

Data Encoding Protocol
Using Key-Length-Value



Table of Contents	Page
Forward	3
1 Scope	4
2 Conformance Notation	4
2.1 Conformance Implementations	5
3 Normative References	5
4 KLV Protocol	5
4.1 ANSI/SMPTE 298M Universal Label Key	6
4.1.1 UL Designator	8
4.1.2 Registry Designator	10
4.1.3 Structure Designator	10
4.1.4 Version Number	10
4.1.5 Item Designator	11
4.2 Encoding of the KLV Length Field	11
4.3 Encoding of Data Values	12
4.4 Empty Data Items	12
5 KLV Coding of Individual Data Items	12
5.1 Registers of Individual Data Items	12
5.1.1 Metadata Dictionaries	12
5.1.2 Essence Dictionaries	13
5.1.3 Control Dictionaries	13
5.1.4 Types Dictionaries	13
5.2 Identification of Value Data Representatives	13
6 KLV Group Coding	14
6.1 Universal Sets	15
6.2 Global Sets	16
6.3 Local Sets	19

6.4 Variable-Length Packs	21
6.5 Defined-Length Packs	23
6.6 Forbidden Use	25
7 Wrappers and Containers.....	25
8 SMPTE Labels.....	26
9 Registered Private Information	27
Annex A (Normative) Glossary of Terms and Acronyms.....	28
Annex B (Informative) Organization of Reference.....	30
Annex C (Informative) Example Usage of the SMPTE UL	31
Annex D (Informative) Example of the KLV Encoding of a Single Metadata Item	32
Annex E (Informative) Example of a Universal Set	33
Annex F (Informative) Example of a Global Set	34
Annex G (Informative) Example of a Local Set	35
Annex H (Informative) Example of a Variable-Length Pack	36
Annex I (Informative) Example of a Defined-Length Pack	37
Annex J (Informative) Example of a Label	38
Annex K (Informative) ASN.1 BER <i>Length</i> Coding	39
Annex L (Informative) ASN.1 BER Encoding of an Object Identifier Value.....	40
Annex M (Informative) Bibliography	41

Foreword

SMPTE (the Society of Motion Picture and Television Engineers) is an internationally-recognized standards developing organization. Headquartered and incorporated in the United States of America, SMPTE has members in over 80 countries on six continents. SMPTE's Engineering Documents, including Standards, Recommended Practices and Engineering Guidelines, are prepared by SMPTE's Technology Committees. Participation in these Committees is open to all with a bona fide interest in their work. SMPTE cooperates closely with other standards-developing organizations, including ISO, IEC and ITU.

SMPTE Engineering Documents are drafted in accordance with the rules given in Part XIII of its Administrative Practices.

SMPTE 336M was prepared by Technology Committee W25 on Metadata and Wrappers.

1 Scope

This standard defines a byte-level data encoding protocol for representing data items and data groups. This protocol defines a data structure which is independent of the application or transportation method used.

The standard defines a Key-Length-Value (KLV) triplet as a data interchange protocol for data items or data groups where the Key identifies the data, the Length specifies the length of the data and the Value is the data itself. The KLV protocol provides a common interchange point for all compliant applications irrespective of the method of implementation or transport.

The standard also provides methods for combining associated KLV triplets in data sets where the set of KLV triplets is itself coded with KLV data coding protocol. Such sets can be coded in either full form (Universal Sets) or in one of four increasingly bit-efficient forms (Global Sets, Local Sets, Variable Length Packs and Defined Length Packs). The standard provides a definition of each of these data constructs.

The standard also describes implications of KLV coding including the use of a SMPTE Universal Label as a value within a KLV coding triplet or whose meaning is entirely conveyed by the SMPTE UL itself. The two kinds of usage for such standalone SMPTE Universal Labels are a) as a value in a K L V construct and b) as a Key that has no Length and no Value. This standard defines where SMPTE ULs may be used for each kind of construct.

The standard also defines the use of KLV coding to provide a means to carry information that is registered with a non-SMPTE external agency.

The encoding byte range (length of the payload) specified in this document may accommodate unusually large volumes of data. Consequently, a specific application of KLV encoding may require only a limited operating data range and those details shall be defined in a relevant application document.

Annex A provides a glossary of terms used in this standard.

2 Conformance Notation

Normative text is text that describes elements of the design that are indispensable or contains the conformance language keywords: "shall", "should", or "may". Informative text is text that is potentially helpful to the user, but not indispensable, and can be removed, changed, or added editorially without affecting interoperability. Informative text does not contain any conformance keywords.

All text in this document is, by default, normative, except: the Introduction, any section explicitly labeled as "Informative" or individual paragraphs that start with "Note:"

The keywords "shall" and "shall not" indicate requirements strictly to be followed in order to conform to the document and from which no deviation is permitted.

The keywords, "should" and "should not" indicate that, among several possibilities, one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required; or that (in the negative form) a certain possibility or course of action is deprecated but not prohibited.

The keywords "may" and "need not" indicate courses of action permissible within the limits of the document.

The keyword "reserved" indicates a provision that is not defined at this time, shall not be used, and may be defined in the future. The keyword "forbidden" indicates "reserved" and in addition indicates that the provision will never be defined in the future.

2.1 Conforming Implementations

A conformant implementation according to this document is one that includes all mandatory provisions ("shall") and, if implemented, all recommended provisions ("should") as described. A conformant implementation need not implement optional provisions ("may") and need not implement them as described.

3 Normative References

The following standards contain provisions which, through reference in this text, constitute provisions of this recommended practice. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this recommended practice are encouraged to investigate the possibility of applying the most recent edition of the standards indicated below.

ANSI/SMPTE 298M-1997, Television – Universal Labels for Unique Identification of Digital Data

ISO/IEC 8825-1:2002 (ITU-T X.690), Information Technology – ASN.1 Encoding Rules – Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER), and Distinguished Encoding Rules (DER): BER Paragraphs 8.1.3.4 and 8.1.3.5 for *Length*, 8.19 for Object Identifier Coding.

4 KLV Protocol

Table 1 and Figure 1 present an introductory view of the KLV Protocol for encoding data. The data encoded may be a single data item, or a data group. The coding of data items is described in section 5 while the coding of data groups is described in Section 6 of this standard.

The KLV Coding Protocol is composed of a Universal Label (UL) identification "Key", followed by a numeric "Length" (Value Length), followed by the data "Value".

The composition of the Key is described in Section 4.1 of this standard. The length of the full Key shall be 16 bytes. The Length field is described in Section 4.2 of this standard. The byte-length of the Length field varies as defined in Section 4.2 of this standard. The Value is described in Section 4.3 of this standard. The Value is a sequence of bytes of the data type as specified in a relevant standard and is not further encoded by the KLV Protocol. The length of the Value field is variable and any limitations are defined in a relevant defining standard.

Table 1 – KLV Fields for Encoding of Data

Field	Description	Length	Contents/Format
Key	UL for identification of the Value	16 bytes	Section 4.1
Length	Length of the Value field	Defined in a relevant register, essence, application standard, but variable length	Section 4.2
Value	Value associated with the Key	Variable	Section 4.3

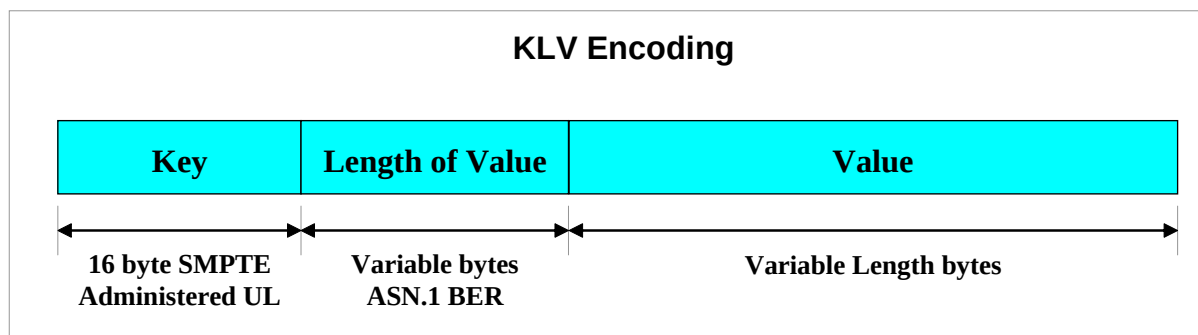


Figure 1 – KLV Encoding

4.1 ANSI/SMPTE 298M Universal Label Key

The KLV Coding Protocol shall use and only use a fixed 16-byte SMPTE-administered Universal Label, according to ANSI/SMPTE 298M, as the Key to identify the data in the Value field. The term UL is used throughout this standard to refer to a SMPTE-administered Universal Label.

The full Key shall consist of a 16-byte field including an Object ID (0x06) and the UL size (0x0e indicating a total Key size of 16 bytes) followed by a series of sub-identifiers starting with the UL Code (0x2b) and SMPTE Designator (0x34). The sub-identifiers define the UL Designator (bytes 3~8 inclusive) and Item Designator (bytes 9~16 inclusive) as specified in Table 2.

The first two sub-identifiers after the SMPTE Designator shall have reserved values for the KLV Coding Protocol according to this standard.

Each word in the ANSI/SMPTE 298M UL is coded using ASN.1 Basic Encoding Rules (BER) for Object Identifier coding specified in ISO/IEC 8825-1, paragraph 8.19. (See Annex L.)

The values of each byte of the UL Designator shall be limited to the range 0x01 to 0x7F, which in BER Object Identifier coding are represented by a single byte.

The value of the Item Designator is coded using ASN.1 Basic Encoding Rules (BER) for Object Identifier coding and shall be 8 bytes in length.

The sub-identifiers in the UL Designator and Item Designator shall have left to right significance with the first sub-identifier as the most significant. The leftmost sub-identifier of value 0x00 in the Key shall define the termination of the label and all sub-identifiers of lower significance shall also be set to 0x00. Sub-identifiers of value 0x00 shall have no significance to the meaning of the Key.

ANSI/SMPTE 298M defines only the first four bytes of a UL: the Object ID, UL Size, UL Code and SMPTE Designator. This standard specifies the application of ANSI/SMPTE 298M ULs for the purpose of Key-Length-Value coding and defines the semantics of the remaining sub-identifiers of the UL Designator (bytes 5 to 8). The semantics of the Item Designator (bytes 9-16) are defined by a number of separate documents, which together cover all the defined values of the UL Designator.

Table 2 – Field Descriptions for the Key for the KLV Encoding of Data

No.	Field	Description	Length	Content/Format
UL Header				
1	OID	Object Identifier	1 byte	Always 0x06
2	UL Size	16-byte size of the UL	1 byte	Always 0x0E
UL Designator				
3	UL Code	Concatenated sub-identifiers ISO, ORG	1 byte	Always 0x2B
4	SMPTE Designator	SMPTE sub-identifier	1 byte	Always 0x34
5	Category Designator	Category designator identifying the category of registry described (e.g. Dictionaries)	1 byte	See Table 3
6	Registry Designator	Registry Designator identifying the specific register in a category (e.g. Metadata Dictionaries)	1 byte	See Table 3
7	Structure Designator	Designator of the structure variant within the given registry designator	1 byte	Section 4.1.3
8	Version Number	Version of the given register which first defines the item specified by the Item Designator	1 byte	Incrementing number
9 – 16	Item Designator	Unique identification of the particular item within the context of the UL Designator	8 bytes	See relevant standard and version

Annex C shows an informative example of a Metadata Key from the SMPTE Metadata Dictionary in tabular and figure formats.

Decoders which recognize the Key but do not want to, or cannot, decode the associated Value may ignore the item and should continue the decoding process of subsequent items using the Length value to 'skip' the Value of the un-decoded item. If decoders store and forward the item, they shall forward the item unaltered.

Bytes 5 and 6 of the Key shall be used to identify the contents of, and define the interpretation of the Value for all values of the Item Designator within a given category and registry designator. Table 3 defines the use of bytes 5 and 6. When bytes 5 and 6 do not match any of the values in Table 3, the parser shall not interpret the contents of the "V" bytes, shall make the K L and V available to application processing, and shall continue parsing with the byte immediately following the end of the "V".

NOTE – Application writers should be aware that public and private registers of SMPTE UL number spaces exist and these registers will contain valid KLV keys which may be unknown to the parser. Provision of application level interpretation of unrecognized KLVs keys is important for interoperability.

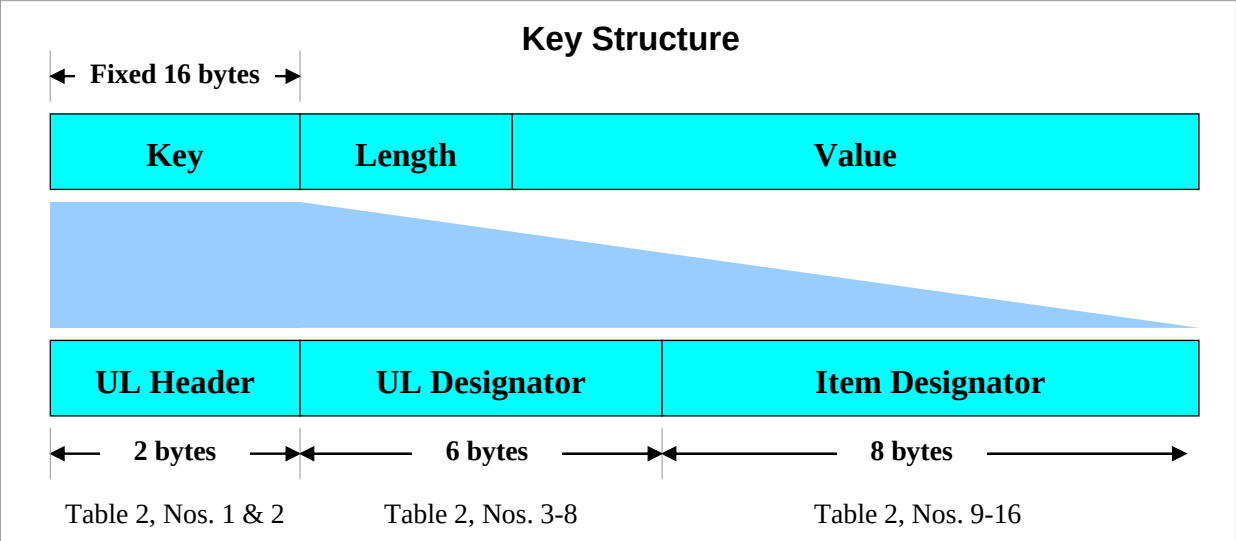


Figure 2 – Key Structure

4.1.1 UL Designator

Table 3 defines byte values for the designators to be used in bytes 5 to 7 of the UL Designator. SMPTE Standards and Recommended Practices (RPs) which define a Key with the value of byte 5 (Registry Category Designator) in the range 0x01 to 0x04 shall register the full Key or Keys used with the SMPTE Registration Authority in the registry identified by bytes 6 and 7 (Registry Designator and Structure Designator).

Table 3 – UL Designator for Bytes 5 through 7

Category Designator	Registry Designator	Defined In:	Structure Designator	External References (informative)
Byte 5	Byte 6	Byte 7		
0x01 – Dictionaries		Section 5		
	01 – Metadata Dictionaries	Section 5.1.1	0x01~0x7F	0x01: SMPTE 335M
	02 – Essence Dictionaries	Section 5.1.2	0x01~0x7F	
	03 – Control Dictionaries	Section 5.1.3	0x01~0x7F	
	04 – Types Dictionaries	Section 5.1.4	0x01~0x7F	0x01: Types Draft, CD2003
0x02 – Groups (Sets and Packs)		Section 6		SMPTE 395M
	01 – Universal Sets	Section 6.1, Table 4	0x01~0x7F	
	02 (default) – Global Sets	Section 6.2, Table 6	Section 6.2, Table 5	
	03 (default) – Local Sets	Section 6.3, Table 8	0x01~0x7F	
	04 (default) – Variable Length Packs	Section 6.4, Table 10	0x01~0x7F	
	05 – Defined-Length Packs	Section 6.5, Table 11	0x01~0x7F	
	06 – Reserved	Section 6.6	0x01~0x7F	
03 – Wrappers and Containers		Section 7		
	01 – Simple Wrappers and Containers	Section 7.1	0x01~0x7F	
	02 – Complex Wrappers and Containers	Section 7.2	0x01~0x7F	
04 – Labels		Section 8		
	Labels Register	Section 8	0x01~0x7F	0x01: SMPTE 400M
05 – Registered Private Information		Section 9		RP225
06 – 7E - Reserved				

4.1.1.1 Dictionaries

SMPTE Standards and RPs which define the value of word 5 of the Key as 0x01 are Dictionary standards and shall be used to define single data items with the KLV data construct.

The coding of individual data items is defined in Section 5.

4.1.1.2 Groups (Sets and Packs)

SMPTE Standards and RPs which define the value of word 5 of the Key as 0x02 are Set and Pack standards and shall be used to define groups of KLV coded data items.

The coding of data groups is defined in Section 6.

4.1.1.3 Wrappers and Containers

SMPTE Standards and RPs which define the value of word 5 of the Key as 0x03 are Wrapper and Container standards and use the Key to identify the Wrapper or Container and its contents. The definition of the terms 'Wrapper' and 'Container' are defined in Annex A.

The use of Wrappers and Containers is defined in Section 7.

4.1.1.4 Labels

The coding of Labels is defined in Section 8.

4.1.1.5 Registered Private Information

The coding of Registered Private Information is defined in Section 9.

4.1.1.6 Reserved

Reserved Registry Categories are reserved for future extension to this standard or for provisions defined by other SMPTE standards. No other specifications shall use these reserved values.

4.1.2 Registry Designator

As illustrated in Table 2 and Table 3, byte 6 shall define the Registry Designator identifying the specific register in a category (e.g., Metadata Dictionary). Global Sets, Local Sets and Variable Length Packs use more than one value to identify the lengths of the Length field and, in the case of Local Sets, the Local Tag field.

Usage of the Registry Designator values is defined in Table 3.

4.1.3 Structure Designator

As Illustrated in Table 2 and Table 3, byte 7 shall define the Structure Designator for the given register.

Structure Designator values are allocated to distinguish between incompatible versions of the same register. They may be thought of as a major version number.

Usage of the Structure Designator values is defined in Table 3.

4.1.4 Version Number

For items registered by SMPTE, byte 8 shall define the version number of the given register which first defines the item specified by the Item Designator.

For items not registered by SMPTE, the entity responsible for registration of items shall define its own version numbering policy.

NOTE – In any register, SMPTE may assign parts of the register to another organization whether as a public or private organization, or for other reasons make a part of the register an open space for undefined use. The policy of assigning these parts of the registry is not specified by this standard but by the individual registry standards. Typically, SMPTE will assign the first node that identifies the non-SMPTE organization and this node will be given a SMPTE version number. The values of the version number that lie under the SMPTE assigned node will then be assigned by the appropriate non-SMPTE organization, not by SMPTE.

New items may be added to registers after initial approval of the controlling Standard or RP. Each time a set of item definitions is added, the current Version Number of the particular register is incremented. Each entry in a register includes the version number in which the item was first defined. It is this number which is carried in byte 8.

Parsers may ignore the version number or use the version number as an additional guide and consistency check in the process of parsing a Key.

4.1.5 Item Designator

Bytes 9 through 16 of the Key comprise the Item Designator.

The Item Designator field shall be a fixed 8 bytes in length. Item Designator values are from 1 to 8 bytes long, padded on the right with zero bytes to fill the 8 byte field, and coded using ASN.1 BER Object Identifier coding as described in Section 4.1.

The precise meaning and construction of the Item Designator depends upon the specific register and Structure Designator value, and is described further in the sections below.

4.2 Encoding of the KLV Length Field

In the KLV coding protocol, the value of the Length field shall be encoded using the Basic Encoding Rules (BER) for either the short form or long form encoding of length bytes specified in ISO/IEC 8825-1, paragraph 8.1.3, 8.1.3.3 through 8.1.3.5 (see Annex K). This method of encoding the Length field is self-contained and allows for efficient parsing of KLV encoded data. When the KLV coding protocol is applied to groups of KLV coded units, the Length field for individual units may adopt a different method as defined by the standard for the coding of that group (see Section 6).

Where appropriate, individual application Standards and RPs may define the maximum byte-length of the Length field or may place limitations on the value range of the Length field in order to simplify decoder requirements.

NOTES

1 While there are no restrictions in this standard on the maximum number of bytes in the Length field, the presence of large Length fields can be determined from the first byte of the ASN.1 BER Long Form Length Encoding.

2 It is suggested that the short form of ASN.1 BER be used for all Value fields smaller or equal to 127 (0x7f).

Implementations shall make every effort to apply a valid value to the Length field. However, in certain operations it may prove impractical to establish the length of the Value field. Such a case is an incoming data stream which is assigned a Key and a Length field at the start point. In this case, the value of the Length field cannot be established until the termination of the stream and at that point, it may prove impossible to return to the Length field to enter the value. In such cases, the Length field shall be set to [0x80] which shall indicate a non-deterministic length of the Value field. Any application document which allows the length of the Value field to be undefined must define an alternative method of locating the end of the Value field.

NOTE – The length value [0x80] is used because it is normally meaningless as an ASN.1 BER long-form value as it indicates zero subsequent bytes.

4.3 Encoding of Data Values

Data values may be either individual data items or data groups. In either case, the data is a byte string whose length is specified by the Length field value. The last byte of the Value field shall be the terminating byte of the data sequence.

4.4 Empty Data Items

Specifications for contiguity of KLV packets including any gaps between KLV packets are outside the scope of this standard and are addressed in the appropriate transport-layer documents.

However, should applications so require, breaks in the data sequence may be inserted by the use of a specific “Empty” data item. Use of “Empty” data items is not mandatory.

The “Empty” data item is a KLV coded packet which shall define a length value followed by an empty Value field. No attempt should be made to interpret the data in the Value field.

“Empty” data items may be coded as individual items, or within sets when allowed by the specific set definition.

Applications may delete or skip any or all “Empty” data items upon receipt. Applications may insert “Empty” data items, but shall not require other applications to preserve such items.

The “Empty” data item shall be defined in the metadata dictionary and may be defined in other dictionaries.

NOTES

1 Section 4.1.1 provides guidance on the use of the version number byte. In the particular case of the empty data item, implementations have been deployed with different version values so it is recommended that decoders ignore the version number value.

2 The empty data item is widely known as the Fill Item.

5 KLV Coding of Individual Data Items

The KLV coding of individual data items is a simple application of the Key, Length and Value as defined in Section 4.

The Key of individual data items is defined in a register together with the ranges of Length and the specification of the Value itself. For individual data items the value of byte 5 of the Key shall be 0x01.

5.1 Registers of Individual Data Items

Individual data items that are KLV coded are collected into registers for the purpose of data management.

This standard defines four different registers as identified in Table 3. These registers are ‘dictionaries’ and they are identified by the value of byte 6 of the Key as follows:

5.1.1 Metadata Dictionaries

Metadata dictionaries shall be identified by byte 6 having the value of 0x01. Metadata dictionaries shall be registers of metadata items.

Metadata is information other than Essence, that has no inherent stand-alone value, but is related to Essence (i.e. it is contextual and has no meaning outside its relationship to the associated Essence). Examples of Metadata include: URL, URI, timecode, MPEG-2 PCR, filename, program labels, copyright information, version control, watermarking, conditional-access keys, etc.

5.1.2 Essence Dictionaries

Essence dictionaries shall be identified by byte 6 having the value of 0x02. Essence dictionaries shall be registers of essence items.

Essence is the data that represents pictures, sound and text. Types of Essence include Video, Audio and Data of various kinds, including captions, graphics, still images, text, enhancements and other data as needed by each application.

5.1.3 Control Dictionaries

Control dictionaries shall be identified by byte 6 having the value of 0x03. Control dictionaries shall be registers of control data items.

5.1.4 Types Dictionaries

Types dictionaries shall be identified by byte 6 having the value of 0x04. Types dictionaries shall be registers of data types.

Many register values share a common set of definitions for multiple data representations. To simplify the register definitions, the 'Types' Dictionary shall be used to define these data representations. The Types Dictionary shall be used as a shared resource for all other dictionaries.

5.2 Identification of Value Data Representations

The Value of many data items can be represented in more than one way. For example, a start time in the metadata dictionary can be represented as a character string of the timecode or as an efficient bit-packed form. The first offers direct mapping to a display whereas the second offers high transmission efficiency for use in narrow-band data channels. There are many such metadata dictionary items that have multiple data representations for the same descriptor.

Where a data item has more than one data representation for the value, one representation shall be designated as the default representation and shall be assigned a Key with at least one trailing zero byte. Alternate representations shall be assigned Keys by replacing the leftmost trailing zero byte with non-zero values, which shall be assigned sequentially. Each representation shall be documented as a separate entry in the register.

Informative Example:

01.02.03.04.00.00.00.00 is 'Name' (default data representation in 16-bit Unicode characters)

01.02.03.04.01.00.00.00 is 'Name' (different data representation in ISO 7-bit characters)

01.02.03.04.02.00.00.00 is 'Name' (another data representation in UTF-8 Unicode characters)

The parser treats all representations as the same data item, i.e. it recognizes 01.02.03.04.00, then looks for xx in place of the '00' to identify different encodings. Since the default representation is defined, the extra non-zero term in the 5th position is known to be a new data representation of the default register item. It should be noted that different representations may constrain the values that a data item may use.

Informative Annex D shows an example of KLV coding for a single metadata item.