

## 6.2.1.2 7-bit Alphabet Identifier (National Language Identifier)

### 6.2.1.2.1 Introduction

The 7-bit alphabet tables are used for representing the characters used in various regions around the world. They provide for more compact representation of text than using UCS2/UTF-16BE. For Europe, which mostly use the Latin script, the 0x10 alphabet should be used rather than UCS2. For Greek, the 0x11 alphabet should be used, rather than UCS2. Etc. for the additional alphabets defined. The default 7-bit alphabet (0x00, but usually not explicitly given) should be used only where explicitly required by the SMS/CBS standards, though should be the preferred alphabet for USSD. The tables come in triplets (Base, SS2, SS3) for each Natural Language Identifier (reference value) defined.

The principle is to use the National Language Identifier to indicate to a receiving entity that the message has been encoded using a national language table. Both single shift and locking shift mechanisms are defined, though setting either to a National Language Identifier (alphabet reference number) to a value 0x10 or higher automatically sets all three tables (Base, SS2, SS3). If both are set, they must be the same. (If set to a lower value than 0x10, the setting are separate and both must be to values less than 0x10. Not setting either, automatically sets the default alphabet (National Language Identifier 0x00) for both Base (Locking shift) and SS2 Single Shift. Alphabets with National Language Identifier (reference value) less than 0x10 do not have an SS3 table (or equivalently, an empty SS3 table).

The setting of the base table applies throughout the message, or the current segment (submessage) in case of a concatenated message, and it sets the Alphabet Base Table (see subclauses A.3 and A.4) that defines the base character set. A base character need one 7-bit code unit.

The setting of the single shift tables applies throughout the message, or the current segment (submessage) in case of a concatenated message, and it sets the Alphabet Extension Tables (see subclauses A.2 and A.4). There are two single shift tables, one for SS2, and one for SS3. An SS2 based character requires two 7-bit code units, the SS2 (0x1B) and one more, and SS3 based character requires three 7-bit code units, SS2, SS3, and then one more.

In case that several languages are used, which require different sets of Alphabet tables, it is recommended to encode the message in UCS-2. However, it is possible to use different National Language Identifiers of value 0x10 or greater, as well as UCS2 (UTF-16BE) in different submessages (segments) of the same full message (but not Alphabet Identifiers of value 0x00 (default alphabet) to 0x0F; using those, all segments must use the same Alphabet Identifiers). The reason for that is that for those Alphabets, segment cuts can be in SS2

If an Alphabet Identifier is set, for either or both of the locking shift or single shift, that sets all three tables to those for that Alphabet Identifier. If both are set, it must be to the same (non-default) value.

### 6.2.1.2.2 Base table use

A 7-bit code unit (that is not after an SS2 or SS3) is looked up in the current base table (whether the default or set via an Alphabet Identifier). If the result is not the control character SS2 (always as 0x1B of the base table) then the character denoted is the one in that position in the base table. If the lookup gives SS2, see section 6.2.1.2.3. If the result of the lookup is an empty (or explicitly undefined) position of the table, the result is the character REPLACEMENT CHARACTER, U+FFFD. If REPLACEMENT CHARACTER is not representable in the target encoding, use instead SUBSTITUTE, U+001A, with a visible spacing glyph in display; this character is available in most legacy character sets.

### 6.2.1.2.3 Extension (SS2 and SS3) table use

If the base table lookup gives the control character SS2, then use the follow-on 7-bit code unit to look up in the currently set (whether default or set via an Alphabet Identifier) SS2 table. If that is not the control character SS3 (always at position 0x1B of the SS2 table), then the character denoted is the one in that position in the SS2 table. If the lookup gives SS3, then use the follow-on 7-bit code unit to look up in the currently set SS3 table. The character denoted is the one in that position in the SS3 table. An omitted SS3 table is the same as an empty SS3 table. If the result of the lookup is an empty (or explicitly undefined) position of the table, the result is the character REPLACEMENT CHARACTER, U+FFFD (or SUBSTITUTE, U+001A, as described in 6.2.1.2.2).

#### 6.2.1.2.4 List of 7-bit Alphabet Identifiers

A National Language Single Shift IE and a National Language Locking Shift IE can be included in the TP User Data Header, as defined in 3GPP TS 23.040 [4]. The receiving entity shall set the three character tables (base, SS2, SS3) according to the given IE, if the receiving entity supports that IE. If both are set, they must have the same value. If the receiving entity does not support the given IE, it shall use the 7-bit default alphabet.

The National Language Identifier octet is encoded as shown in table 6.2.1.2.4.1.

**Table 6.2.1.2.4.1**

<b>7-bit Alphabet Identifier <math>b_7 \dots b_0</math> (0xh<sub>1</sub>h<sub>0</sub>)</b>	<b>7-bit Alphabet</b> <i>Even though referred to by a language name, the alphabet can often be used for other languages (geographically near).</i>	<b>Base Table</b>	<b>Single Shift Tables (SS2 and SS3)</b> <i>For 0x00 to 0x0D the SS3 tables are empty/absent.</i>
00000000 (0x00) <i>(cannot be explicitly set)</i>	Default 7-bit alphabet (Latin and uppercase Greek) (strongly deprecated, use 0x10 instead; 0x11 for Greek)	Subclause 6.2.1.1 (Greek reading is in subclause 6.2.1.3, strongly deprecated)	Subclause 6.2.1.2
00000001 (0x01)	Turkish (strongly deprecated, use 0x10 instead)	Subclause A.3.1	Subclause A.2.1
00000010 (0x02)	Spanish (strongly deprecated, use 0x10 instead)	Subclause A.3.2 (referencing subclause 6.2.1.1)	Subclause A.2.2
00000011 (0x03)	Portuguese (strongly deprecated, use 0x10 instead)	Subclause A.3.3	Subclause A.2.3
00000100 (0x04)	Bengali (strongly deprecated, use 0x14 instead)	Subclause A.3.4	Subclause A.2.4
00000101 (0x05)	Gujarati (strongly deprecated, use 0x15 instead)	Subclause A.3.5	Subclause A.2.5
00000110 (0x06)	Hindi (strongly deprecated, use 0x16 instead)	Subclause A.3.6	Subclause A.2.6
00000111 (0x07)	Kannada (strongly deprecated, use 0x17 instead)	Subclause A.3.7	Subclause A.2.7
00001000 (0x08)	Malayalam (strongly deprecated, use 0x18 instead)	Subclause A.3.8	Subclause A.2.8

<b>7-bit Alphabet Identifier <math>b_7 \dots b_0</math> (0x<math>h_1 h_0</math>)</b>	<b>7-bit Alphabet</b> <i>Even though referred to by a language name, the alphabet can often be used for other languages (geographically near).</i>	<b>Base Table</b>	<b>Single Shift Tables (SS2 and SS3)</b> <i>For 0x00 to 0x0D the SS3 tables are empty/absent.</i>
00001001 (0x09)	Oriya (strongly deprecated, use 0x19 instead)	Subclause A.3.9	Subclause A.2.9
00001010 (0x0A)	Punjabi (strongly deprecated, use 0x1A instead)	Subclause A.3.10	Subclause A.2.10
00001011 (0x0B)	Tamil (strongly deprecated, use 0x1B instead)	Subclause A.3.11	Subclause A.2.11
00001100 (0x0C)	Telugu (strongly deprecated, use 0x1C instead)	Subclause A.3.12	Subclause A.2.12
00001101 (0x0D)	Urdu (strongly deprecated, use 0x13 instead)	Subclause A.3.13	Subclause A.2.13
00001110 (0x0E) – 00001111 (0x0F)	Reserved	n/a	n/a
00010000 (0x10)	European (Latin script)	Subclause A.4.10	
00010001 (0x11)	Greek	Subclause A.4.11	
00010010 (0x12)	Hebrew	Subclause A.4.12	
00010011 (0x13)	Urdu (Eastern Arabic script)	Subclause A.4.13	
00010100 (0x14)	Bengali/Bangla	Subclause A.4.14	
00010101 (0x15)	Gujarati	Subclause A.4.15	
00010110 (0x16)	Hindi (Devanagari script)	Subclause A.4.16	
00010111 (0x17)	Kannada	Subclause A.4.17	
00011000 (0x18)	Malayalam	Subclause A.4.18	
00011001 (0x19)	Oriya/Odia	Subclause A.4.19	
00011010 (0x1A)	Punjabi (Gurmukhi script)	Subclause A.4.1A	
00011011 (0x1B)	Tamil	Subclause A.4.1B	
00011100 (0x1C)	Telugu	Subclause A.4.1C	

<b>7-bit Alphabet Identifier <math>b_7 \dots b_0</math> (0x<math>h_1 h_0</math>)</b>	<b>7-bit Alphabet</b> <i>Even though referred to by a language name, the alphabet can often be used for other languages (geographically near).</i>	<b>Base Table</b>	<b>Single Shift Tables (SS2 and SS3)</b> <i>For 0x00 to 0x0D the SS3 tables are empty/absent.</i>
00011101 (0x1D)	Thai	Subclause A.4.1D	
00011110 (0x1E)	Lao	Subclause A.4.1E	
00011111 (0x1F)	Khmer	Subclause A.4.1F	
00100000 (0x20)	Manipuri	Subclause A.4.20	
00100001 (0x21)	Sinhala	Subclause A.4.21	
00100010 (0x22)	Armenian	Subclause A.4.22	
00100011 (0x23)	Georgian	Subclause A.4.23	
00100100 (0x24)	Ukrainian (Cyrillic script)	Subclause A.4.24	
00100101 (0x25) – 11111111 (0xFF)	Reserved	n/a	

### 6.2.1.2.5 Processing of national language characters

When supporting a specific 7-bit alphabet, the sending entity shall support the encoding of messages using the corresponding Alphabet Identifier defined in subclause 6.2.1.2.4.

The receiving entity should be able to decode messages using the Alphabet Identifiers defined in subclause 6.2.1.2.4 for the 7-bit alphabets that are supported by that entity.

If a message is received, containing an Alphabet Identifier indicating a reserved value (or 0x00, which is for the default alphabet, but cannot be explicitly set) or a value that is not supported by the receiving entity, the receiving entity shall ignore the IE (see 3GPP TS 23.040 [4]) in which the Alphabet Identifier was indicated.

The receiving entity shall be capable of processing 7-bit code units referring to both the base table and the SS2 and SS3 (single shift) tables within a message/segment.

NOTE 1: A message using a Alphabet Identifier not supported by the receiving entity may not be properly displayed by the receiving entity.

NOTE 2: Several Alphabets (when using characters other than A-Za-z0-9 and simple punctuation, which are in all of the alphabets) requires complex script handling for proper display of messages. That handling varies depending on Alphabet. For instance, bidi processing, handling and positioning of combining mark, and much more. Giving details on this goes way beyond this standard, and we refer to the Unicode standard, which does describe some of that script dependent display handling.

NOTE 3: The Alphabets (will have)/has machine readable mapping tables (to/from Unicode) at <https://.....>