guard Pattern also functions as a 1X black quiet zone, so that in the case of encroachment on the white exterior Quiet Zone the symbol can still be clearly recognized.

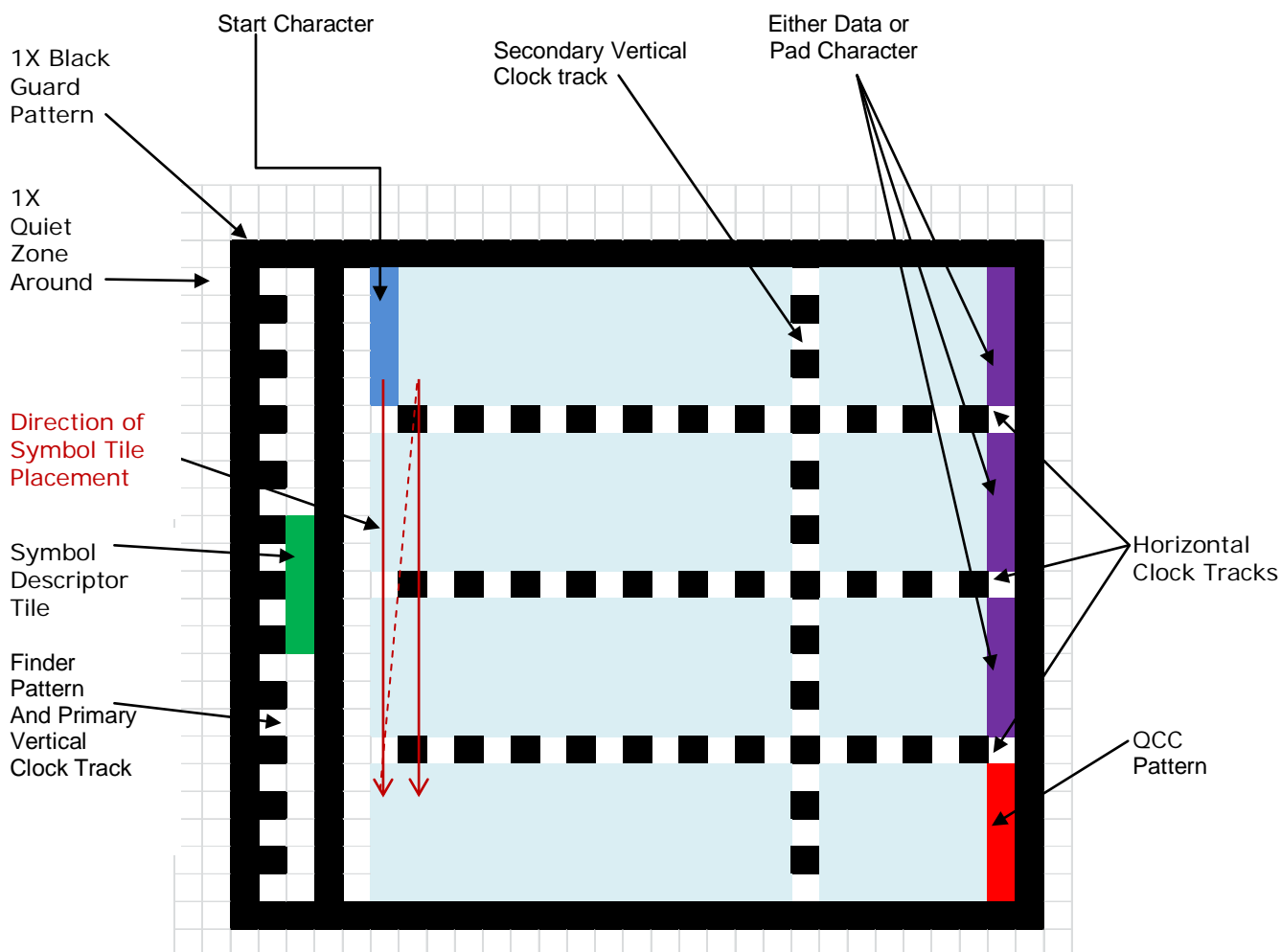


Figure 4 – 4-row Ultracode symbol structure example

4.3.5 Finder Patterns

The purpose of the black/white interior Finder Pattern is to assist in identifying the symbol, and trigger a search for the module centers. The Finder Pattern is the last 5 columns in the symbol (see Figure 4). It consists of a white column, black column, white column, and a vertical clock track abutting the black column formed by the Guard Pattern.

4.3.6 Clock tracks

There is a primary vertical black/white clock track as part of the Finder Pattern. Secondary vertical clock tracks are used at interior column numbers (see Figure 4) 16, 32 and 48 when these interior columns are present. These are very useful for finding column centers in large or distorted symbols.

Horizontal clock tracks are used as row separators and extend to the Finder Pattern, always with a white module in interior column 1. This ensures there will always be a black module in interior columns 16, 32 and 48 to enable forming of a secondary vertical clock track in these locations.

4.3.7 Symbol decriptor tile

A single pattern tile called the symbol descriptor tile is encoded at the center of the white bar next to the vertical clock track. Its value is formed from symbol descriptor base number from Table 1 dependent on the number of tile rows in the symbol to which is added the total number of interior columns. Since it uniquely identifies the symbol row and column count, it may be used as an aid in decoding when the right edge of the symbol is missing or damaged.

4.4 Ultracode symbol construction

4.4.1 Symbol Tile Sequence

The symbols Symbol Tile Sequence is comprised of the Message Codewords (see 3.1.2), RSEC Codewords, and structural codewords (see 3.1.5) and patterns in the order shown in Table 2.

Table 2 – Ultracode symbol tile sequence

Ultracode Symbol Region Description	Tile Type	Encoded Value(s)	Reference Clauses
Start Character	Codeword	Start Codeword	4.5.1
MCC Character	Codeword	MCC	4.5.2
RSEC Region Codewords	Codeword	RSEC Codewords	4.5.3
TCC Pattern	Pattern	TCC	4.5.4
RSEC/Data Separator Pattern	Pattern	None	4.5.5
ACC Character	Codeword	ACC	4.5.6
Symbol Control Region (if present)	Codeword	Structural Info	4.5.7
Data Region Codewords	Codeword	Data Codewords	4.5.8
Pad Characters (if present)	Pattern	None	4.5.9
QCC Pattern	Pattern	QCC	4.5.10

4.4.2 Symbol encoding

The symbol is encoded following the ordering in Table 2. One 5-module data encodation tile (see 4.4.3 and Annex D.3) is used for each codeword or pattern. Tiling proceeds column by column, starting with the tile position in the top row and proceeding through the bottom row as in Figure 4.

As an example, the layout of the tiles in the Ultracode symbol of Figure 1 "ULTRACODE_123456789!" is shown in Figure 5. Note that this example has no Symbol Control Region. Note also that tile names are written vertically in each tile position in Figure 5.

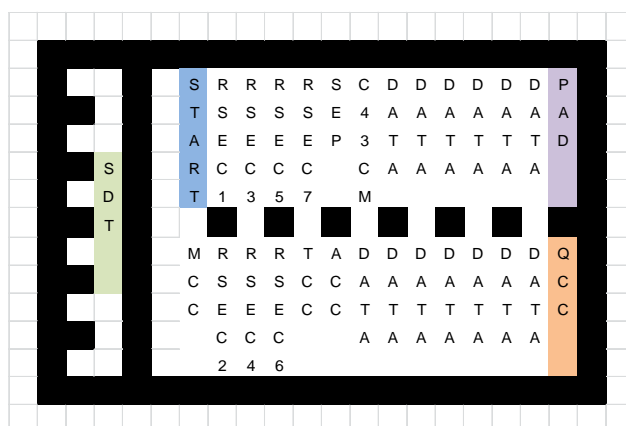


Figure 5 – Layout of data in the Ultracode symbol in Figure 1.

Complete details of construction of the Ultracode symbol shown in Figures 1 and 5 are given in [Annex G.1](#), along with the tile codeword values and special pattern values.

The special considerations and rules for construction of each symbol component or region will be defined in detail in the clauses referenced in Table 2. The development of the codeword sequence for Ultracode symbols is defined in clauses 6, and 7.

4.4.3 Data encoding tiles

Specific selection rules define the allowed data encoding tiles that encode symbol characters or special graphical patterns. [Annex D.3](#) describes the design of all the 283 data encoding tiles used in Ultracode symbol construction. These encode both Message Codewords and RSEC Region codewords, as well as data values carried by the TCC and QCC patterns; one value per tile.

4.5 Ultracode symbol components

4.5.1 Start Character

The Start Character is the colored tile adjacent to the top left Guard Pattern corner. It is always contains a non-zero start codeword based on the *symbol type* and the data to be encoded, and is specified in Table 8 in [6.2](#) for codeword value, interpretation and Start Sequence details for each Start Character. The top module is always green.

4.5.2 Message codeword count (MCC)

The total number of codewords, C , in the Message Codeword Sequence (see [3.1.2](#)) is encoded as the second Message Codeword using an RGB data encoding tile from Table D-5 in [Annex D-3.2](#).

4.5.3 Reed-Solomon error control (RSEC) Region

The RSEC Region uses the Data Encoding tiles of Table D-5 in [Annex D-3.2](#) to represent the RSEC Codewords. The number of RSEC Codewords, Q , is always calculated as a function of the Message Codeword length, C , and the desired minimum percentage of error control codewords (EC0 through EC5) using the algorithm in [6.7.2](#). Note that $Q = ACC + 3$, where the ACC is the available error-correction codeword count defined in 4.5.6.

4.5.4 Total codeword count (TCC) pattern

The total number of Symbol Codewords (Message Codewords, C , plus RSEC codewords, Q) is encoded as a pattern using a Data Encoding tile from Table D-5 in [Annex D-3.2](#). The purpose of encoding it in the symbol is to help determine the number of missing data encoding tiles during decoding of heavily damaged symbols (see [10.2](#) and Table 27). The TCC and ACC (see 4.5.6) surround the RSEC/Data Separator Pattern, and thus are easy to find during decoding. The MCC (see 4.5.2) can then be recovered (if damaged in the symbol) using the relationship:

$$TCC = MCC + ACC + 3$$

Since both the MCC and ACC are error-corrected Message Codewords, the value of the TCC can always be reconstructed. Therefore, the TCC value is encoded as a graphical pattern and not an error-corrected codeword.

4.5.5 RSEC/Data Region Separator Pattern

The purpose of this Separator Pattern is to allow scanners to determine the start of the Data Region and the end of the RSEC Region, aiding in recovering the data from heavily damaged symbols. It also signals the position of the TCC Pattern, which appears immediately to its left. The pattern, shown in Figure 6 is unique in that it has 3Y modules (rows 1, 3 and 5) and 2 green modules (rows 2 and 4). It has the octal representation 56,56₈ (See [Annex D.2.2](#)).

Module	Color
Z ₁	Yellow
Z ₂	Green
Z ₃	Yellow
Z ₄	Green
Z ₅	Yellow

Figure 6. RSEC/Data Region Separator Patterns and their octal values

The RSEC/Data Region Separator Pattern tile is inserted after the TCC pattern and before the ACC codeword tile. It enables quickly finding the TCC and ACC during decoding. The presence of this Separator Pattern is especially useful in alternate decoding methods when one or both ends of the message are missing.

4.5.6 Available error-correction codeword count (ACC)

The total number of RSEC codewords available for error-correction is (Q-P).

4.5.6.1 ACC codeword encodation format

The ACC codeword is encoded with a second piece of information, the Link1 value, which determines if the optional Symbol Control Region (SCR; see 4.5.7 and [6.4](#)) is present following the ACC, and in what form. See Table 3.

Table3. Encoding of the ACC codeword

ACC codeword range	Link1 parameter	ACC value
0 to 158	0 to 1	0 to 74

pattern

Encoding: ACC Codeword = $78 * \text{Link1} + (\text{ACC value})$

Decoding: Link1 value = $(\text{ACC Codeword}) \div 78$
ACC value = $(\text{ACC Codeword}) \bmod 78$

4.5.6.2 Link1 parameter

The Link1 parameter has values 0 or 1. Interpretation is as follows:

Link1 = 0 means no Symbol Control Region is present
Link1 = 1 means that the SCR Type codeword is present

4.5.7 Symbol Control Region

In some Ultracode symbols, additional codewords are used to encode special information about the symbol. There may be up to 4 optional parameter codewords in the Symbol Control Region, depending on the optional features present. This optional sequence, as defined in [6.4](#), follows the ACC Data Encodation tile and is converted to symbol character tiles using Table D-5 in [Annex D-3.2](#).

4.5.8 Data Region

Depending on the Start Character used, one additional Start Sequence codeword may be required (see Table 8 and [6.2.1](#)). This additional Start Sequence codeword shall be placed in the Data Region (see Figure 4) preceding the first data codeword. Data Region codewords are successively encoded into symbol characters using the Data Encodation tiles in Table D-5 in [Annex D-3.2](#).

4.5.9 Pad characters

The QCC is always the bottom tile in the first interior data column. Any pad characters required in the symbol are placed between it and the end of the Data Region tiles to preserve the position of the QCC. This include any additional pad characters require to make a minimum size symbol.

Pad characters are patterns and do not encode a codeword value. The pattern, shown in Figure 6 is unique in that it has 3Y modules (rows 1, 3 and 5) and 2 green modules (rows 2 and 4). It has the octal representation 51,515₈ (See [Annex D.2.2](#)).

Module	Color
Z ₁	Yellow
Z ₂	Green
Z ₃	Yellow
Z ₄	Green
Z ₅	Yellow

Figure 7 – Pad Character patterns

4.5.10 Redundant error-correction codeword count (QCC) pattern

A frequent form of symbol damage is the loss of part of the symbol. However, to reconstruct the symbol, the exact number of RSEC Codewords near the left end of the symbol must always be known. Therefore, the total RSEC codeword count, Q, is encoded redundantly as a pattern, using a single data encodation tile from in Table D-5 in [Annex D-3.2](#).

4.6 Ultracode symbol construction

The process by which the Symbol Tile Sequence is calculated and placed in the symbol and then converted into symbol character tiles takes place in the following manner:

1. Examine the input data and desired error correction level to determine the symbol version and the Start Character codeword per [6.2](#)
2. Determine if the Symbol Control Region is needed and, if so, its size and codewords using [6.4](#)
3. Encode the data into Message codewords using [6.5](#)
4. From the specified EC level, using [6.7.2](#), calculate the number of RSEC codewords, Q and the ACC
5. Form the Message Codeword Sequence as defined in [3.1.2](#) and calculate the MCC codeword
6. Using [6.7.4](#) and the methods of [Annex B.3](#), calculate the Q RSEC codeword values.
7. Determine the TCC and QCC pattern values
8. Using Table 2 in [4.4.1](#) above, form the Symbol Tile Sequence of all the codeword and pattern values, and determine the number of interior columns required.
9. Place the Start Character tile at the top of interior column 1, and continue filling interior columns from top to bottom, left to right.
10. If the interior column number is a multiple of 16, then insert a secondary vertical clock track at that interior column position and continue filling in data encodation tiles to the right of it until the entire message has been tiled.
11. Place the QCC pattern in bottom row of the last interior column (adding an extra column if necessary), and fill any empty tile positions above it to the last data codeword tile with pad characters.
12. Add the Finder Pattern, Guard Pattern, and Quiet Zone.

5 Code characters

The Ultracode symbology fundamentally encodes the 7-bit ASCII (equivalent to ISO/IEC 646:1991) and the ISO/IEC 8859/1-16 character sets.

- Eight-bit symbol types support a wide range of 8-bit character sets for all Latin and some non-Latin languages.
- Seven-bit symbol types support 7-bit ASCII.
- Through use of the AIM Extended Channel Interpretation Protocol byte/multi-byte encoding may be used for ideographic languages such as Chinese, Japanese, and Korean.
- Non-language byte data may be directly encoded
- There are mechanisms for efficient encoding of URLs

5.1 *8-bit character set encoding*

An 8-bit character set consists of a collection of 1 to 256 relevant characters. Typically, this collection encodes a specific native language, or a particular type of information. An 8-bit character set is just a way of collecting up to 256 language- or application-relevant characters. Unused character positions are allowed.

5.2 *Formal 8-bit language character sets and their ECIs*

Internal support for specific 8-bit character sets within the Ultracode symbology consists of supporting the generation and storage of their ECIs within the symbol (see [6.6](#)) and the transmission of the stored Character Set ECI upon symbol scanning (see [12.4](#)). All 8-bit character sets are otherwise treated identically in terms of their direct conversion of 8-bit values to codewords. See [6.5.3](#).

6 Ultracode Symbol Encoding

This clause describes the generation of the complete Message and RSEC Codeword Sequences within Ultracode symbols. These components of the Symbol Tile Sequence are given in [4.4.1](#).

Ultracode symbols are divided into two groups. Each group is optimized for efficient encoding of different types of data structures.

- Language-specific symbols encode a wide range of 8-bit character sets
- The 7-bit ASCII (ISO/IEC 646:1991 IRV) character set

Table 4 – Encoding modes and features for each Ultracode symbol type


Symbol Type	Symbol Characteristics
ISO/IEC 8859/x Symbols	These symbols using character sets ISO/IEC 8859/1 through ISO/IEC 8859/16 throughout for 8-bit characters. Multiple ASCII Submodes may be employed for efficient data compaction of the data. See 6.5.3 for details.
ECI-selected 8-Bit Character Set Symbols	These symbols utilize the AIM ECI Protocol to select other 8-bit character sets; byte/multi-byte Oriental languages such as JIS-X-208 (1997) Annex I "Shift-JIS" Japanese and CNS11643-1992 Chinese and non-language byte data. This may also be used for encoding non-language byte data. See 6.5.3 for details.
8-Bit Bar Code Emulation Mode Symbols	Symbols contain 1 or more emulations of other linear or stacked bar code symbologies using ISO/IEC 8859/1 characters. An optional unique Emulated Symbology Identifier string allows the Ultracode reader output data to "masquerade" as if the data was read from actual bar codes of the emulated symbology. See 7.6 for details.
7-Bit ASCII data Symbols	This symbol uses 7-bit ASCII Mode throughout. Multiple submodes are used for efficient data compaction of the 7-bit ASCII character set. See 6.5.2 for details.
7-bit Bar Code Emulation Mode Symbols	Symbols contain 1 or more emulations of other linear or stacked bar code symbologies using 7-bit ASCII Mode. An optional Emulated Symbology Identifier sequence allows the Ultracode reader output data to "masquerade" as if the data was read from actual bar codes of the emulated symbology. See 7.5 for details.
Non-language bytes data	Byte data is encoded throughout. ECI \000899 is implied. See 6.5.3.4

6.2.3 Start Characters codewords for each symbol type

Table 5 specifies the start character codewords for each symbol type in Table 4, together with their default or assigned character set and AIM-assigned ECI value. These are encoded in the Ultracode symbol per 4.5.4.

Reserved codeword [280] and [282] are not defined. Symbols starting with these values as Start Character codewords are not decodable.

Table 5 – Assignment of Start Sequence parameters for each symbol type

Data type and any special consideration	Character set	Start Character start codeword	Additional Start Sequence parameter codewords	Default ECI	See:
8-bit ISO/IEC 8859/x	8859/1 Latin-1 W. European 8859/2 Latin 2 C. European 8859/3 Latin-3 S. European 8859/4 Latin-4 N. European 8859/5 Latin/Cyrillic 8859/6 Latin/Arabic 8859/7 Latin/Greek 8859/8 Latin/Hebrew 8859/9 Latin-5 Turkish 8859/10 Latin-6 Nordic 8859/11 Latin/Thai 8859/13 Latin-7/Baltic Rim 8859/14 Latin-8 Celtic 8859/15 Latin-9 8859/16 Latin-10 SE European	[257] [258] [259] [260] [261] [262] [263] [264] [265] [266] [267] [268] [269] [270] [271]	None	\000003 \000004 \000005 \000006 \000007 \000008 \000009 \000010 \000011 \000012 \000013 \000015 \000016 \000017 \000018	6.5.3
7-bit ASCII characters	ISO/IEC 646:1991	Codeword [272]	None	\000027	6.5.2
7-bit ASCII GS1 start with FNC1	ISO/IEC 646:1991	Codeword [273]	None	\000027	6.5.2.7
7-bit Bar Code Emulation Mode symbol	ISO/IEC 646:1991	Codeword [274]	Two; Emulated Symbology Identifier and option value	\000027	7.4.1
8-bit characters sets with defined ECIs, including Asian byte/multi-byte languages or byte data Range of \h is \000003 to \000898 corresponding to the 8-bit char sets.	Per AIM document: "Extended Channel Interpretations: The ECI Character Set Registry"	Codeword [275+y] Start:  y: [275] 0 [276] 1 [277] 2 [278] 3	Required ECI \h parameters x, y: If $898 \geq h \geq 4$ then: (x,y) are: $x = h \bmod 256$ $y = h \div 256$ and [x] is the Start Sequence Parameter	Must be encoded in symbol. Input \h; range is: \000004 to \000898	6.5.3.1
8-bit ISO/IEC 8859/1, Bar Code Emulation Mode symbol	ISO/IEC 8859/1	Codeword [279]	Two; Emulated Symbology Identifier and option value	\000003	7.4.2
Non-language byte data	(None)	[280]	None	\000899	6.5.3.4
<i>Reserved</i>		[281],[282]			

6.5 Data Region encodation

There are 2 *base encoding modes* (7-bit ASCII Mode and 8-bit Mode), some of which have (or act themselves as) compaction submodes. These are detailed in the remainder of this clause 6.5.

6.5.1 Encoding mode considerations

There are rules for switching between base encoding modes and in and out of submodes:

- Note that 8-bit Mode uses ASCII Submode and/or C43 Compaction Submode as compaction submodes, for efficient encoding of things such as numeric strings and common alphanumerics
- Note that Bar Code Emulation Mode uses either 7-bit ASCII Mode or 8-bit Mode as a base encoding mode, and is fully described in [7](#).
- 7-bit ASCII Mode may not switch to 8-bit Mode

Symbol data is encoded in the order of reading of the input data characters. This is independent of the direction in which the human readable interpretation is printed. A specific example of a Hebrew Ultracode symbol using 8-bit encoding appears in [Annex G.2](#).

Often several different methods are available for encoding the same string of characters. Guidelines for the use of 7-bit ASCII Submode and C43 Compaction Submode to minimize symbol length are given in [Annex E](#). Some encoding examples appear in [Annex G](#).

6.5.1.2 Encoding mode reserved codewords

Reserved codewords in encoding Tables 10 through 14 corresponding to the various encoding modes have no action in either the symbol encoding or decoding processes. They should never be encoded in Ultracode symbols.

If encountered in the decoding process other than in the RSEC codewords, reserved codewords are to be treated as nulls, i.e., act as if they did not exist at all in the Data Region codeword string.

6.5.2 7-bit ASCII Mode and ASCII Submode

Much of the data used in linear bar codes—especially numerics and uppercase alphabets-- can be encoded in 7-bit ASCII, as defined in ISO/IEC 646:1991 IRV. Since 7-bit ASCII comprises the first 128 characters of both ISO/IEC 8859/1, ASCII Submode is used throughout 7-bit symbols.

ASCII Submode is also very useful in 8-bit symbols for encoding numeric data at 2 digits per codeword. C43 Compaction Submode also uses characters from 7-bit ASCII.

In Table 7, the character values 0 through 127 are encoded directly as codewords of the same value. The remaining codeword values [128] through [282] are available for command codewords and submodes compacting different types of 7-bit data into a reduced number of codewords. A method of selecting the submode sequence in order to minimize the total data codeword length, T, is given in Annex E.

Table 7 – Codeword assignments for 7-bit ASCII Mode/Submode

Codeword values	Meaning or interpretation	Refer to:
0 - 127	7-bit ASCII character values	6.5.2
128 - 227	Double digit numerics ; 00 to 99	6.5.2.2
228 - 237	Single digit followed by selected decimal point character (denoted by '*'): 0*, 1*, 2*, 3*, 4*, 5*, 6*, 7*, 8*, 9*	6.5.2.2
238 - 247	Selected decimal point character (denoted by '*') followed by single digit: *0, *1, *2, *3, *4, *5, *6, *7, *8, *9	6.5.2.2
248 - 258	Single digit or decimal point ('*') followed by field delimiter (denoted by '¶'): 0¶, 1¶, 2¶, 3¶, 4¶, 5¶, 6¶, 7¶, 8¶, 9¶, *¶	6.5.2.3
259 - 269	Field delimiter (denoted by '¶') followed by single digit or decimal point ('*'): ¶0, ¶1, ¶2, ¶3, ¶4, ¶5, ¶6, ¶7, ¶8, ¶9, ¶*	6.5.2.3
270	In 7-bit Bar Code Emulation Mode symbols only: Start new bar code emulation , DCI and option value follow (<i>invalid in non-emulation mode symbols</i>)	7.5
271	FNC3 , reader programming	6.5.2.7
272	FNC1 , use per Annex C	6.5.2.7
273	Encode 06 Macro Sequence	6.5.2.6
274	Temporarily latch to C43 Compaction Submode C1 for 6 C43 characters and return	6.5.2.4
275	Temporarily latch to C43 Compaction Submode C1 for 9 C43 characters and return	6.5.2.4
276	Temporarily latch to C43 Compaction Submode C1 for 12 C43 characters and return	6.5.2.4
277	Temporarily latch to C43 Compaction Submode C1 for 15 C43 characters and return	6.5.2.4
278	Latch to C43 Compaction Submode C1 for multiple groups of 3 characters	6.5.2.4
279	Select delimiter character ; C43 delimiter character value follows (starting default is '%')	6.5.2.3
280	Latch to C43 Compaction Submode C2 for multiple groups of 3 characters (especially useful for URLs)	6.5.2.8
281	Insert a single 8-bit character with value > 128 in the codeword string without leaving ASCII Submode (lower 7 bits of the 8-bit character value follows)	6.5.2.5
282	In 7-bit symbols: Unlatch C43 Compaction Submode In 8-bit symbols: <ul style="list-style-type: none"> If in C43 Compaction Submode, exit C43 submode to 8-bit Mode If in ASCII Submode, exit submode and return to 8-bit Mode 	6.5.2.8

6.5.2.2 Decimal double-density encoding

In Table 7, the selected decimal point is denoted by "*". The starting default is the <period> character. Codeword [279] provides a toggle between the ASCII <period> and the ISO<comma> decimal point characters.

- Codewords [128] to [227] encode digit pairs 00 through 99 respectively in a single codeword
- Codewords [228] to [237] encode the single digits 0 through 9 followed by the selected decimal point
- Codewords [238] to [247] encode the single digits 0 through 9 preceded by selected decimal point.

6.5.2.4 C43 Compaction Submode

Seven-bit ASCII Mode, 8-bit Mode and 7-bit ASCII Submode may all latch directly C43 Compaction Submode to encode either a fixed or an arbitrary multiple of 3 ASCII characters into 2 codewords each. This includes efficient encoding of entirely lower-case ASCII URLs.

C43 Compaction Submode uses radix conversion together with the three 43-character sets shown in Table 8. These include all the characters of the Code 39 C43 set and the Data Matrix C41 set, albeit in a different order. The Ultracode C43 Sets 1 and 2 supports typical bar code data. C43 Sets 3 is primarily tools for efficient URL encoding under RFC 3986. Set 1 contains uppercase alphanumerics and enables percent encoding of hexadecimal octets of form "%hh" under section 2.1 of RFC 3986. Set 2 contains all the punctuation symbols used as general delimiters. Set 3 contains most of the general and all of the sub-delimiters punctuation marks. Among the 3 sets are all the printable ASCII characters with values from 32-126.

Encoding of data always starts using a specified C43 encoding set, either Set 1 or Set 2. Latching between the C43 Sets 1 and 2 may be done freely within C43 Compaction Submode. The Shift command enables reaching into the alternate C43 set to encode a single character without latching. Encoded data may end in either C43 Set 1 or 2.

Unlike Sets 1 and 2, there is no latch to Set 3. Only single C43 characters may be encoded using a shift to Set 3. However, Set 3 is very efficient as it encodes standard top-level domains (ex: ".com", ".edu"), complex expressions (ex: <https://www.>) and common file suffixes (ex: ".html" and ".asp") as single C43 Set 3 characters.

A single codeword macro for starting efficient URL encoding is given in [6.5.3.7](#) for 8-bit encoding. Once that macro is invoked, actual encoding of the URL may be done using C43 submode and ASCII submode when long strings of digits are present.

Exactly three base-43 characters c_1 , c_2 , c_3 are encoded in two base-282 codewords using radix conversion in following formula, where character values c_1 , c_2 and c_3 are obtained from Table 11:

$$\begin{aligned} 1^{\text{st}} \text{ codeword} &= (43^2 c_1 + 43 c_2 + c_3) \text{ div } 282 \\ 2^{\text{nd}} \text{ codeword} &= (43^2 c_1 + 43 c_2 + c_3) \text{ mod } 282 \end{aligned}$$

In the case of variable length C43 encoding invoked by latching to C43 Compaction Submode for multiple groups of 3 characters, C43 compaction proceeds until a non-C43 character is found. The last C43 group may consist of 3 characters or 2 characters and a set latch command as a pad: A single C43 excess character is not encoded in C43 mode. Codeword [282] unlatches C43 Compaction Submode.

In the case of fixed-length C43 encoding invoked by latching to C43 Compaction Submode for a fixed number of C43 characters (including any required latches and shifts):

- It is assumed that there are the exact number of C43 Characters encoded, including any pad characters to fill the final triplet. One or two repeated alternating latch (42) characters are to be used as padding.
- No explicit unlatch is required. This saves an exit codeword for each C43 group.

Table 8 – ASCII C43 Compaction Submode character value assignment

C43 Set 1	C43 Set 2	C43 Set 3	Encoded Value, c		C43 Set 1	C43 Set 2	C43 Set 3	Encoded Value, c
A	a	http://	0		W	w	(22
B	b	https://	1		X	x)	23
C	c	http://www.	2		Y	y	"	24
D	d	https://www.	3		Z	z	+	25
E	e	ftp://	4		0	:	'	26
F	f	www.	5		1	/	<	27
G	g	.com	6		2	?	>	28
H	h	.edu	7		3	#		29
I	i	.gov	8		4	[\$	30
J	j	.int	9		5]	;	31
K	k	.mil	10		6	@	&	32
L	l	.net	11		7	=	\	33
M	m	.org	12		8	_	^	34
N	n	.mobi	13		9	~	*	35
O	o	.coop	14		<space>	!	.cgi	36
P	p	.biz	15		<period>	<period>	.asp	37
Q	q	.info	16		<comma>	<comma>	.aspx	38
R	r	mailto:	17		%	<hyphen>	.php	39
S	s	GS	18		Shift to Set 2 for 1 char	Shift to Set 1 for 1 char	.htm	40
T	t	{	19		Shift to Set 3 for 1 char	Shift to Set 3 for 1 char	.html	41
U	u	}	20		Latch to Set 2	Latch to Set 1	.shtml	42
V	v	`	21					

Seven-bit ASCII Mode, 8-bit Mode and 7-bit ASCII Submode may all temporarily latch directly C43 Compaction Submode for encoding groups of 6, 9, 12, or 15 Ultracode C43 characters. The calling mode's command codeword defines the number characters encoded, and the corresponding 4, 6, 8 or 10 codewords following that command codeword are therefore known to be C43 codewords. Because the position of the next command or data codeword in the calling mode is known, there is no need for a mode exit codeword in C43 Compaction Submode in these 4 cases.

All the C43 characters are included in 7-bit ASCII (ISO/IEC 646:1991) and therefore in the first 128 characters of ISO/IEC 8859/1. Therefore, ECI considerations for C43 Compaction Submode are the same as that of the calling base mode (see [6.5.2.1](#)).

6.5.2.7 Function codes FN1 and FNC3

Codeword [277] encodes a FNC1 character in the symbol. Usage should follow [Annex C.](#) Note that if there is a prefix to the data codewords in the symbol for any needed Start Sequence codeword for an ECI and any Symbol Control Region codewords, the first data codeword is count as the first codeword position immediately following this prefix.

Start Character [273] starts an ASCII symbol with an implied FNC1 in the first position, saving one codeword over the normal ASCII Start Character, [272]. Since double density numerics are only available in ASCII mode/submode this is the most efficient method of encoding GS1 data structures.

Codeword [278] encodes a FNC3 character in the symbol. This remainder of this Ultracode symbol is used internally for reader programming. Interpretation of the remaining codewords is on a reader-by-reader basis.

6.5.3 8-bit Mode

In 8-bit symbols, each input data value 0 through 255 corresponds to the 256 character addresses within the default ISO/IEC 8859/1 character set or the ECI-selected 8-bit or multi-byte character set. Start Characters are described in [6.2.3](#), with additional information on the Start Sequence for ECI-selected character sets in [6.5.3.1](#). Command codeword assignments are shown in Table 9.

6.5.3.4 Special considerations for byte mode encoding

When non-language byte data is encoded throughout an 8-bit symbol, language-related features such as C43 compaction may or may not be useful. The start codeword is [279] and there is no Start Sequence codeword [x], with ECI \000899 is implicit. The language based command set in Table 12 allows compaction of random sequences of data favorable to the use of ASCII Submode or C43 Compaction Submode.

6.5.3.7 URL C43 macro sequences

Encoding a command codeword in the range [277] to [281] provides a means of abbreviating a URL header using one symbol character, reducing the number of symbol characters required to encode URLs. These macro codewords respectively correspond to the URL prefixes:

- <http://>
- <https://>
- <http://www>.
- <https://www>.
- <ftp://>

While C43 Set 2 encoding for an unlimited number of characters is initiated, nothing is encoded in the C43 string itself: C43 encoding starts with the first character after the sequence replaced by the macro call. What the macro codewords does signal the decoder to insert the selected URL prefix to the C43 string in the decoded output. A [282] C43 mode exit command is required at the end of the C43 data. See Annex [E.4](#) for an example of use.

Data transmission from the reader shall conform to [12.2.4](#) based on the symbol type in use.

Table 9 – 8-bit Mode codeword assignments

Codeword values	Meaning or interpretation	Refer to:
0 to 255	8-bit character values	6.5.3
256	Temporarily latch to C43 Compaction Submode C1 for 6 C43 characters and return to 8-bit mode	6.5.2.4
257	Temporarily latch to C43 Compaction Submode C1 for 9 C43 characters and return to 8-bit mode	6.5.2.4
258	Temporarily latch to C43 Compaction Submode C1 for 12 C43 characters and return to 8-bit mode	6.5.2.4
259	Temporarily latch to C43 Compaction Submode C1 for 15 C43 characters and return to 8-bit mode	6.5.2.4
260	Latch to C43 Compaction Submode C1 for multiple groups of 3 characters and return to 8-bit mode	6.5.2.4
261	Select delimiter character ; C43 delimiter value follows (starting default is "%")	6.5.2.3
262	Temporarily latch to C43 Compaction Submode C2 for 6 C43 characters and return to 8-bit mode	6.5.2.4
263	Temporarily latch to C43 Compaction Submode C2 for 9 C43 characters and return to 8-bit mode	6.5.2.4
264	Temporarily latch to C43 Compaction Submode C2 for 12 C43 characters and return to 8-bit mode	6.5.2.4
265	Temporarily latch to C43 Compaction Submode C2 for 15 C43 characters and return to 8-bit mode	6.5.2.4
266	Latch to C43 Compaction Submode C2 for multiple groups of 3 characters and return to 8-bit mode	6.5.2.4
267	Latch to ASCII Submode	6.5.2
268	FNC1 , use per Annex D	6.5.2.7
269	FNC3 , reader programming	6.5.2.7
270	In 8-bit Bar Code Emulation Mode symbols only: Start new bar code emulation , Symbology Identifier and option value follow (<i>invalid in non-emulation mode symbols</i>)	7.5
271	Encode 06 Macro Sequence in 8-bit default character set or or Bar Code Emulation Mode symbols only (<i>otherwise no action</i>)	6.5.2.6
272	Restart 8-bit Mode with new 8-bit character set corresponding to ECI \h. Two 2 codeword parameters y and x follow , where $h = 256y + x$ and h ranges from 3 to 899	6.5.3.2
273	Invoke Encodable ECI sequence , 1 parameter codeword follow	6.6.2
274	Invoke Encodable ECI sequence , 2 parameter codewords follow	6.6.2
275	Invoke Encodable ECI sequence , 3 parameter codewords follow	6.6.2
276	Encode http : URL C43 macro sequence	6.5.3.7
277	Encode http:// URL C43 macro sequence	6.5.3.7
278	Encode https:// URL C43 macro sequence	6.5.3.7
279	Encode http://www. URL C43 macro sequence	6.5.3.7
280	Encode https://www. URL C43 macro sequence	6.5.3.7
281	Encode ftp:// URL C43 macro sequence	6.5.3.6
282	Unlatch C43 Compaction Submode and return to 8-bit Mode	6.5.4.2

6.6 Extended Channel Interpretation

The Extended Channel Interpretation (ECI) protocol allows the output data stream to have interpretations different from that of the default character set. The ECI protocol is fully specified in the *AIM International Technical Specification: Extended Channel Interpretation (ECI) – Part 1: Identification Schemes and Protocol*. The ECI protocol provides a consistent method, independent of symbology or data carrier type, to specify particular interpretations of data values during symbol decoding.

6.7.1 Reed-Solomon error control process

The RSEC codewords can be used to correct *erasures* (erroneous codewords at known locations) and *errors* (erroneous codewords at unknown locations). An erasure is typically an unscanned, damaged, or undecodable symbol character tile. An error is typically a misdecoded symbol character tile. The relation between the number of erasures and errors correctable within a complete block is given by:

$$e + 2t \leq Q - P = E_{cap}$$

where: e = number of erasures
t = number of errors
Q = total number of RSEC Codewords per symbol
P = number of misdecode protection codewords included in Q
 E_{cap} = maximum number of codewords that can be error corrected in the symbol

If all of the error control capacity was used to correct erasures (i.e., if $P = 0$), then the possibility of an undetected error increases. Since the Ultracode symbology is designed for "rough service environments," erasures may occur frequently due to torn, smeared, faded, scratched, overwritten or otherwise damaged symbols. Therefore, a constant value $P = 3$ codewords are reserved at all EC levels in all sizes of Ultracode symbols to prevent misdecoding of a damaged symbol containing a large number of erasures. The probability of a misdecoding of any Ultracode symbol is estimated at $1: 2 \times 10^{11}$ in [Annex B.6](#).

6.7.2 Number of RSEC Codewords used

The encoded Message Codewords are comprised of a sequence of T data codewords plus w structural codewords. At the selected EC level, there will be n codewords available in the single 282-codeword block to accommodate the Message Codewords, and the Message Codeword Sequence will rarely fill all n available codewords exactly. The number of surplus codewords, V, which are truncated from the symbol, is calculated by the formula in [6.7.5](#).

There are from 3 to 77 RSEC Codewords present in the symbol. Six EC levels are provided, with EC0 being error-detection (i.e., no error-correction capability) using the $P=3$ misdecode protection codewords (see [6.7.3](#)). Levels EC1 through EC5 have increasing percentages of error correction capability, with a minimum of $ACC = (Q-P) = 3$ codewords available for error correction even if the Data Region has but a single data codeword. The minimum percentage of error correction codewords may be selected, appropriate to the number of Message Codewords by use of Table 10. The default error correction level is EC2, 9% minimum.

The formula for calculating the number of RSEC Codewords, Q, given the EC level, EC1 through EC5; the number of misdecode protection codewords, which for Ultracode symbols is always $P = 3$; and the number of Message Codewords, C, (data plus w structural codewords) is:

if $(C \bmod 25) = 0$ **then** $Q = K(EC) \times (C \div 25) + P + 2$ **else** $Q = K(EC) \times ((C \div 25) + 1) + P + 2$

Table 10 – Parameters to calculate the number of RSEC Codewords

EC level	Description of error control provided	ACC/Message Codewords, $(Q-P)/C$	K(EC) value	C, Maximum # of Message Codewords, 5-Row symbol	Min RSEC Codewords $(ACC_{min} + 3)$	Max RSEC Codewords $(ACC_{max} + 3)$
EC0	None	0%	—	274	3	3
EC1	Low	$\geq 5\%$	1	261	6	16
EC2	Normal (default)	$\geq 9\%$	2	250	7	27
EC3	Medium	$\geq 18\%$	4	232	9	45
EC4	High	$\geq 25\%$	6	218	11	59
EC5	Extreme	$\geq 36\%$	8	200	15	77

This formula ensures that when C is an exact multiple of 25 codewords, the minimum specified percent ratio of the ACC Ratio (Ratio of ACC codewords to Message Codewords) as a function of the specified EC level is always met. The actual percentage of ACC Codewords will be slightly higher when C is less than an exact multiple of 25. An example is shown in Table 11, for the "Normal" or default error control level, EC2.

Table 11 - ACC Ratio for default EC2 symbols as a function of Message Codewords

Message Codewords, C	(C div 25)	Total RSEC Codewords, Q In Symbol	ACC, Codewords Available for Error Correction, (Q-P)	% Ratio of ACC/Message Codewords
1	0	7	4	400.0%
6	0	7	4	66.7%
12	0	7	4	33.3%
24	0	7	4	16.7%
25	1	7	4	16.0%
26	1	9	4	15.4%
50	2	9	6	12.0%
51	2	11	6	11.8%
75	3	11	8	10.7%
76	3	13	8	10.5%
100	4	13	10	10.0%
101	4	15	10	9.9%
150	6	17	14	9.3%
151	6	19	14	9.3%
200	8	21	18	9.0%
201	8	23	18	9.0%
250	10	25	22	8.8%
251	10	27	24	9.6%
255	10	27	24	9.4%

6.7.3 EC0 option for error detection only

EC0 produces the smallest possible Ultracode symbol, but has no error correction capability. At EC0, the P=3 RSEC Codewords per block allocated for misdecode protection are used for Reed-Solomon error detection only. Since ACC=0 always, no error correction is performed on decode.

6.7.4 Generating the RSEC Codewords

The RSEC Codewords for each block are generated using polynomial arithmetic in a prime Galois Field of size GF(283) with a Prime Modulus of M=3. See [Annex B.4](#) for further details.

7 Emulation of other bar code symbologies

7.1 Overview

The Ultracode Bar Code Emulation Mode symbols provide a simple upgrade path from applications utilizing many older 7-bit/8-bit symbologies to the use of the error-correcting Ultracode symbol types. This upgrade is through emulation of those older symbologies within Ultracode symbols, including acceptance of the existing character sets, character values and data structures.

Existing bar code data structures may be carried within Ultracode symbols exactly as they would appear in their original symbologies. The use of an Emulated Symbology Identifier string "masquerading" as that of the original bar code is supported (see [7.6](#)) in each emulation.

The advantage of keeping the bar code emulation function at the character set level is that it can be performed entirely within the firmware of the bar code printer and scanner. This means that there is

no necessity to rewrite most of the application software surrounding the printer and scanner, as the input data structure into the bar code printer and the output data structure from the scanner are unchanged; only the printed symbol is different.

During scanning, each bar code emulation within a symbol is output by the scanner as a separate transmitted message (see [12.6](#)) with its own Emulated Symbology Identifier (ESI) Sequence, if Symbology Identifier output is enabled in the reader.

7.2 *Bar codes emulated*

Any symbology with characters in either the ASCII or the 8-bit character of ISO/IEC 8859/1 (Latin 1) and assigned Symbology Identifiers may be emulated. This includes linear (1-dimensional) and non-error correcting stacked (2-dimensional) symbologies such as Codabar, Code 11, Code 16K, Code 39, Code 49, Code 93, 93i, Code 128, Interleaved 2 of 5 and all the other 2 of 5 variants. See [Annex J.1](#) for symbology document sources.

7.6 *Emulated Symbology Identifier (ESI) Sequence*

Assuming Symbology Identifiers are enabled by the reader, the transmitted data of each emulated bar code masquerades as the complete output of the original bar code by transmission of the emulated bar code's Symbology Identifier and option value (ESI Sequence) prior to the data of the emulated bar code.

There are certain cases where transmission of the ESI Sequence is essential, such as when one or more of the emulated bar codes is Code 128 containing an FNC1 in the first or second data character position.

In either a 7-bit or an 8-bit Bar Code Emulation Mode symbol, to masquerade as a native bar code, the two components of the ESI Sequence must be encoded. Two 7-bit ASCII characters for the emulated bar code's Symbology Identifier defined under ISO/IEC 15424 and an appropriate option value, m are encoded as a symbol's Start Sequence as prefix to the data. Each subsequent emulated bar code in the symbol has an similar ESI Sequence encoded as a Start Sequence which is transmitted as a prefix to the data (if enabled in the reader).

An example showing the use of the ESI Sequence in accordance with the rules above is given in [Annex G.5](#).

8 Ultracode printing and display considerations

8.1 *Processes used*

It is assumed that most processes printing Ultracode symbols will use CMYK subtractive color printing processes and synthesize the RGB modules. Common processes are color inkjet or color laser printing or flexographic or offset lithographic printing.

The symbol may also be displayed rather than printed, as on a mobile telephone, LCD monitor or other RGB display, with the CMYW colors synthesized using the additive color process.

8.2 *Nominal dimensions*

Ultracode symbols are designed to be printed with square modules of nominal extent X in both orthogonal directions, and shall conform to the following nominal dimensions:

Module width: The height and width, X, of a module shall be specified by the application; taking into account the scanning technology to be used, and the technology employed to produce the symbol.

The value of X shall be a minimum of 5 printer or display pixels

Minimum Quiet Zone: 1X minimum on all 4 sides

11 Ultracode usage considerations

11.1 Autodiscrimination Capability

Ultracode symbols may be read by suitably programmed decoders which have been designed to autodiscriminate it from other symbologies. However, representations of short linear symbols may be found in any matrix symbol, including the Ultracode symbol types. Therefore, the decoder's valid set of symbologies should be limited to those needed by a given application to maximize reading security.

11.2 Symbol overhead

Each Ultracode symbol includes structural codewords as symbol overhead, exclusive of any data encoded. Overhead tiles include graphical patterns and structural codewords. All structural codewords are required even if the symbol contains neither data, nor a Symbol Control Region, nor any additional RSEC Codewords for error correction.

	1	Start Character codeword
	1	Message Codeword Count (MCC)
	3	RSEC misdecode protection codewords (P)
	1	Available Error Correction Codeword count (ACC)
Overhead:	6	Codeword tiles

Graphical overhead tiles are used to delineate the structural and data codewords.

	1	Total Codeword Count Pattern (TCC)
	1	RSEC/Data Separator Pattern
	1	QCC Pattern
Overhead:	3	Pattern tiles

The total Ultracode symbol overhead is 9 tiles per symbol.

11.3 Minimum symbol size

Under the assumptions of 11.2 the minimum 2-row Ultracode symbol at EC0 and containing either 0 or 1 data codewords occupies a rectangular area is 13X wide by 21Y high including Quiet Zones. In Figure 12, the blue tiles are overhead codewords; the yellow tiles are overhead patterns; and the green tile is either a data or pad codeword.

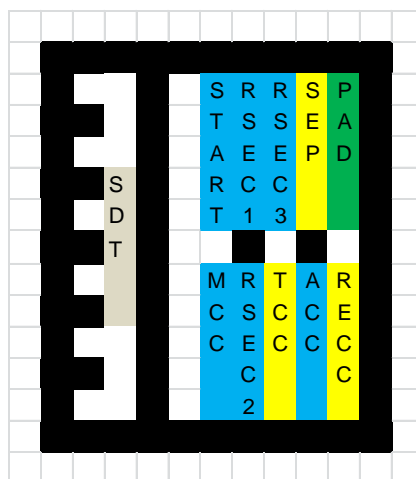


Figure 13 – Minimum Ultracode Symbol

11.3 Maximum Message Codeword capacity

The theoretical maximum tile capacity is of the largest 5-row Ultracode symbol is 280 tile, of which 9 are used for overhead. To estimate the maximum Message Codeword capacity this must be reduced by both the symbol structural codewords, w , and the codewords used for the Reed-Solomon error correction for the EC level in use.

Three illustrative cases are examined: Decimal digits, byte data and uppercase English text. The maximum number of characters of each data type encodable in the available data codewords are calculated for error control levels EC0 through EC5 and shown in Table 24. Three overhead structural codewords are assumed throughout (see [11.2.1](#)). In addition, it is assumed that there are no command codewords used other than those specifically required to initiate the encoding mode.

11.3.1 Decimal digits

Within a 7-bit symbol, two decimal digits are encoded at two digits per codeword. At default EC2 level, 494 digits may be encoded in this manner.

11.3.2 Byte data

Byte data is encoded in 8-bit symbols with ECI \000899 using one byte per codeword. At default EC2 level, 247 bytes may be encoded.

11.3.3 Uppercase Latin alphabet

Within a 7-bit symbol, the C43 Compaction Submode may be used to encode the characters of the uppercase Latin alphabet at 3 characters per 2 codewords. Occasionally codewords can be encoded prior to shifting to C43 Compaction Submode for an unlimited number of characters, with the end of the symbol functioning as a C43 mode terminator. At default EC2 level, 369 characters may be encoded.

Table 24 - Maximum Ultracode symbol data capacity

	EC0	EC1	EC2	EC3	EC4	EC5	EC Level
Total RSEC codewords (Q+P)	3	16	27	45	59	77	codewords
ACC codewords available for error correction, Q	0	13	24	42	56	74	codewords
Start, MCC, ACC overhead	3	3	3	3	3	3	codewords
Available data codewords (T+ACC)	271	258	247	229	215	197	codewords
Maximum encodable digits (ASCII symbol, 2 digits/codeword)	542	516	494	458	430	394	digits
Maximum encodable byte data (8-bit non-language symbol start)	271	258	247	229	215	197	bytes
Maximum uppercase English characters (Assumes ASCII symbol, C43 mode; encoded at 3 char/2 codewords; any extra chars encoded at 1/codeword prior to C43 mode start which requires 1 codeword)	405	385	369	342	321	294	chars

Annex D (Normative) Design of Ultracode tiles

D.1 Color science considerations in Ultracode tile design

D.1.1 Colors utilized in Ultracode

The primary colors red, green and blue (RGB) and the secondary colors cyan, magenta and yellow (CMY) are interrelated in printing processes. Modern 4-color electronic printing is typically based on *subtractive color mixing* of the 3 secondary colors cyan, magenta and yellow together with black, known as CMYK printing. The CMYK dyes or pigments absorb all wavelengths of light other than those for which they are named and reflect only that color for which they are named: Yellow dyes or pigments look yellow because all other colors have been *subtracted* from either the transmitted or reflected white light. Combining dyes or pigments from two secondary colors produces a primary color:

$$C + M = B \quad C + Y = G \quad M + Y = R$$

Black (K) and white (W) may also be generated:

$$C + M + Y = K$$

$$R + G + B = W$$

D.1.2 Color definition

Colors C,M,Y,R,G,B,W,K as used here are defined in hue and intensity in IEC 61966-2-1 (1999-10), *Multimedia systems and equipment — Colour measurement and management — Part 2-1: Colour management — Default RGB colour space — sRGB*.

This defines to the sRGB system used in digital cameras and displays, as seen at a color temperature of 6500 °K.

D.1.3 The color hexagon

The CMY and RGB colors may be combined in a single hexagonal diagram as in Figure D-1 where each primary color RGB is produced by mixing the dyes or pigments of the secondary color on the adjacent vertices, and the color at vertex opposite the primary color is the secondary color, which, when mixed with that primary color, would produce black (K).

The 6 colors RGB and CMY in Figure D-1 are now arbitrarily assigned numeric values to allow for ease of tile naming.

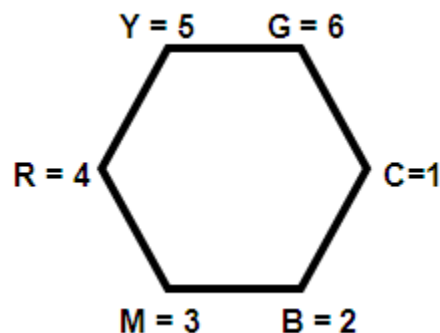


Figure D-1 – Value assignments to each vertex color

There are two additional colors which must be considered: W, white, the nominal color of the surrounding Quiet Zone, and K, black, the color of the Guard Pattern modules. The additional colors W (white) and K (black) are arbitrarily assigned values $W = 0$ and $K = 7$.

D.1.4 The magenta-fading problem

Many printing inks and CMYK inkjet dyes suffer from fading of the magenta color under UV light exposure or due to oxidation. Only recently have inorganic magenta pigments been developed which are fade resistant. Under conditions of UV (or direct sunlight) exposure, the magenta dye-based inks often fade, ultimately to white.

By selection of only C, M, Y, G, K and W for the Ultracode symbol design, only 1 color is subject to fading: Magenta itself, fading to white.

See the example in Figure D-3 for the original symbol and the same symbol where only the magenta dye has completely faded out, leaving only a CWY GK image. The rest of the colors are invariant in the case of magenta (only) fading. When there are no magenta tiles present, it may be inferred that the white modules in the data encodation tiles are faded magenta tiles, and decoding proceeds as if these white modules were magenta.

Since both the below symbols are "equivalent" why bother with magenta at all?

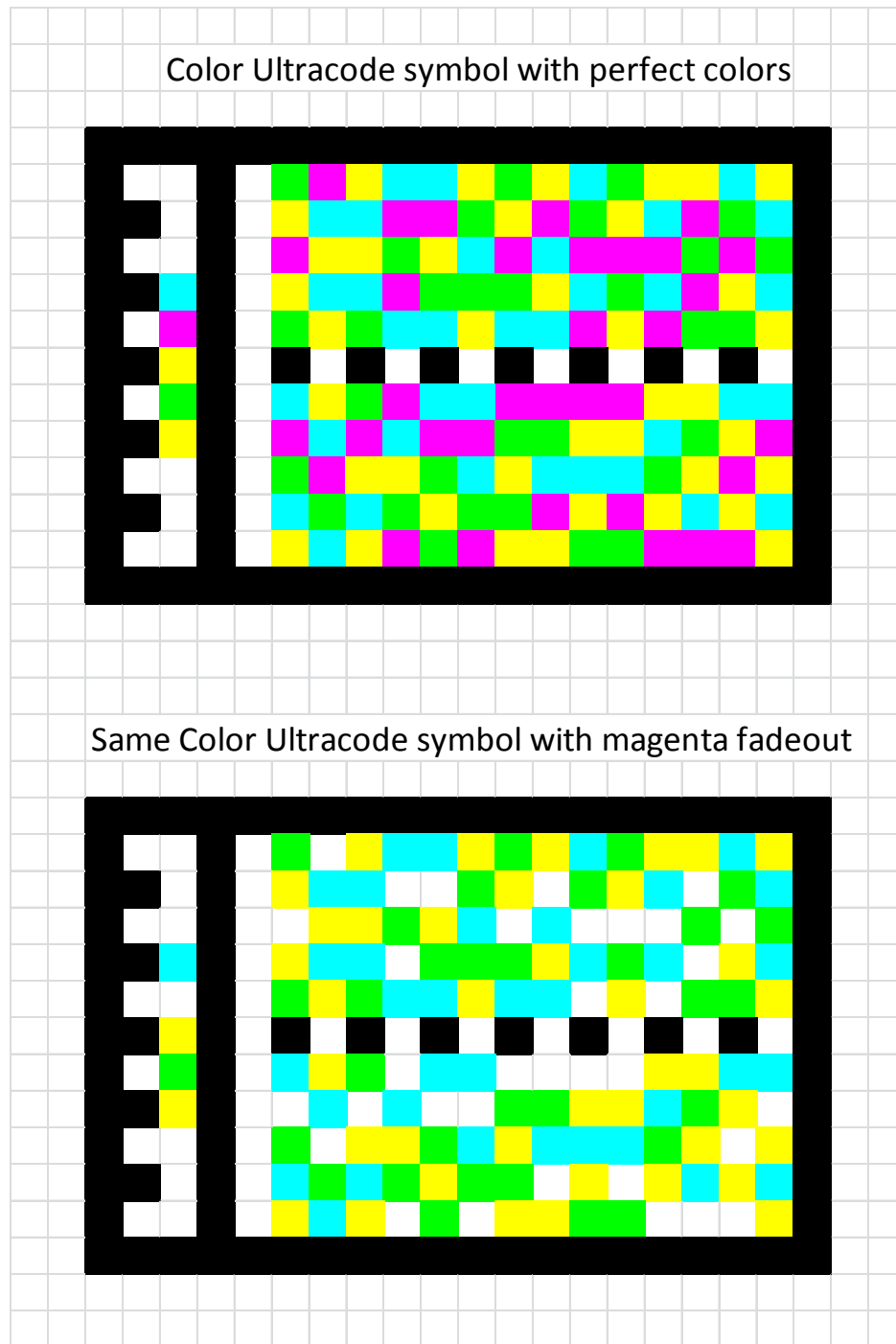


Figure D-3 – A Ultracode symbol before and after magenta fadeout

D.2 Tile designs

D.2.1 Data density versus decodability

In theory, the use of tile designs that incorporate all six colors C, M, Y, R, G, B would allow 283 codeword values to be encoded in 4 modules. (6^4 combinations). Other color bar codes have been designed which focused on maximizing data spatial density through the use of color. In practice the more color used, the more difficult it is to discriminate between them, particularly under poor lighting

conditions, especially red versus magenta and blue versus cyan. Magenta fading only makes the decoding problem worse.

By restricting the color tile set to C, M, Y, G and adding a fifth module to encode the 283 codewords, magenta fading can be dealt with easily, and the colors are more easily discriminated than if R and B were also present. Using color, the 283 codewords can be encoded in 5 color modules using tiles with strong tile self-checking rules, as contrasted to 9 black and white modules without self-checking. The Ultracode tiles are 45% smaller than a black and white tile, but still retain tile self-checking, resistance to magenta fade, and good color separation to enhance decodability.

D.2.2 Rules for data encodation tile design

Each data encodation tile in Ultracode consists of 5 modules and complies with five design rules:

- Rule 1. The modules in rows 1 through 5 are either all from the CMYG set.
- Rule 2. No two vertically-adjacent modules in any tile may be the same color, they are always separated by a black or white horizontal clock track module

The Rule 2 restriction that no 2 adjacent tiles are the same color limits the number tiles available. The total number of compliant tile design choices is determined as follows:

Row 1: 4 color choices
Rows 2-5: 3 color choices each

Total compliant tile designs = $4 \times 3 \times 3 \times 3 \times 3 = 324$

The total number of CMY compliant tile designs was reduced from $4^5 = 1\,204$ to 324 by the use of Rule 2. This set was further reduced by Rules 3:

- Rule 3. All data encodation tiles must have 3 or 4 colors present (no 2-color tiles)

The total number of compliant tile designs was found to be 313 from the 2-color restriction of Rule 3.

- Rule 4. No data encodation tile may have 3 modules of the same color.
- Rule 5. There must be at least 1 Y or G module in each data encodation tile, as these are considered Y and C are consider the most fade resistant

Rules 4 and 5 further reduced the number of available data encodation tiles to 287. Four additional tiles with GCGC or CGCG sequences were eliminated: CGCGY, MCGCG, MCGCG and GCGCM. This reduced the data encodation tile set to 283, matching the GF(283) tile requirement.

D.2.3 Design of data encodation tiles

Data encodation tiles are utilized for encoding codewords in the RSEC and Data Regions of a Ultracode symbol. In addition, they are used to encode the start, MCC and ACC character codewords and the TCC and QCC pattern values.

Each data encodation tile can then be described by a 5-octal-digit (15-bit) number, where the highest octal digit corresponds to the color of the module in top row and the lowest digit corresponds to the color of the module in the bottom row. Using the numeric assignments of Figure D-1, C=1, M=3, Y=5 and G=6. Note the use of separating commas to enhance readability.

Ex: The tile: CGMYC is written: $16,351_8$

Table D-5 contains the CMYG set of 283 Data Encodation tiles. The left-most column contains the codeword value, and then octal tile color values for the CMYG encodation of that codeword value are given along with the tile designs.

Command codewords 256-282 were assigned a G module in their top row. (Note that the data encodation tiles for codewords 212-255 have a G module in their top row as well.)

Table D-5 Data Encodation tiles

<u>CW</u>	<u>Octal</u>	<u>Top</u>						<u>CW</u>	<u>Octal</u>	<u>Top</u>					
0	13,135							50	16,156						
1	13,136							51	16,165						
2	13,153							52	16,313						
3	13,156							53	16,315						
4	13,163							54	16,316						
5	13,165							55	16,351						
6	13,513							56	16,353						
7	13,515							57	16,356						
8	13,516							58	16,361						
9	13,531							59	16,363						
10	13,535							60	16,365						
11	13,536							61	16,513						
12	13,561							62	16,515						
13	13,563							63	16,516						
14	13,565							64	16,531						
15	13,613							65	16,535						
16	13,615							66	16,536						
17	13,616							67	16,561						
18	13,631							68	16,563						
19	13,635							69	16,565						
20	13,636							70	31,315						
21	13,651							71	31,316						
22	13,653							72	31,351						
23	13,656							73	31,356						
24	15,135							74	31,361						
25	15,136							75	31,365						
26	15,153							76	31,513						
27	15,163							77	31,515						
28	15,165							78	31,516						
29	15,313							79	31,531						
30	15,315							80	31,535						
31	15,316							81	31,536						
32	15,351							82	31,561						
33	15,353							83	31,563						
34	15,356							84	31,565						
35	15,361							85	31,613						
36	15,363							86	31,615						
37	15,365							87	31,631						
38	15,613							88	31,635						
39	15,615							89	31,636						
40	15,616							90	31,651						
41	15,631							91	31,653						
42	15,635							92	31,656						
43	15,636							93	35,131						
44	15,651							94	35,135						
45	15,653							95	35,136						
46	15,656							96	35,151						
47	16,135							97	35,153						
48	16,136							98	35,156						
49	16,153							99	35,161						

Table D-5 (continued) Data Encodation Tiles

<u>CW</u>	<u>Octal</u>	<u>Top</u>						<u>CW</u>	<u>Octal</u>	<u>Top</u>					
100	35,163							150	51,516						
101	35,165							151	51,531						
102	35,315							152	51,536						
103	35,316							153	51,561						
104	35,351							154	51,563						
105	35,356							155	51,613						
106	35,361							156	51,615						
107	35,365							157	51,616						
108	35,613							158	51,631						
109	35,615							159	51,635						
110	35,616							160	51,636						
111	35,631							161	51,651						
112	35,635							162	51,653						
113	35,636							163	51,656						
114	35,651							164	53,131						
115	35,653							165	53,135						
116	35,656							166	53,136						
117	36,131							167	53,151						
118	36,135							168	53,153						
119	36,136							169	53,156						
120	36,151							170	53,161						
121	36,153							171	53,163						
122	36,156							172	53,165						
123	36,163							173	53,513						
124	36,165							174	53,516						
125	36,315							175	53,531						
126	36,316							176	53,536						
127	36,351							177	53,561						
128	36,356							178	53,563						
129	36,361							179	53,613						
130	36,365							180	53,615						
131	36,513							181	53,616						
132	36,515							182	53,631						
133	36,516							183	53,635						
134	36,531							184	53,636						
135	36,535							185	53,651						
136	36,536							186	53,653						
137	36,561							187	53,656						
138	36,563							188	56,131						
139	36,565							189	56,135						
140	51,313							190	56,136						
141	51,315							191	56,151						
142	51,316							192	56,153						
143	51,351							193	56,156						
144	51,353							194	56,161						
145	51,356							195	56,163						
146	51,361							196	56,165						
147	51,363							197	56,313						
148	51,365							198	56,315						
149	51,513							199	56,316						

Data Encodation Tiles

CW	Octal	Top
200	56,351	
201	56,353	
202	56,356	
203	56,361	
204	56,363	
205	56,365	
206	56,513	
207	56,516	
208	56,531	
209	56,536	
210	56,561	
211	56,563	
212	61,313	
213	61,315	
214	61,316	
215	61,351	
216	61,353	
217	61,356	
218	61,361	
219	61,363	
220	61,365	
221	61,513	
222	61,515	
223	61,516	
224	61,531	
225	61,535	
226	61,536	
227	61,561	
228	61,563	
229	61,565	
230	61,615	
231	61,631	
232	61,635	
233	61,651	
234	61,653	
235	63,131	
236	63,135	
237	63,136	
238	63,151	
239	63,153	
240	63,156	
241	63,161	
242	63,163	
243	63,165	
244	63,513	
245	63,515	
246	63,516	
247	63,531	
248	63,535	
249	63,536	

D.2.4 RSEC/Data Separator Pattern and pad character pattern designs

These are shown in Figure 6 in [4.6.7](#) and Figure 7 in [4.6.12](#) respectively.

Both are visually distinguished from data encodation tiles in the following manner:

- They have 3 Y modules, at the top, center and bottom (violate Rule 4)
- They have only 2 colors (violate Rule 2)
- Neither encode a codeword or pattern value
- The Separator Pattern is $56,565_8$
- The pad character pattern is $51,515_8$

D.2.5 Tile self-checking algorithm

The tile self-checking algorithm appears in Figure 11 in [10.3](#). Its purpose is to determine whether a tile is valid, and if so, recover the associated codeword or pattern value. If not, the tile is marked as erasure.

Codeword tile values for erasure tiles are later recovered using Reed-Solomon Error Correction. Since pattern tiles are not included in the RSEC process, the values of erasure pattern tiles are simply ignored, or may be recovered by other means (see Table 17 in [10.3](#)).

The algorithm uses the 5 tile design rules of [D.2.1](#). It also detects and checks the Separator and pad character patterns.

Annex G (informative)

Examples of symbol encodation

This clause illustrates through examples how the various encoding modes and features may be used in practice. Printed symbols are provided here which may be used for scanner testing or comparison with printing programs.

G.1 Encoding of 7-bit ASCII (ISO/IEC 646:1991) data

The input data "ULTRACODE_123456789!" is encoded at default error correction level EC2 (nominal 9%). Since the data is all ASCII characters, a 7-bit symbol is used. The Ultracode symbol appears in Figures 4 and G-1.

There are 20 data characters, which by use of C43 submode (saves 2 codewords) and double-density numeric compaction (saves 4 codewords) require only 14 Data Region codewords to encode. Three structural codewords are required for the Start Character, the MCC and the ACC. From Table 18 in [6.7.2](#), 7 RSEC codewords are used, for a total of 24 symbol codewords. From Table 1, a 2-row symbol is selected, as it can hold up to 31 codewords.

The Ultracode symbol uses the following codeword and special pattern value sequence with colored tiles from Table D-2 in [Annex D.3.2](#).

Character	Codeword	Octal Tile Design	Additional Explanation
Start Character	[273]	65,536 ₈	column 1
(MCC)	[17]	13,616 ₈	
RSEC Region:			
7 RSEC codewords	[23]	13,656 ₈	col 2
	[79]	31,531 ₈	
	[278]	65,615 ₈	col3
	[3]	13,156 ₈	
	[163]	51,656 ₈	col 4
	[281]	65,651 ₈	
	[178]	53,563 ₈	col 5
Total Codeword Count (TCC) = 24		15,135 ₈	Pattern
Data/Separator pattern		56,565 ₈	col 6
EC Codeword Count (ACC)	[4]	13,163 ₈	#RSEC for error correction
Data Region:			
Start C43 compaction	[275]	65,363 ₈	col 7: C43: 12 chars in 9 CW
"ULT"	[132]	36,515 ₈	
	[248]	65,535 ₈	col 8
"RAC"	[111]	35,631 ₈	
	[133]	36,516 ₈	col 9
"ODE"	[92]	31,656 ₈	
	[75]	31,365 ₈	col 10
" "	[95]	35,136 ₈	Single ASCII character
"12"	[140]	51,313 ₈	col 11: Dble-density numerics
"34"	[162]	51,653 ₈	
"56"	[184]	53,636 ₈	col 12
"78"	[206]	56,513 ₈	
"9"	[57]	16,356 ₈	col 13: Leftover digit as ASCII
"!"	[33]	15,353 ₈	
Pad character pattern		51,515 ₈	col14
QCC pattern, value = 7		13,515 ₈	Always last character encoded

The final Ultracode symbol is shown in Figure G-1 and is 15X by 22X including Quiet Zones.

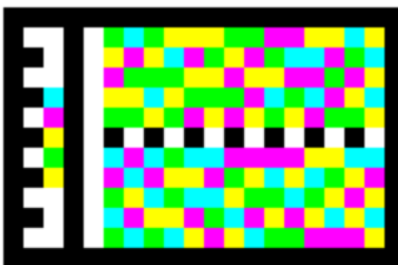


Figure G-1 – Ultracode ASCII EC2 symbol encoding “ULTRACODE_123456789!”

G.2 Western European language encoding using ISO 8859/1

This example encodes an Icelandic university name: “HEIMASÍÐA KENNARAHÁSKÓLA ÍSLANDS” (*Iceland University of Education*) at EC2 using 8-bit encoding with ISO 8859/1. A complete discussion of the markup and encoding appears as an example in [Annex E.3.2](#).

Since the Total Codeword Count (TCC) is 40, a 3 row symbol is required.

The start codeword specified the default 8-bit encoding of ISO/IEC 8859/1 throughout, with no Start Sequence parameter required. The complete Symbol Codewords Sequence is as follows:

Character	Codeword	Explanation	Reference
Start Character	[257]	8-bit ISO 8859/1 (Western European)	
MCC	[31]		
RSEC codewords	[249], [195], [87], [130], [155], [236], [85], [190], [29]		
TCC pattern = 40			
RSEC/Data Separator			
ACC	[6]		
C43 for 6 chars	[256]		
HEI	[46], [151]		
MAS	[78], [210]		
Í	[205]		
Ð	[208]		
C43 for 9 chars	[257]		
A K	[5], [148]		
ENN	[28], [72]		
ARA	[2], [167]		
H	[72]		
Á	[193]		
S	[83]		
K	[75]		
Ó	[211]		
L	[76]		
A	[65]		
	[32]		
Í	[205]		
C43 for 6 chars	[256]		
SLA	[119], [197]		
NDS	[85], [214]		
Pad Pattern			
QCC Pattern = 9			

The complete symbol is shown in Figure G-2. I uses 2 pad characters in the last column prior to the QCC. It is 21X high and is 23X long including Quiet Zones. It has 6 codewords available for correction of the 40 chraxater Message, or 15% error correction.

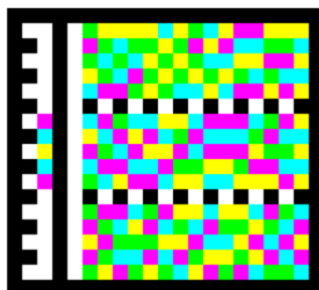


Figure G-2 – An Icelandic language symbol using 8-bit ISO 8859/1

G.3 Hebrew encoding examples

In Hebrew, alphabetic text is written right-to-left, but numbers are written left to right. An encoding of the Hebrew form of "Ultracode" (*shown below without vowel marks*) followed by the numeric sequence "1234" will now be examined. The complete message in human-readable form is:

1234אולטרה-קוד

The ordering of data in an Ultracode symbol is specified in 6.5.1 as the order in which human readable characters are *read*. Since the Hebrew segment precedes the numeric string and is written from right to left, therefore the numeric string(which is read from left to right) is written after (i.e., to the left of) the last Hebrew character. Therefore, the character reading order in the Ultracode symbol Data Region is: א, ו, ל, ט, ר, ה, -, ק, ו, ד, 1, 2, 3, 4.

The 8-bit symbol is encoded at EC2 using the 8-bit character set ISO/IEC 8859/8, *The Latin/Hebrew Alphabet*, with default ECI \000010. The Data Region continues with the 8-bit character codes, in order of their reading. The Symbol Tile Sequence is:

Character	Codeword	Additional Explanation
Start Character	[264]	ISO 8859/8
Message Codeword Count (MCC)	[16]	
7 RSEC codewords	[176],[82],[165],[201],[249],[274],[175]	
TCC Pattern = 23		
RSEC/Data Separator Pattern		
Error Corr Cwd Count (ACC)	[4]	#RSEC for error correction
א (Aleph)	[224]	0xE0
ו (Vav)	[229]	0xE5
ל (Lamed)	[236]	0xEC
ט (Tet)	[232]	0xE8
ר (Resh)	[248]	0xF8
ה (He)	[228]	0xE4
- (hyphen)	[45]	0x2D
ק (Kaf)	[247]	0xF7
ו (Vav)	[229]	0xE5
ד (Dalet)	[227]	0xE3
Latch to ASCII Submode	[267]	
12	[140]	128+12
34	[162]	128+34
QCC Pattern = 7		

The Hebrew symbol is shown in figure G-3 and is 15X by 21X including 1X Quiet Zones.

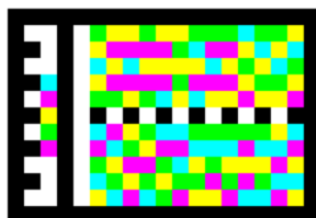


Figure G-3 – Hebrew Ultracode symbol using ISO 8859/8 encoding

G.4 Symbol encoding a URL

Other symbologies such as QR code are frequently utilized to encode URLs for access by mobile devices. These tend to have short URLs, for example :

<http://aimglobal.org/jcrv3tX>

For scanning reliability, a higher level of error correction is desirable. Here, EC2 is used, capable of correcting 4 tile erasures.

G.4.1 ASCII encoding of a URL

Since all URL characters are 7-bit ASCII, an ASCII start may be used

Here the C43 compaction macros for the <http://> and [.com](http://) in Table 2 Set 3 may be used to shorten the message even further. Following [Annex E.4](#) for the mark up and compaction process, the data is marked up as follows:

ey ——— <https://a img lob al .com/j crv 3t X>
km-----u uuu uuu uk m---uu uuu jcu s a

The C43 groupings are: kmu uuu uuu uuu kmu uuu jcu s a

Character	Codeword	Additional Explanation
Start Character	[273]	ASCII start
(MCC)	[20]	

RSEC Region:

7 RSEC codewords [86],[162],[138],[88],[266],[228],[158]

Total Codeword Count (TCC) = 27 Pattern

Data/Separator pattern

EC Codeword Count (ACC) [4] #RSEC for error correction

Data Region:

Latch set 2 for unlimited data [280]

" https:// "	[268],[276]	https:// macro in C43 Set 3; "a" in set 2
"img"	[54],[86]	
"lob"[[74],[74]	
"al"	[1],[232]	"al" in set 2; shift to set 3
".com"/"/j"	[82],[234]	".com" in set 3; encode '/j' in set 2
"crv"	[15],[220]	
"3t"	[266],[214]	Shift to Set 1, "3"; then "t" in Set 2
Exit C43	[282]	
'X'	[X]	in ASCII
No pad tiles needed		
QCC pattern, value	= 7	Always last character encoded

The final symbol is shown in Figure G-4a and is 15X by 23X including Quiet Zones. There are 6 available error correction codewords for the 20 codeword Message, or 30% error correction codewords. The symbol will be highly damage resistant.

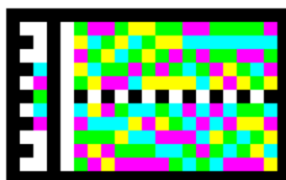


Figure G-4a – A 7-bit ASCII symbol encoding a URL

G.4.2 8-Bit Encoding of a URL

The same data is now encoded in 8-bit Mode at EC2 using the "[https://](#)" URL command macro [277] of [6.5.3.7](#) which also starts ASCII submode for C43 encoding in set 2. Then the C43 compaction macro in set 3 for the ".com" in Table 2 may be used to shorten the message even further. Following [Annex E.4](#) for the mark up and compaction process, the C43 data is marked up as follows:

```
em-----https:// aim glo bal .com/ jcr v 3 t x
----- uuu uuu uuu km---u uuu ujc ujc
```

Since the data end in C43 mode, with an even multiple of 3 characters, there is no need for an C43 mode exit character.

Character	Codeword	Additional Explanation
Start Character	[257]	ISO 8859/1 start
(MCC)	[18]	

RSEC Region:

7 RSEC codewords [108],[260],[52],[6],[88],[56],[241]

Total Codeword Count (TCC) = 23

Pattern

Data/Separator pattern

EC Codeword Count (ACC) [4]

#RSEC for error correction

Data Region:

[https://](#) command macro

Starts ASCII submode and latches

to set 2 for unlimited data [278]

encodes [https://](#)

"aim"

[1],[74]

"aim" in set 2

"glo"

[41],[19]

"bal"

[6],[168]

".com", "/"

[270],[212]

Shift to set 3 for ".com"; encode '/' in set 2

"jcr"

[59],[106]

"v3t"

[143],[252]

"v" then Shift to Set 1, "3"

"tX"

[131],[18]

"t" then Shift to Set 1 for "X"

No C43 mode exit tile needed

(end of symbol)

No pad tiles needed

QCC pattern, value

= 7

Always last character encoded

The final symbol is shown in Figure G-4b and is 15X by 21X including Quiet Zones. This is one column narrower than the ASCII encoding in Figure G-4a. The 8-bit symbol has 2 fewer Message codewords than the ASCII symbol(18), but the same number of available error correction codewords (6), for 33% error correction, which is slightly more damage resistant than the ASCII symbol.



Figure G-4b – A 8-bit ISO 8859/1 symbol encoding the same URL

G.4.3 Impact of macro and C43 compaction

For comparison with the symbols in Figure G-4a and G-4b, a non-standard ASCII symbol using only C43 compaction without macro compaction and without the start character macros was artificially constructed. This symbol uses unlimited C43 compaction, encoding all but the last 'X' in C43, and the 'x' in ASCII. Here the Message length is 26, but since the Message is longer there are proportionally more RSEC codewords, with now ACC=10 codewords available for error correction, or 38%. The symbol appears in Figure G-4c, and has 3 rows and 22 columns. An 8-bit symbol would be of similar size.

An ASCII-only symbol was artificially constructed, disabling C43 compaction, and is shown in Figure G-4d. Here the Message length is 32, and again since the Message is longer now there are again proportionally more RSEC codewords, with now ACC=13 codewords available for error correction, or 41%. An 8-bit symbol would be of similar size.

All 4 URL EC3 symbols encode <http://aimglobal.org/jcrv3tX> and are shown below at the same scale for comparison. Note that Figure G-4b, the 8-bit symbol using the URL command macro for <https://> is the smallest symbol.

Figure G-4a: ASCII mode

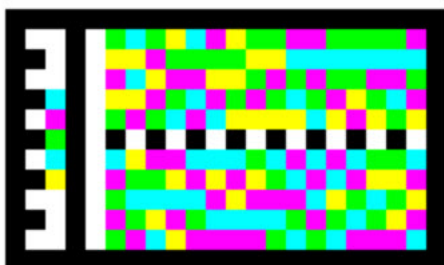


Figure G-4b: 8-bit mode

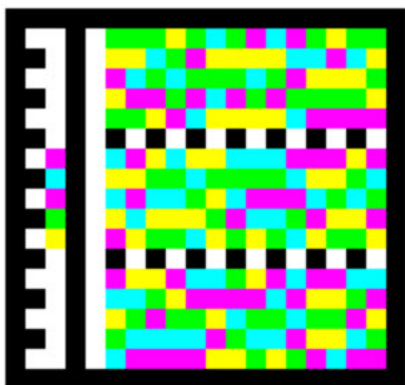
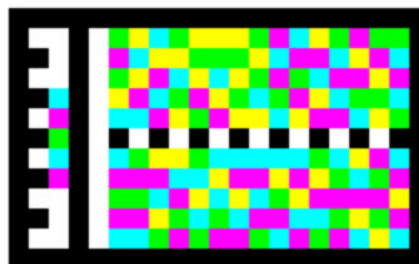


Figure G-4c: No macro compaction

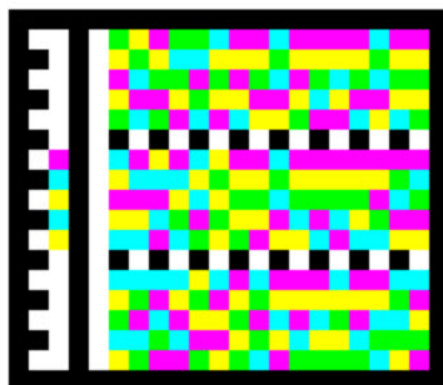


Figure G-4d: No C43 Compaction

G.5 Bar Code Emulation Mode symbol utilizing an ESI sequence

For example, construct a 7-bit Bar Code Emulation Mode symbol using EC2 containing 2 International Society for Blood Transfusion *ISBT128* data structures (See [Annex J.5](#)). These are typically found on a blood bag encoded as two separate Code 128 linear bar codes which in the ISBT-128 system may be concatenated during the reading process. However, when other bar code symbologies are used, the preferred method is to concatenate them in a single symbol.

The ASCII-character data structures are:

=W12340212345644 Blood donation number
=%6200 Blood type A, Rh (D) positive

For this bar code emulation to be interpreted properly by a blood bank computer system supporting the use of Symbology Identifiers, the Ultracode reader output must masquerade as if the data came from a concatenated read of two Code 128 symbols. The Code 128 Symbology Identifier "C" will be emulated with a modifier value "4" to indicate that a data concatenation according to ISBT-128 specifications has been performed and that concatenated data follows:

=W12340212345644=%6200

To encode the Emulated Symbology Identifier (ESI) Sequence, the concatenated bar code data structure must be prefixed by the ESI sequence "C4" which is treated as a Start Sequence. Note that use of the C43 compaction starts with the first "=", and does not include the Start Sequence parameters, which are never compacted (see 7.4.3).

The complete EC2 Symbol Tile Sequence is given below.

Character	Codeword	Explanation	Reference
Start Character	[270]	7-bit bar code emulation	see 6.2.3
MCC	[19]		
7 RSEC codewords	[268], [96], [279], [81], [11], [104], [264]		
TCC Pattern = 26			
RSEC/Data Separator			
ACC	[4]		
C	[67]	Symbology Identifier under emulation	
4	[52]	Modifier value	
=	[61]	Start of concatenated data	
W	[87]		
12,34	[140],[162]	Double density numerics	
02,12	[130],[140]		
34,56	[162],[184]		
44	[172]		
=	[61]		
%	[37]		
62,00	[190],[128]		
QCC Pattern = 7			

The printed symbol is shown in Figure G-5 and is 15x by 23x including Quiet Zones.

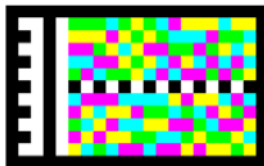


Figure G-5 – A 7-bit Bar Code Emulation Mode symbol with an ESI