X3.2/647
X3.2.4/252
February 7, 1968
Revision of X3.2/622

PROPOSED

USA STANDARD

CODE EXTENSION TECHNIQUES

FOR INFORMATION INTERCHANGE

X3.2/647
X3.2.4/252
February 7, 1968

FOREWORD

(This foreword is not a part of the USA Standard Code Extension Procedures for Information Interchange, X3....)

The USA Standard Code for Information Interchange (ASCII - USASI X3.4-19) provides coded representations for a set of graphic and control characters having general utility in information interchange. In some applications, it may be desirable to augment the standard repertoire of characters with additional graphic symbols or control functions.

The Code includes several special characters intended to facilitate the representation of such additional symbols or functions, a process known as code extension. Although the basic nature of code extension - providing for encoding of information beyond the standard - limits the degree to which it may be standardized, there are advantages to adherence to certain standard rules of procedure. These advantages include: minimized risk of conflict between systems required to interoperate, and the possibility of including advance provision for code extension in the design of general purpose data handling systems.

These standard procedures were developed after extensive study of various potential applications and of trends expected in system design.

This standard was approved as a USA Standard by the United States of America Standards Institute on

Suggestions for improvement gained in the standard will be welcomed. They should be sent to the United States of America Standards Institute, 10 East 40th Street, New York 10016.

USASI Subcommittee X3.2, Codes and Input-Output, through which the standard was developed and processed, had the following membership at the time of approval:

L. L. Griffin - Chairman

R. Adams
G. C. Arndt
E. A. Avakian
J. B. Booth
R. M. Brown
N. Clark
M. Cohen
R. J. Donahue
S. Erdreich
H. L. Hill
T. O. Holtey
H. R. Hoots
R. M. Ireland
D. A. Kerr
R. E. Lows
W. G. Leary
J. L. Little
N. H. Locke
W. H. McKenzie
M. Mendelsohn
G. L. Murphy
J. Murphy
F. G. Smith
F. W. Smith
J. L. Smith
H. Spielman
G. E. Williams

The following members of USASI Task Group X3.2.4, Code Development, participated in the development of this standard:

J. L. Little, Chairman

E. A. Avakian
J. B. Booth
F. D. Biggam
E. H. Clamons
N. Clark
M. Cohen
W. S. Crosby
R. J. Donahue
S. Erdreich
S. Grunkorn
H. R. Hoots
H. F. Ickes
D. A. Kerr
C. N. Mooers
J. K. Nelson
H. Spielman
E. F. Vidro, Jr.
G. E. Williams

It should be recognized that although X3.2 and X3.2.4 members are variously affiliated, work on a USASI Subcommittee or Working Group is achieved primarily on an individual and experience basis.

Contents

Development, participated in the development of this standard:

J. L. Little, Chairman

E. A. Avakian	S. Grunkorn
J. B. Booth	H. R. Hoots
F. D. Biggam	H. F. Ickes
E. H. Clamons	D. A. Kerr
N. Clark	C. N. Mooers
M. Cohen	J. K. Nelson
W. S. Crosby	H. Spielman
R. J. Donahue	E. F. Vidro, Jr.
S. Erdreich	G. E. Williams

It should be recognized that although X3.2 and X3.2.4 members are variously affiliated, work on a USASI Subcommittee or Working Group is achieved primarily on an individual and experience basis.

X3.2/647
X3.2.4/252
February 7, 1968

USA STANDARD

CODE EXTENSION PROCEDURES

FOR INFORMATION INTERCHANGE

1. Scope

This standard specifies a set of procedures for the representation, by characters of ASCII*, of graphic symbols or control functions, not directly represented in ASCII, which may be required for a specific application or system. This standard does not make specific assignment of such characters or functions.

* USA Standard Code for Information Interchange

2. General

2.1 The following characters were included in ASCII specifically for code extension purposes:

SO	Shift Out
SI	Shift In
ESC	Escape
DLE	Data Link Escape

2.2 SO and SI were included in ASCII to provide an efficient means of signaling excursions between the standard (ASCII) character set and an alternate set. The SO signals an excursion to the alternate set; the SI a return to the standard (ASCII) set. It should be noted that the SO does not define the alternate set, only the excursion. The standard procedures for the use of SO and SI are described in Section 3.

2.3 ESC was included in ASCII to provide an efficient means of representing, on an individual basis, characters not directly represented in the standard code. Standard procedures for the use of ESC are described in Section 4.

2.4 Interchange of data including the use of these techniques requires agreement between sender and recipient on specific character assignments.

2.5 DLE is intended for use in representing communication link control functions not directly represented in the standard code. Standard procedures for its use are to be covered in standards for data communication control procedures (See Appendix section A5).

2.6 The promulgation of these standard procedures is in no way meant to deprecate the use of other code extension procedures, so long as the implications of such usage upon systems compatibility are recognized. (See also Appendix section A2.4).

3. Use of SO (Shift Out) and SI (Shift In)

3.1 The characters SO and SI are used to identify or select which of two sets of characters is to be associated with the bit patterns of the standard code.

3.2 SO indicates that the bit patterns which follow are to be identified with the characters in an alternate set.

3.3 SI indicates that the bit patterns which follow are to be identified with the characters in the standard set.

3.4 There is no implication that all characters of the alternate character set be different from the corresponding members of the standard set.

3.5 Care should be used when assigning code positions to characters in such an alternate set where these code positions represent control characters in the standard set (See Appendix A3.1.4).

3.6 SI shall appear in any such alternate set and shall occupy the same character position there as in the standard set.

3.7 If more than one alternate set is required to coexist in a given system, escape sequences (see section 4) may be assigned to identify which of the defined alternate sets is to be subsequently put into force by the use of SO. In all cases, SI will signal a return to the standard.

4. Use of ESC (Escape)

4.1 The character ESC (Escape) is used as a prefix to a sequence of one or more additional characters used to represent a character not directly represented in the standard code.

4.2 An escape sequence is considered to include its associated ESC character, and its length is described accordingly.

4.3 Such sequences - known as "escape sequences" - should not be used to represent additional communication control functions. (See Appendix section A5)

4.4 This standard provides a uniform method for the definition of escape sequences of any length (two characters or greater).

4.5 The means of marking the end of a sequence depends upon division of the characters of the code into two classes, known as "intermediate" and "final" characters, respectively.

4.6 A standard variable length escape sequence begins with ESC, continues, if necessary, with any number of "intermediate" characters, and invariably ends with one "final" character. Two-character sequences, therefore, contain no "intermediate" characters, but consist of ESC followed by one "final" character.

4.7 The intermediate characters are those in column 2 of the ASCII code table, that is, space and certain special graphics.

4.8 The final characters are those in column 0, 1, and 3 through 7 of the code table, that is, the control characters, the alphabetics, the numerics and several of the special graphics, except as noted in section 4.9.

4.9 The following characters should be excluded from assignment in escape sequences: (See Appendix section A4.3)

Designation	Name	Code Table Column/Row
NUL	Null	0/0
SOH	Start of Heading	0/1
STX	Start of Text	0/2
ETX	End of Text	0/3
EOT	End of Transmission	0/4
ENQ	Enquiry	0/5
ACK	Acknowledge	0/6
DLE	Data Link Escape	1/0
NAK	Negative Acknowledge	1/5
SYN	Synchronous Idle	1/6
ETB	End of Transmission Block	1/7
CAN	Cancel	1/8
SUB	Substitute	1/10
ESC	*Escape	1/11
DEL	Delete	7/15

* Except as first character

APPENDIX

(This appendix is not a part of the USA Standard Code Extension Procedures for Information Interchange, but is included to facilitate its use.)

Contents

A1. Introduction

This appendix to the USA Standard Code Extension Procedures for Information Interchange contains a discussion of the objectives, criteria, and other considerations that were used in the development of the standard, as well as supplementary information to facilitate the effective application of these procedures.

A2. General

A2.1 Background

In the establishment of a general purpose code such as the USA Standard Code for Information Interchange (ASCII), or its international counterpart, the ISO 7-bit code, a fundamental decision must be made as to the size of the code. In making such a decision there is usually a conscious effort to avoid the most obvious problems with a code which is either too large or too small. Should the number of characters included be too small, many individual users will find their needs not accommodated, and will be forced to adopt "parochial" codes for their applications. Should the number of characters be too large, many potential users will find the standard code disproportionately costly to implement, or untenably inefficient in transmission or storage, and will again be driven to the use of some other code. Thus, either extreme in code sizing will reduce the generality of application of the code, defeating the very purpose of standardization in this field.

The 7-bit size (128 characters) adopted for ASCII is thought to be near optimum at present with respect to the above considerations. Nevertheless, there will doubtlessly be numerous applications with requirements that are not accommodated by a code of this size, or at least not by the specific characters assigned within it. Still, it is hoped that many of these applications can be served by the use of the standard code augmented in some appropriate manner. Through such an approach, the user may be able to implement much of his system with standard hardware or software. More significantly, perhaps, he will thereby be able to retain compatibility with other systems for the interchange of that information which can adequately be directly represented by the standard code.

The concept of augmenting the standard code for such purposes may be spoken of in a generic way as "code extension".

A2.2 Standardization of Procedures

The codes with which we are concerned contain four characters whose definitions indicate their relationship to code extension. They are:

SO	(Shift Out)
SI	(Shift In)
DLE	(Data Link Escape)
ESC	(Escape)

The use of these characters is not treated in detail in the code standards. Actually, the very nature of code extension inherently limits the degree to which standards for it may be constructed: it is a means of operating "beyond the standard". Nevertheless, there are several advantages to establishing a standard general procedure.

First, such standardization can prevent undesirable conflict between independently contrived applications of code extension. For example, a code extension procedure used by a data communication terminal device should be inherently free from any hazard of conflict with a code extension procedure used in a communications system which may be called upon to serve the terminal.

Second, the availability of such standards can provide guidance to system designers to facilitate the advance inclusion of general provisions for code extension operations in information handling equipment.

A2.3 Application of Standard Procedures

The standard procedures are directed at the application of code extension to those portions of a system where the use of the standard code itself would ordinarily be appropriate; that is, in what is spoken of as "information interchange".

A2.4 Related Approaches

The suggested procedures presented here for code extension should in no way be considered to deprecate the practice of using sequences of graphic characters to represent machine instructions, graphic characters not otherwise available, and so forth. Programming languages used in data processing, for example, are based upon such an approach.

A2.5 ASCII

Page 19 shows the USA Standard Code for Information Interchange (per USAS X3.4-1967) and is provided for reference. The code consists of two general categories of characters, graphics and controls. There are 32 controls, 95 graphics, and the character DEL (Delete) which in reality is neither. The 95 graphics include both upper and lower cases of the Latin (often called "Roman") alphabet, the Arabic numerals 0 to 9, a number of punctuation marks and special symbols, and SP (space), the "nonprinting graphic".

A3. Use of SO and SI

A3.1 Basic Concepts

A3.1.1 There are applications which require more characters than are provided in the ASCII. Examples are scientific texts, expressions employing an extended set of mathematical symbols, and languages which cannot be represented directly by the Latin alphabet. The code extension procedures provide a means of accommodating additional characters over and above those provided in the ASCII without sacrificing or substituting for those that are provided in the standard.

A3.1.2 As pointed out in section 3.4, there is no implication that the alternate set should be entirely different from the standard set, or that all positions are even assigned. The alternate set may contain whatever repertoire of characters are needed for operation in a particular environment. It is recommended that any symbols common to both the standard and alternate sets be assigned to the same code table position (bit pattern) in the alternate set as in the standard set. The character SI must be assigned in its usual position to assure a standard means of return to the standard character set.

A3.1.3 It is recommended that terminal devices and other such equipment be arranged to automatically revert to the SI condition whenever the association of the terminal with another terminal or system has been discontinued or suspended: that is, at the end of a call, transmission or whatever is appropriate.

A3.1.4 Section 3.5 indicates that care should be used when assigning code positions to characters in an alternate set when these code positions represent control characters in the standard set. This warning principally relates to the possibility that communication systems or data terminals used to interchange such data may be sensitive to the bit patterns of these positions.

A3.2 Application to Devices of Modest Graphic Repertoire

A3.2.1 It should be noted that useful application of these principles may in some cases be made in devices having a relatively modest capacity for different graphics. Consider, as an example, the problem of making a terminal device to render messages in both Latin (standard) and Greek (alternate) alphabets, and requiring the conventional numerals and punctuation in connection with either. In many situations it would be satisfactory to render all letters in the upper case: that is, the receipt of the coded representation for either "A" or "a" would cause "A" to be printed*. Extending this principle to the special symbols coded in the same area of the code with the letters, it is seen that 32 printing characters can suffice for 64 characters of the code. (Actually 63, since DEL, though coded in the graphic region, is not a graphic.) Adding provision for the 10 numerals and the 22 remaining symbols, the machine need have but a 64-character graphic capacity for its work in the Latin alphabet.

A3.2.2 An additional 31 printing characters can serve, in the same manner, for both the upper- and lower case Greek alphabets and some associated special symbols when in the alternate set. The 10 standard numerals and the 22 standard punctuation marks are used in the alternate set operation. This postulated application can therefore be implemented in this manner with a terminal device having only the 95-character graphic capacity which would ordinarily be required for full rendition of the standard set.

---

* This technique is already widely in use where 64-graphic printers are used in systems which utilize all 95 graphic characters.

A3.2.3 Such a device when in its standard mode may receive, without hazard, information containing any of the 95 ASCII graphics. If a graphic set shift were not used in this application, the bilingual capability could only be served with a 95-graphic printer by making the Greek alphabet a graphic substitution for the lowercase Latin letters in the code table. The device could not then be safely used for interchange of information with systems which might use the lower case Latin letters, since the receipt of these would of course cause the printing of Greek letters.

A4. Use of ESC (Escape) Sequences

A4.1 Sequence Length

A4.1.1 In order that an Escape sequence may invariably be identifiable as such, each such sequence begins with the prefix character ESC (Escape), which has no other use.

A4.1.2 It was at one time proposed that code extension sequences should be standardized as always consisting of ESC and a single-following character. While this would be adequate for many applications, there are a number of considerations which may make longer sequences desirable in many cases. One such consideration is just that of having an adequate number of sequences available for the functions required in one system, or in a number of systems requiring nonconflicting function representations. Another consideration is that it is sometimes desirable to represent a critical function by a long sequence to gain security against accidental or malicious operation. A third consideration is the desire, in some systems, to have a mnemonic relationship between the character sequence and the designation of the function to be controlled.

A4.1.3 In many systems it is very useful to have a doctrine which allows sequences of various lengths to coexist in the same system.

Paramount among the requirements for a variable-length doctrine is the need to have a simple means for a device to determine the end of each sequence which it receives: that is, how many of the

characters following ESC are associated with it. This is necessary so that the device may avoid giving the normal interpretation to individual characters of a code extension sequence, even when the specific sequence is not to be recognized and acted upon.

A4.1.4 The procedures of section 4 provide this flexibility without requiring the use of an "ending" character in each sequence, which carries no other information.

A4.2 Partition of the Code

A4.2.1 There are a number of criteria which affected the way in which the characters of the code was divided into "intermediate" and "final" groups. Among the significant ones were:

1. "Intermediates" should be distinguishable from "finals" by a simple logical test, preferably by the sense of 1 bit in the coded representation.

2. A given class of character, such as alphabetic, numeric, etc., should be entirely within one group.

3. Upper- and lower-cases of any specific alphabetic character should be in the same group. This allows a system designer to assign sequences so that no distinction is made on the basis of case, if desired.

4. A number of 2-character (i.e., ESC-plus-one-"final") sequences should be available which use only letters or numerals, because such sequences are convenient for use by humans.

5. The "final" group should contain some characters which are likely to occur with reasonable frequency in a stream of data. This ensures that, should the legitimate final character of a sequence be lost or mutilated, the system will soon be restored to its normal mode of character interpretation.

A4.2.2 These criteria led to partition of the code into "intermediate" and "final" characters is as follows: (see section 3.6)

Columns 0, 1, and 3 through 7 of the code table contain final characters. $(b_7, b_6, b_5) \neq (0, 1, 0)$

Column 2 of the code table contains intermediate characters $(b_7, b_6, b_5) = (0, 1, 0)$

(See A4.3 below for restrictions)

This partition is felt to produce the most useful balance in the degree to which the criteria are satisfied.

A4.3 Restrictions

The restrictions of section 4.9 were imposed in order to avoid certain potentially serious problems.

A4.3.1 The ten communication control characters should never be used in Escape sequences. Such use could cause interference with the control logic of communication systems through which the data may be passed, unless the systems were arranged to detect the sequences and determine their lengths, which would be an unnecessary burden. These ten characters are:

Designation	Name	Code Table Column/Row
SOH	Start of Heading	0/1
STX	Start of Text	0/2
ETX	End of Text	0/3
EOT	End of Transmission	0/4
ENQ	Enquiry	0/5
ACK	Acknowledge	0/6
DLE	Data Link Escape	1/0
NAK	Negative Acknowledge	1/5
SYN	Synchronous Idle	1/6
ETB	End of Transmission Block	1/7

Also, additional communication controls should not be represented by ESC sequences, but rather by DLE sequences, as described in section A5.

A4.3.2 The following characters also should not be assigned in Escape sequences:

Designation	Name	Code Table Column/Row
NUL	Null	0/0
CAN	Cancel	1/8
SUB	Substitute	1/10
ESC	Escape*	1/11
DEL	Delete	7/15

A4.3.2.1 NUL is excluded due to the hazards associated with the lack of clearly established conventions for its use and because some systems may be unable to process this character.

A4.3.2.2 CAN is excluded since its purpose is to "cancel" a portion of the data, and may thus appear abruptly in a stream of data and may even be used to "cancel" an Escape sequence.

A4.3.2.3 SUB is similarly excluded because it may be used to replace a character determined to be in error, and may thus unpredictably appear in an escape sequence as a result of this process.

A4.3.2.4 ESC is excluded to avoid confrontation with the paradox created by its definition as a "final" character: after the first ESC of a sequence another would mean at once that the sequence was starting and ending. It therefore seems better to avoid this problem than to become dependent upon specific resolutions of it in equipment.

A4.3.2.5 Finally, DEL is excluded because in some systems it may unpredictably appear as a result of correction of operator errors in perforated tape, and because some portions of a system may "delete" this character from the data stream.

A4.3.3 The use of the remaining control characters in any ESC sequence should be avoided whenever possible, due to the effects which they may cause if the ESC is lost or mutilated or is not recognized for some other reason. (see also A4.4)

---

* Except, of course, as the first character

A4.4 Anomalies

A sequence should always be considered ended if a final character is recognized, whether or not the sequence is recognized and regardless of whether the final character is an "allowable" one in escape sequences. If that final character is ESC it should preferably be considered to start a second sequence.

A5. Code Extension for Communication Controls: Use of DLE (Data Link Escape)

Standardization of specific procedures for the use of DLE falls within the jurisdiction of USASI Task Group X3.3.4, Communication Control Procedures. The subject is discussed here only to show the relationship of this use to other aspects of code extension.

It is necessary for a communication system to be able to readily distinguish between the communication controls which are of concern to it and other controls with which it is not concerned. The assignment of specific communication control characters in the code provides this distinction under ordinary circumstances. It is necessary that this distinction be preserved when additional controls of one type or the other are represented by escape sequences. The character DLE (Data Link Escape) is provided for use in lieu of ESC as the first character of sequences used to represent additional communication controls. Thus, communication link control logic may ignore ESC entirely, passing it and the characters which follow as any other "text" characters. Code extension sequences of concern to the communication control logic can invariably begin with DLE, to which the logic may be made sensitive.

The prohibition previously expressed against the use of communication control characters in ESC sequences is intended to prevent direct interference with the communication control logic.

USA STANDARD CODE FOR INFORMATION INTERCHANGE
USAS X3.4-1967

b_7 b_6 b_5 →					0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
Bits	b_4 ↓	b_3 ↓	b_2 ↓	b_1 ↓	Row ↓ Column →	0	1	2	3	4	5	6	7
	0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
	0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
	0	0	1	0	2	STX	DC2	"	2	B	R	b	r
	0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
	0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
	0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
	0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
	0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
	1	0	0	0	8	BS	CAN	(	8	H	X	h	x
	1	0	0	1	9	HT	EM	)	9	I	Y	i	y
	1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
	1	0	1	1	11	VT	ESC	+	;	K	[	k	{
	1	1	0	0	12	FF	FS	,	<	L	\	l	|
	1	1	0	1	13	CR	GS	-	=	M	]	m	}
	1	1	1	0	14	SO	RS	.	>	N	^	n	~
	1	1	1	1	15	SI	US	/	?	O	_	o	DEL