XEROX

Xerox System Integration
Standard

Character Code Standard

XEROX

Xerox System Integration
Standard

Character Code Standard

XSIS 058404
April 1984

Xerox Corporation
Stamford, Connecticut 06904

Notice

This *Xerox System Integration Standard* describes the Character Code Standard—the assignment of numeric codes to all commonly-used characters on a fully international basis and the forms in which sequences of them can be represented.

1. This standard includes subject matter relating to patent(s) of Xerox Corporation. No license under such patent(s) is granted by implication, estoppel, or otherwise, as a result of publication of this specification.

2. This standard is furnished for informational purposes only. Xerox does not warrant or represent that this standard or any products made in conformance with it will work in the intended manner or be compatible with other products in a network system. Xerox does not assume any responsibility or liability for any errors or inaccuracies that this document may contain, nor have any liabilities or obligations for any damages, including but not limited to special, indirect, or consequential damages, arising out of or in connection with the use of this document in any way.

3. No representations or warranties are made that this specification, or anything made in accordance with it, is or will be free of any proprietary rights of third parties.

©Copyright 1984 Xerox Corporation.
All rights reserved.

XEROX®, Xerox Network Systems, and NS
are trademarks of XEROX CORPORATION.

Preface

This document is one of a family of publications that collectively describe the protocols underlying Xerox Systems—Printing Systems and Network Systems.

This character code standard is the Xerox corporate character code standard. It supersedes all previous Xerox character code standards including Character Code Standard XSIS 058402, Character Encoding Standard XSIS 058305, and the NS Character Set Specification, October 1981.

The Character Code Standard specifies the character codes to be used for exchange of text information among Xerox System Elements, and the forms in which sequences of numerical codes can be represented. Fragments of text which are referred to as *strings*—and a *Xerox string* means a string consisting of character codes defined by this standard—are used for such purposes as communication protocols, electronic print files, and document interchange.

This document assigns codes to a set of graphic characters, rendering characters, and control characters; defines and explains the character sequence encoding; gives examples of its use; addresses a number of technical questions concerning the details of the Xerox character code assignments; and provides a cross reference of commonly used text characters and symbols. The primary purpose of this document, however, is to provide an accurate specification of character codes and the encoding of a string of these codes.

Additions to this standard will be made and the string "XCa-b-c-d" will identify the current version of the standard. "a" represents the present version number for the standard, and its numeric value would change only in the event of a radical revision of the standard. "b" refers to the graphic character codes, and will increase each time graphic characters are added to the graphic character code space; "c" refers to the rendering character codes, and will increase each time rendering characters are added to the rendering character code space; and "d" refers to the control character codes, and will increase each time control characters are added to the control character code space. In the event of an increase in "a," the values for "b", "c", and "d" will be reset to zero. All numbers are integers (base 10) without leading zeros. The present version of this standard has the "IDENTITY" ***XC1-1-1-0***.

Comments and suggestions on this document and its use are encouraged. Please address communications to:

Xerox Corporation
Printing Systems Group
Printing Systems Administration Office
701 South Aviation Boulevard
El Segundo, California 90245

Table of contents

Figures

1

Introduction

1.1 Purpose

The purpose of this standard is to permit multilingual textual information to be stored and transmitted in the form of a sequence of numerical codes. The numerical codes are called *character codes*, and a sequence of them forms the body of what is called a *string*. When two information-processing systems agree on a standard interpretation of character codes and a standard format for strings, they may communicate text without danger of degrading its information content.

The particular standard presented here is used throughout the Xerox Systems. It is a generalization of familiar ISO and ANSI standards for coded character sets for text communication, and includes numerical codes for rendering entities—non-graphic character codes which may be used in fragments of text.

In this standard a character code is any code representing a graphic character, a rendering character, or a control character, usually unique within the set, most commonly represented as a non-negative integer. Xerox' generalization of current standards to create this two-byte character code standard is motivated by the desire to handle symbols beyond simple punctuation, and text in languages other than English.

The range of numbers available for character codes is taken to be [0 .. 177777_8], i.e., 16 bits. A general method for extending this to 24-bits and to 32-bits is outlined but not required at this time. The allocation of code space is specified in Section 2, and the assignment of numbers to graphic characters, rendering characters, and control characters is specified in Sections 3 through 5. All assignments are summarized in the reference charts in Appendix B.

Valid methods of representing a sequence of 16-bit character codes are described in Section 6, String encoding. This format for strings is one solution to the problem of compatibility with existing 8-bit character strings.

1.2 Scope

This Xerox standard assigns codes to a set of graphic characters spanning at least the following:

- All ISO 646 IRV graphic characters [5]
- All ISO 5426 graphic characters [8]
- All ISO 5428 graphic characters [9]
- All ISO 6937 graphic characters [10]
- All ANSI 7-bit ASCII graphic characters [1]
- All CCITT 8-bit Teletex "G0" & "G2" graphic characters [2]
- All Xerox 860 graphic characters [16]
- All EBCDIC 8-bit graphic characters [4]
- All JIS C 6226 graphic characters
  (including all 6,349 most-frequent Japanese *kanji*) [11]
- All characters required to write the following languages:
  English, Russian, German, French, Spanish, Italian, Portuguese, Dutch, Swedish, Norwegian, Danish, Japanese, Malay/Indonesian, Greek, Ukranian, Polish, and other languages
- All standard office typewriter keyboard characters for the European languages above
- The most commonly used office, technical, and general symbols

A graphic is a symbol produced by a process such as handwriting, drawing, or printing. And, in this standard, a graphic character is defined as a character, other than a control character or rendering character, that is normally represented by a graphic.

In addition to assigning codes to graphic characters, the standard also assigns codes to rendering characters—a character other than a graphic character or control character, which can include any of the following:

- a non-conventional representation of a control code
- a sequence of graphic characters, i.e., ligature or accented character
- a contextually-dependent alternate representation for a graphic character, i.e., initial, medial, or final form for an alphabet such as Arabic
- a "variant" representation for a graphic character, e.g., the rendering character Ā instead of the graphic character A

The need for alternate graphic representations arises from language constraints, aesthetics, typographic preferences of users, etc. Since any alternate form of a graphic character violates the semantic uniqueness requirement for graphic characters codes (see 1.4), rendering character codes are "non-graphic character codes." To distinguish one type of character code from another, i.e., graphic character code from a rendering character code, a separate section of the code space is allocated for each type of character code.

In addition to assigning a section of the code space to sets of graphic characters and sets of rendering characters, this standard also makes provision for control characters, whose occurrence in a particular context initiates, modifies, or stops a control operation. While selected areas of the code space are reserved for control characters, only a single code assignment is made at this time. This single control code assignment, having the identity *Character Select Code*, is pivotal and is given special treatment throughout this standard. Since it is the only control character code assignment at this time and is discussed throughout this document, the control code section (Section 5) is a place holder only. Additions to this section, the assignment of control characters to the reserved control code spaces, will be made in the near future.

Other entities which may be associated with text characters (e.g., logos, signatures, etc.) are not addressed in this document, but a portion of the rendering code space is reserved for private use for this purpose. The intent is to allow for those code assignments that are unique for private use, to enable the incorporation of logos, signatures, and specialized graphics within strings destined for printers. However, entities contained within this specific private use code space are the responsibility of the communicant—this includes administration and utilization of such a code space.

The assigned graphic character codes currently number 7,142 characters and symbols, of which 793 are European letters and symbols, and 6,349 are Japanese *kanji*. Assigned rendering character codes currently number 189 rendering entities, the majority being accented characters for the Latin alphabet. The current list should be sufficient for general use, and is complete in the domains covered.

However, further Xerox characters will certainly be *added* in the future to include new technical domains and the alphabets of more languages (in preparation: Arabic/Farsi, Hebrew, Hindi, Chinese, Korean). Xerox will periodically publish updates to this document to reflect such additions.

1.3 How to use this standard

This standard specifies the character codes to be used for exchange of text information among Xerox System Elements, for such purposes as communications protocols, electronic print files, and document interchange. Since the identified codes are to be used to represent text information that is transmitted between separate Xerox systems, sometimes spanning multilingual boundaries and code conversion interfaces, *a unique, unambiguous, and absolute numerical code is assigned to each semantically different character*. Such an assignment permits the efficient accomplishment of text storage and text-manipulation processes while also ensuring proper interpretation of information.

Sometimes transmitted information may contain commands, names, and other forms of information. If the information is translated to a different external code while actually being transmitted, it must be translated back to these standard numerical codes before being interpreted at the other end, to ensure correct semantic meaning. The codes in this

standard are also used for text and structure information in generalized, editable documents. When content is important, the assigned codes are the Xerox standard graphic character codes.

However, when only shape or form is relevant, such as when printing on an external medium, a rendering character code can be used for entities such as ligatures that the printing (or display) systems will be prepared to interpret. In the rendering of ligatures such as ffi where a single numeric code for ffi violates the singular semantic meaning of a graphic character code, the optimum coded form for printing is a rendering character code. When a rendering character is not used, the correct printing form is established by referencing (or sending along) a complete copy of the appropriate character forms, indexed by the non-standard character codes. Thus there is more flexibility when no semantic meaning is applied to these text strings.

1.4 Design goals

The assignment of Xerox character codes is predicated on specific design goals, as follows:

- To include all commonly used graphic and rendering characters on a fully international basis (see above).
- To assign each distinct character a unique, unambiguous, *absolute* numerical code.
- To allow for high-quality computer typography, including text-rendering devices capable of an indefinite variety of different *appearances* for each distinct character.
- To permit text storage and text-manipulation processes to be as efficient as possible.
- To minimize code translations required to interface with existing information interchange standards, especially ISO, ASCII and Teletex.

1.5 Character codes and text-manipulation processes

Character codes can be viewed as the static representatives of textual content. Operating upon them are many active processes that handle or manipulate text in some way, as illustrated in Figure 1.1.

The assignment of character codes *affects but does not define* the text-manipulation processes.

In a simple regime such as 7-bit ASCII for English text, the relationship between graphic character codes and text-manipulation processes is very direct: there are invariant one-to-one mappings between input keyboard keystrokes, internal character codes, and output printer actions.

In the multilingual, multinational Xerox Systems environment, there are a multiplicity of input keyboards, a variety of output printers and displays, various information interchange encodings, different national collating sequences, and so on. It is impossible, for example, for graphic and rendering character code assignments to correspond optimally to all of these at once. The increased flexibility of a sophisticated information-processing system must come from more powerful text-manipulation processes, not from the assignment of the character codes themselves. In the environment of text-

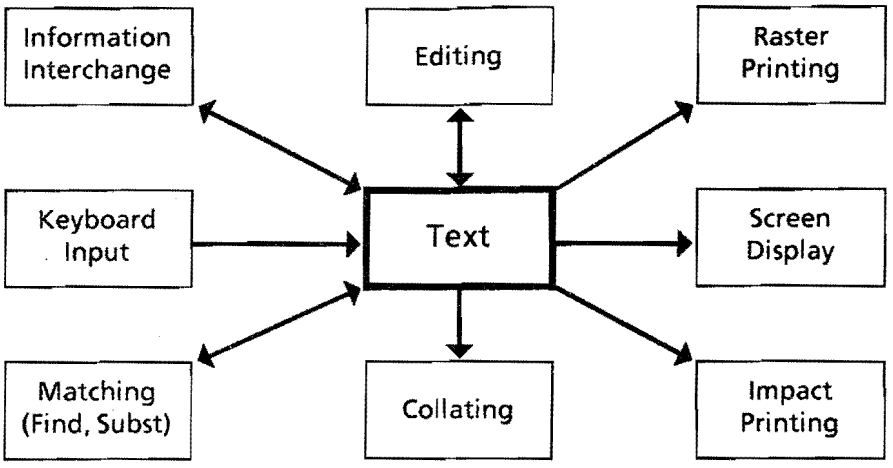


Figure 1.1 Text-manipulation processes

manipulation processes, illustrated in the figure above, the meaning and usage of numerical codes is aimed at the static representation of document text.

The present standard is concerned only with static text content as specified by assigned character codes. It does not define or describe any text-manipulation processes. (It does sometimes allude to their existence, particularly in the discussion of technical details in Appendix C, and in the partitioning of the code space to allocate restricted areas for special purposes.) Figure 1.1 should serve as a reminder that this document tells only part of the story of text processing.

1.6 Character codes and character appearance

A character's numerical code expresses its *identity*, its *content*, its *semantic*, but this does not fully determine a printed or displayed character's visual *appearance*. Separate occurrences of the same graphic character may look quite different. For example, the character code for Latin "C" is always 103_8 in octal notation, but:

This C looks big

This C looks small, perhaps small-caps style.

This C looks superscripted.

This C looks underlined.

This **C** looks bold.

This C looks serif-form.

This C looks sans-serif-form.

This *C* looks italic-style.

The character code models the "C-ness" common to all of the above letters, but there must be *something else* that models their difference in appearance. This appearance-

specification which can be applied to character codes may be called "*looks*" information. Thus we can say:

character appearance = *function of* (character code, character looks)

In other words, textual information (i.e., *content*) is to be stored and transmitted in the form of a sequence of numerical codes, but textual *appearance* is to be stored and transmitted separately in the form of ancillary "*looks*" information.

The notion that character appearance is a function of two variables can be visualized in a 2-dimensional chart. A small fragment of such a chart is shown in Figure 1.2:

Code(s)	*looks* #1	#2	#3	#4	#5	#6	#7
101_8	A	A	A	A	A	A	*A*
102_8	B	B	B	B	B	B	*B*
103_8	C	C	C	C	C	C	*C*

Figure 1.2 Character appearance as a function of character codes and character looks

Each row of the chart represents a character code and each column indicates some particular combination of "*looks*." The column labels ("#1," "#2," etc.) have no meaning except to distinguish one column from another and to indicate the number of character "*looks*" columns. To indicate an entry in the character appearance matrix—Figure 1.2 being only a fragment of such a matrix—we use the following row and column notation: <character code, character "*looks*"> pair.

1.7 Character codes and general "*looks*"

A graphic character appearance matrix may contain a million or more <graphic character code, graphic character "*looks*"> pair entries. Recall from Section 1.2 that graphic character codes currently number 7,142 characters and symbols. In addition to the 7,142 rows, the number of character "*looks*" columns can be as numerous. For example, without standards the variations (the product of the number of sizes, character orientations, character postures, weight, etc.), can exceed a thousand. Other attributes such as setwidth or proportion, strike out, underline, and superscript, add to the number of "*looks*" columns required.

To minimize the number of "*looks*" columns within such a matrix (and the required storage facilities) to several hundred, standards for character "*looks*" have been created. The Xerox General Character Looks Standard [15] specifies the family of standardized character "*looks*." These are called general "*looks*," general "*looks*" being defined as that finite set of "*looks*" upon which agreement has been reached and *which can be broadly applied to all characters in a designated code space.*

While this standard is not concerned with general "*looks*" information as portrayed in Figure 1.2, nor with the text-rendering processes that convert such information into character appearance, it does identify those <rendering character code, rendering character> pairs having non-general "*looks*" within a character appearance matrix. Old Style "*looks*" numerals is an example of non-general "*looks*"; this look fails to meet the

general look test. Not all of the numbers, zero through nine, can have this "named" traditional look.

1.8 Font matrices of <character code, character "*looks*"> pairs

There exists a third dimension to the visualization of character appearance, the 2-dimensional chart portrayed in Figure 1.2. Within this third dimension is the designated typeface which determines the design style of the "pictures" within an instance of a character appearance matrix; a multi-column appearance matrix existing for each named face. A typeface is defined as the features by which you recognize a character's design, hence the word "face."

A partial visualization of this third dimension—a series of small fragments of charts like Figure 1.2, one for each face—illustrates that not all character appearance charts have identical contents. For example, the typeface named Lydian contains one picture of the uppercase character "**A**" with the general look weight attribute value = bold; however, in a non-general "*looks*" column there appears a single, but similar entry—a picture of another uppercase "**A**," but having a different shape. To provide for these "variant" typographic objects which can occur within a typeface design, an identity is required for "variant" forms.

Recall from 1.2 that there are four types of rendering characters, one of which is a "variant" representation for a graphic character. Also recall the example given, the rendering character A instead of A. To provide an identity for the second form of "A," a <rendering character code, "variant" rendering character> pair is required within the Lydian character appearance matrix.

To provide identities for "variant" character forms, a portion of the rendering character code space will be designated for this type of rendering character. However, since this type of rendering entity does not appear in every character appearance matrix, other standards which define the content of a named matrix and a font are also required. The word font or fount is derived from the word "foundry" which is where, originally, type was cast. It has come to mean the vehicle which holds the typeface character collection.

The Xerox Character Grouping Standard [14] provides for several types of character code collections, one very small in size and another of equal or greater size as that of an ISO character collection or a graphic arts font. The code collection similar in size to the ISO concept of character sets is comprised of selected rows from a general "*looks*" character appearance matrix. The result of this selection process is a collection of pairs—<character code, graphic character> pairs.

Another type of Xerox code collection is the small code collection which is used in creation of a character grouping, *a union of several collections having different sizes*—both large (a core collection) and small collections. Small collections having one <character code, character> pair, or several or more <character code, character> pairs, identifies the contents of a character "*looks*" column in a character appearance matrix, one having non-general character "*looks*."

Figure 1.2 serves as a reminder that this document tells only part of the story of text appearance and character design. It also serves as a reminder that additional standards are required, many of which are in preparation.

1.9 Coding of <character code, character> pairs

The question of coding for meaning or shape must be answered in any character code standard. One family of <character code, character> pairs is that comprised solely of <character code, character shape> pairs. An example of such a family is the set of pairs given in the ANSI X3.4-1977 standard [1]. In that standard, character code 47_8 has an identity given as "Apostrophe (Closing Single Quotation Mark; Acute Accent)." For these entities in the ANSI collection (character set), we can say that all of the identities have the same "apostrophe shape." Since all "identities" have a common character code, there exists ambiguity.

Conversely, using a family of <character code, character semantic> pairs and the ISO 6937 standard [10], it is possible to resolve the semantic ambiguity within the ANSI standard. For example, ISO 6937 provides the following <character code, character> pairs: <47_8, Apostrophe>, <271_8, Single quotation mark right>, and <302_8, Acute accent>. The ANSI standard can be derived from the ISO standard by removing (editing) two character codes and using the <47_8, Apostrophe> pair when interpreting octal codes 271_8 and 302_8 at a coding interface.

A primary design objective in the construction of this standard is semantic separation. While the Xerox standard uses national and international standards and maintains compatibility with traditional practices in constructing the standard, the goal is to minimize or eliminate semantic ambiguity (i.e., separation of hyphen from minus in ISO 6937). Additions to this standard, after completion of construction from available standards, will consist primarily of <character code, character semantic> pairs.

While the Xerox standard is concerned with coding for meaning, its composition is such that it does not preclude the creation of a <character code, character> pairs collection based on coding for shape. The advantage in coding for meaning is that an all-graphic character code collection leads to document "text" suitable for subsequent processing or calculation. Such an optimized collection is preferred when it is required to have a processible form of text, a document description used in applications beyond printing. However, in the event that only a final form (not processible) of "text" is required (i.e., for printing originals), editing of a <character code, character semantic> pairs collection provides the necessary <character code, character shape> pairs collection.

1.10 Document organization

Section 2 describes the Xerox 16-bit code space and the Character Set allocations for the different text character code spaces.

Section 3 enumerates the one-to-one Xerox mapping between numerical codes and graphic characters in the graphic character code space.

Section 4 enumerates the one-to-one Xerox mapping between numerical codes and rendering characters in the rendering character code space.

Section 5 enumerates the one-to-one Xerox mapping between numerical codes and control characters in the control character code space.

Section 6 presents the 1-byte (or default) and 2-byte String Encoding.

Appendix A lists references, mainly information interchange standards.

Appendix B presents the same character codes as Sections 3, 4, and 5, in chart form. The user is urged to refer to Sections 3, 4, and 5 to make certain of character semantics.

Appendix C addresses a number of technical questions concerning the details of the Xerox character code assignments.

Appendix D contains differences between the Xerox Character Code Standard and the Japanese Industrial Standard JIS C 6226-1978.

Appendix E is a cross reference of commonly used characters and symbols.

Appendix F is a glossary of common and unique text processing and printing terms.

2

Character code space

This document presumes that textual information is to be stored and transmitted in the form of a sequence of numerical codes called *character codes*. Each character code thus represents a text entity called a *character*. The purpose of this section is to describe the Xerox 16-bit code space and the Character Set allocation for the different types of text characters.

2.1 Background

International Standards define the representation and mapping between numerical codes and text characters:

(1) ISO 646 – 7-bit coded character set for information processing interchange [5].

(2) ISO 2022 – Code extension techniques and use with the ISO 7-bit coded character set [6,7].

The following illustration of the 7-bit code table (Figure 2.1) is used throughout the standards documents of the International Standards Organization (ISO) and various national bodies such as the American National Standards Institute Incorporated (ANSII).

The table includes a set of 32 control characters and a set of 94 graphic characters. Control characters are contained within the first two columns (0 and 1) of the matrix, and the graphic characters fill the remaining six columns (2 through 7). A complete character set in the 7-bit environment is shown in Figure 2.2, and is referred to as the Set G0.

The version illustrated in Figure 2.2 is the International Reference Version (IRV) and is used in international information processing interchange. Control information allocated to the first two columns is shaded to indicate a restricted area which is not a segment of the graphic character code space. The character "space" indicated in the column/row position 2/0, is also shaded and may be regarded as either a control character or a graphic character. The position 7/15 is also excluded from the graphic character set code space and accounts for the set designation as 94 characters (6 columns x 16 rows minus 2/0 and 7/15).

The same character set code space structure is maintained within the Xerox standard to maintain ease of conversion between the Xerox standard and existing international standards. Assignment of graphic characters outside of the outlined structure is avoided to minimize the problems associated with interfacing at the terminal device level. The IRV

					b_7	0	0	0	0	1	1	1	1
					b_6	0	0	1	1	0	0	1	1
					b_5	0	1	0	1	0	1	0	1
b_4	b_3	b_2	b_1			**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	0	0	0	**0**									
0	0	0	1	**1**									
0	0	1	0	**2**									
0	0	1	1	**3**									
0	1	0	0	**4**									
0	1	0	1	**5**									
0	1	1	0	**6**									
0	1	1	1	**7**									
1	0	0	0	**8**									
1	0	0	1	**9**									
1	0	1	0	**10**									
1	0	1	1	**11**									
1	1	0	0	**12**									
1	1	0	1	**13**									
1	1	1	0	**14**									
1	1	1	1	**15**									

Figure 2.1 7-bit code table

set, designated in the absence of a working agreement between a sender and recipient of the data, is also used.

The structure of a family of 8-bit codes remains compatible with the 7-bit structure. Within ISO documents a set is appended to the Set G0 to create columns 8 through 15. Various character sets which may be assigned to the "right" side are referred to as G1, G2, or G3. A member of the right set is obtained by the addition of 1 bit to each of the bit combinations of the 7-bit code. This produces a set of 256 8-bit combinations and results in a graphic character set representative of 188 (94 x 2) characters. These left and right 94 graphic character sets are also referred to as the primary and supplementary sets.

					b_7	0	0	0	0	1	1	1	1
					b_6	0	0	1	1	0	0	1	1
					b_5	0	1	0	1	0	1	0	1
b_4	b_3	b_2	b_1			**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	0	0	0	**0**					0	@	P	`	p
0	0	0	1	**1**				!	1	A	Q	a	q
0	0	1	0	**2**				"	2	B	R	b	r
0	0	1	1	**3**				#	3	C	S	c	s
0	1	0	0	**4**				¤	4	D	T	d	t
0	1	0	1	**5**				%	5	E	U	e	u
0	1	1	0	**6**				&	6	F	V	f	v
0	1	1	1	**7**				'	7	G	W	g	w
1	0	0	0	**8**				(	8	H	X	h	x
1	0	0	1	**9**				)	9	I	Y	i	y
1	0	1	0	**10**				*	:	J	Z	j	z
1	0	1	1	**11**				+	;	K	[	k	{
1	1	0	0	**12**				,	<	L	\	l	|
1	1	0	1	**13**				−	=	M	]	m	}
1	1	1	0	**14**				.	>	N	^	n	~
1	1	1	1	**15**				/	?	O	_	o	

Figure 2.2 Character set G0

Within the Xerox Character Code Standard, both character sets and graphic character codes generally follow the ISO structure, i.e., assignment of graphic characters to 188 of the 256 possible locations. A character set is also said to be composed of two parts, each containing 94 characters. The left side is designated by the letter L following the character set identity and the right side by the letter R.

Specific numeric codes are assigned to every graphic contained within the Xerox Graphic Library [14]. This assignment of numeric codes to graphic characters is patterned after established standards and has a goal of achieving universality of global names, i.e., names for networks, files, or character fonts, for example. This goal is accomplished by assigning

a unique, unambiguous, and absolute numeric code to each graphic. The result is a correct machine interpretation of global names everywhere in the world.

2.2 The code space

2.2.1 Character sets

The total range of numbers available for character codes is taken to be [0 .. 177777_8], i.e., 16 bits, which would permit up to $65{,}536_{10}$ distinct codes. Allowing for reserved control character space, this reduces to 189 x 188 = 35,532.

It is extremely convenient to partition this range into 256 blocks of 256 codes each. Each such block is called a *character set.*

Each 16-bit character code can be viewed as consisting of two 8-bit bytes, where the high-order byte is the character set code and the low-order byte is the character's code *within* the character set (ranging [0 .. 377_8]), as shown below. This point of view is often very useful, especially with regard to string encoding (Section 6).

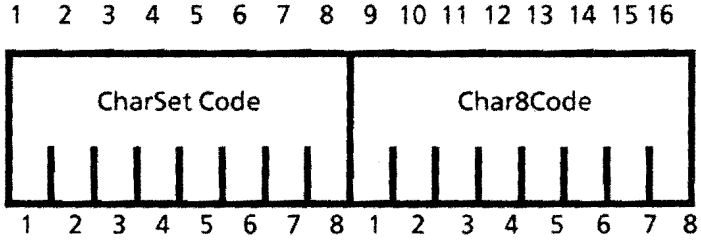


Figure 2.3 16-bit Xerox character code

2.2.2 Character set allocation

The Xerox character codes are assigned so that characters within a single character set tend to be related to each other by traditional usage. The principal goal of this arrangement is to minimize character-set switching in runs of text, but grouping related characters together also tends to clarify the meaning of the character sets themselves. Character sets are also arranged so that successive character sets bear some relationship to each other.

The character sets currently used for Xerox character codes and the standard from which they are derived are listed here for reference; a more detailed explanation of each is given in Sections 3, 4, and 5.

2.2.2.1 Graphic character set allocations

The character sets currently used for graphic character codes and the standard from which they are derived are listed here for reference; a more detailed explanation of each is given in Section 3:

ASCII/ISO/Teletex:

Character Set 0_8 — Latin alphabet and punctuation
Character Set 43_8R — Extended Latin alphabet
Character Set 46_8L — Greek alphabet

JIS:

Character Set 41_8L — JIS Symbol 1–Punctuation and symbols not in Character Set 0
Character Set 42_8L — JIS Symbol 2–Punctuation and symbols not in Character Set 0
Character Set 44_8L — Japanese hiragana syllabary
Character Set 45_8L — Japanese katakana syllabary
Character Set 47_8L — Cyrillic alphabet
Character Sets 60_8L through 117_8L — JIS Level-I Japanese kanji
Character Sets 120_8L through 163_8L — JIS Level-II Japanese kanji

Other:

Character Set 164_8L — Symbol 3 – Miscellaneous Japanese symbols
Character Set 356_8 — General and technical symbols 2
Character Set 357_8 — General and technical symbols 1

2.2.2.2 Rendering character set allocations

The character sets currently used for rendering character codes are listed here for reference; a more detailed explanation of each is given in Section 4:

Character Set 360_8 — Ligatures and format characters
Character Set 361_8 — Accented Latin characters
Character Set 376_8 — Reserved, private use

2.2.2.3 Control character set allocation

Two large blocks of character sets are excluded from availability for use as graphic characters or rendering character codes:

Character Sets 1_8 through 40_8 — Reserved
Character Sets 177_8 through 240_8 — Reserved

These blocks are reserved for control character code assignments.

2.2.3 Character set select code 377_8

The 8-bit byte 377_8 is given unique treatment in the Xerox standard. It is pivotal to the String Encoding (Section 6), where it is called the *character set select code*. Because of this application, the byte 377_8 is not permitted to occur as high- or low-order byte of any Xerox character codes.

2.2.4 Preponderance of kanji characters

It is worth noting that over 90% of the distinct Xerox text characters are kanji ideographs (kanji is a Japanese word meaning *Chinese characters*). Although the Xerox architecture treats kanji the same as non-kanji characters, there are still many cases where it is useful to distinguish the two classes of characters. For example, this document does not enumerate the names of the 6,349 kanji characters that currently have a Xerox character code assigned to them.

2.2.5 Expansion beyond 16 bits

This document specifies a 16-bit character code space. It does not preclude expansion to 24-bit, 32-bit, or greater, character codes. It is anticipated that future versions of this standard will define character code assignments of length greater than 16-bits.

The chart on the following page pictorially summarizes all of the above aspects of the division of the 16-bit code space into character sets and the allocation of the character set sub-spaces.

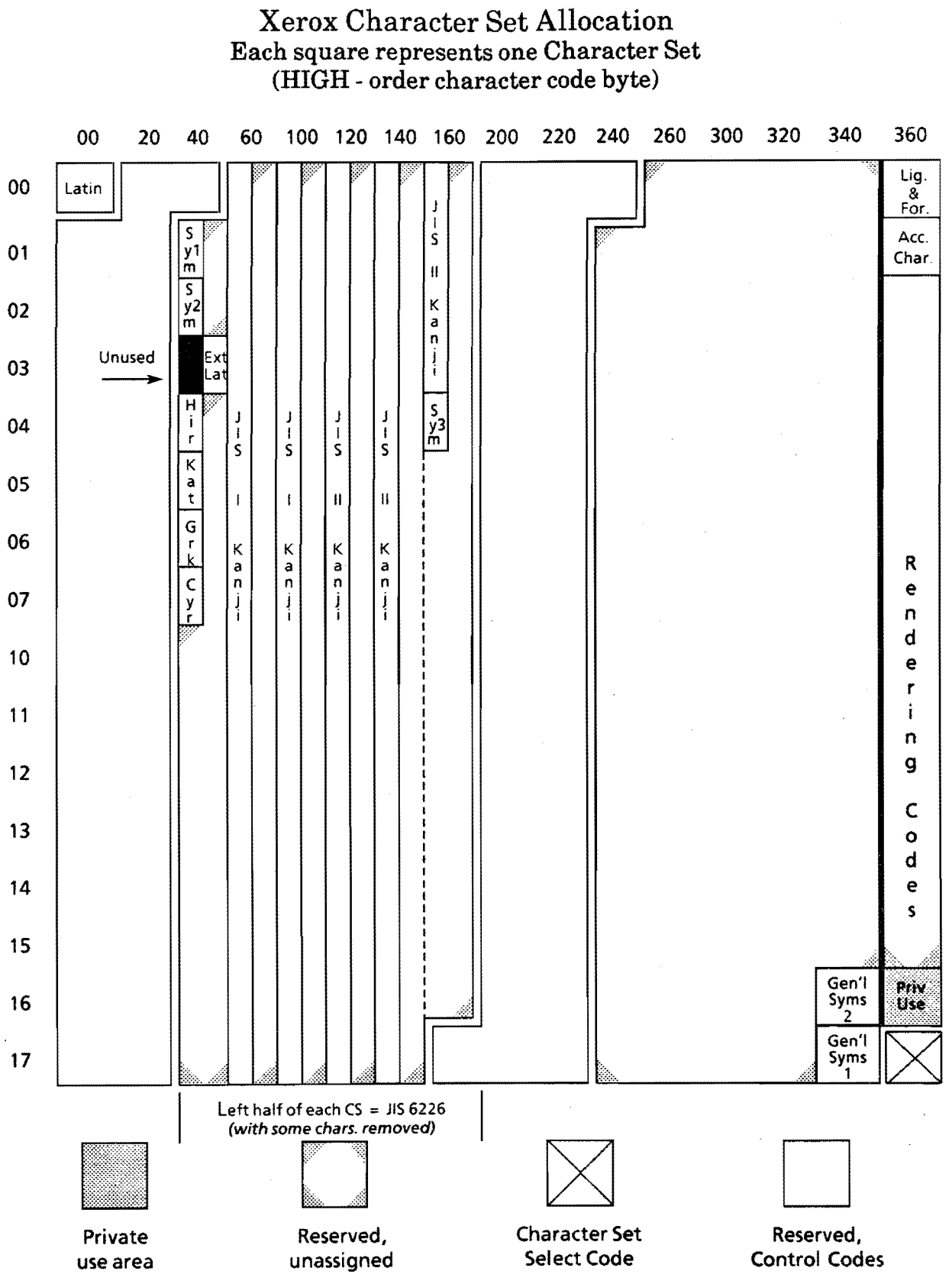


Figure 2.4 Xerox Character Set Allocation

3

Graphic character codes

3.1 Graphic character codes

For us to talk about one character as distinct from another, we can refer to it by its traditional name(s), by its role(s) in a traditional alphabet or symbol system, or by the context(s) in which it is traditionally used. This collection of traditional usage information which identifies the character can be referred to in brief as its *semantic*.

To indicate a graphic character it also helps to show a picture of it, but in some cases this can be misleading since different graphic characters can look very similar or even identical, for example "hyphen" and "minus sign." In the listing which follows, sometimes a picture of a graphic character is shown, but it is the *semantic* information which takes precedence in defining the identity of the character.

In some cases we go so far as to specify what the graphic character is **NOT**, in order to distinguish it from other graphic characters which look similar or identical. In other cases we use the "=" sign to indicate either synonymous names or alternative applications for the same Xerox character. We also use brackets "[ ]" to provide clarifying examples.

Several graphic character shapes traditionally imply the same semantic, i.e., the shapes associated with the semantic "approximately equal." In this case we consider the possibility that the alternate forms are "looks" variants of each other and use a designation such as "approximately equal, type 1" and "approximately equal, type 2" to distinguish one variant from another. While we use this notation to provide information on traditional usage, we recommend using the semantic without the type designation when an "=" sign indicates a synonymous name.

We use a notation such as (41_8 | 124_8) to depict the two-byte structure of a character code. For example, (41_8 | 124_8) refers to the character in Character Set 41_8 which has the 8-bit code 124_8 *within* that character set.

3.2 The graphic character code sets

3.2.1 Character set 0_8: ASCII/ISO/CCITT Latin alphabet and punctuation

Character Set 0_8 serves as Xerox' "default" 8-bit character codespace (in the character sequence body format, see Section 6). This fact makes Character Set 0 the most important Xerox character set, especially with regard to communicating with existing systems.

Therefore Character Set 0 is designed to be fully compatible with existing international standards. It is illustrated in Figure 3.1.

Character Set 0_8L(Left) is identical to the graphics of the ISO International Reference Version (IRV). This version of ISO 646 is used when there is no requirement to use a national or an application-oriented version.

In some cases where ASCII/ISO/CCITT maps more than one distinct semantic into the same character code (e.g., "hyphen" and "minus sign"), the Xerox standard has two distinct characters, one of which is placed outside of Character Set 0. The result is still ASCII/ISO/CCITT compatible, but permits the distinction to be made when necessary.

Character Set 0_8R(Right) is identical to the supplementary graphic set for text communication from ISO 6937. This collection of graphics is also considered to be generic, in that it includes the graphics proposed in the International Text Communication Standard [10]. The composite left and right character sets are shown in the illustration of Character Set 0 in Figure 3.1. The structure follows the pattern established in 2.1, and is presented using octal notation.

The character codes (low-order byte) *within* Character Set 0_8:

40_8	Space (normally nonprinting)
41_8	Exclamation point
42_8	Neutral (vertical) double quote
43_8	Number sign
44_8	General currency symbol
45_8	Percent sign
46_8	Ampersand
47_8	Apostrophe
50_8	Opening parenthesis
51_8	Closing parenthesis
52_8	Asterisk
53_8	Plus sign
54_8	Comma
55_8	Minus sign — **NOT** Hyphen (41_8 | 76_8)
56_8	Period = full stop
57_8	Slant = solidus = virgule = slash
60_8	Digit 0
...	...
71_8	Digit 9
72_8	Colon
73_8	Semicolon
74_8	Less than
75_8	Equals
76_8	Greater than
77_8	Question mark
100_8	Commercial at
101_8	Uppercase Latin (Roman) letter A
...	...
132_8	Uppercase Latin (Roman) letter Z
133_8	Opening bracket
134_8	Reverse slant = backslash = reverse solidus = reverse virgule

Xerox Character Set 0

ASCII/ISO/CCITT Roman Alphabet and Punctuation

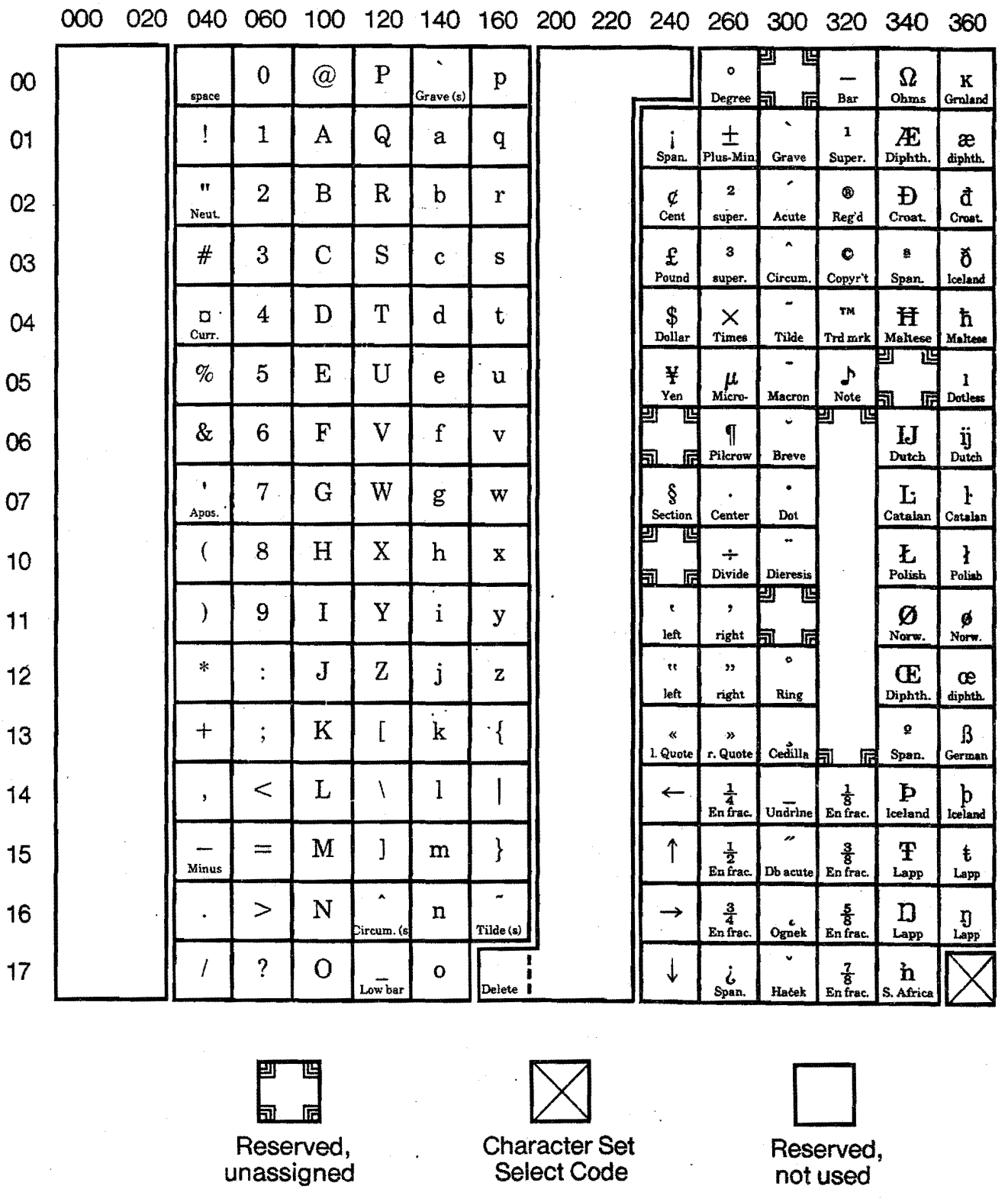


Figure 3.1 Xerox character set 0

135_8	Closing bracket
136_8	Circumflex accent (spacing character)
137_8	Low bar (spacing character)
140_8	Grave accent (spacing character)
141_8	Lowercase Latin (Roman) letter a
...	...
172_8	Lowercase Latin (Roman) letter z
173_8	Opening brace
174_8	Vertical bar
175_8	Closing brace
176_8	Tilde (spacing character)
241_8	Inverted exclamation point (Spanish)
242_8	Cent sign
243_8	Pound-Sterling sign
244_8	Dollar sign
245_8	Yen sign (Japanese)
247_8	Section sign
251_8	Left single Quote = single quote open
252_8	Left double Quote
253_8	Left double guillemet (European quotation mark) — **NOT** much-less-than ($357_8 \mid 102_8$)
254_8	West arrow = leftward arrow
255_8	North arrow = upward arrow
256_8	East arrow = rightward arrow
257_8	South arrow = downward arrow
260_8	Degree sign (spacing character) — **NOT** over-ring accent ($0_8 \mid 312_8$)
261_8	Plus/minus sign
262_8	Superscript 2 as independent character from 2
263_8	Superscript 3 as independent character from 3
264_8	Multiply sign
265_8	Micro sign — **NOT** Greek "mu" ($46_8 \mid 157_8$)
266_8	Paragraph sign = pilcrow
267_8	Centered dot
270_8	Divide sign
271_8	Right single quote = single quote closed
272_8	Right double quote
273_8	Right double guillemet (European quotation mark) — **NOT** much-greater-than ($357_8 \mid 103_8$)
274_8	Fraction one quarter as independent character
275_8	Fraction one half as independent character
276_8	Fraction three quarters as independent character
277_8	Inverted question mark (Spanish)
301_8*	Grave accent
302_8*	Acute accent = phonetic stress mark (non-spacing)
303_8*	Circumflex accent = hat
304_8*	Tilde accent
305_8*	Macron accent = long vowel mark
306_8*	Breve accent
307_8*	Over-dot accent
310_8*	Diaeresis accent = umlaut accent

* These diacritics are non-spacing.

312_8*	Over-ring accent — **NOT** degree sign (0_8 | 260_8)
313_8*	Cedilla undermark
314_8*	Underline (non-spacing undermark)
315_8*	Double acute accent
316_8*	Ogonek undermark = Polish hook
317_8*	Hachek accent = caron
320_8	Horizontal bar
321_8	Superscript 1 as independent character from 1
322_8	Registered sign
323_8	Copyright sign
324_8	Trademark sign (TM)
325_8	Music note
334_8	Fraction one eighth as independent character
335_8	Fraction three eighths as independent character
336_8	Fraction five eighths as independent character
337_8	Fraction seven eighths as independent character
340_8	Ohm sign — **NOT** uppercase Greek "omega" (46_8 | 135_8)
341_8	Uppercase AE digraph
342_8	Uppercase D with stroke (Croatian)
343_8	Feminine Spanish ordinal indicator as independent character from a
344_8	Uppercase H with stroke (Maltese)
346_8	Uppercase IJ digraph (Dutch)
347_8	Uppercase L with middle dot (Catalan)
350_8	Uppercase L with stroke (Polish)
351_8	Uppercase O with slash (Norwegian, Danish)
352_8	Uppercase OE digraph
353_8	Masculine Spanish ordinal indicator as independent character from o
354_8	Uppercase "Thorn" (Icelandic)
355_8	Uppercase T with stroke (Lapp)
356_8	Uppercase "Eng" (Lapp)
357_8	Lowercase n with apostrophe (South African)
360_8	Lowercase k (Greenlandic)
361_8	Lowercase ae digraph
362_8	Lowercase d with stroke (Croatian)
363_8	Lowercase "Eth" (Icelandic)
364_8	Lowercase h with stroke (Maltese) — **NOT** Planck's constant (357_8 | 150_8)
365_8	Lowercase dotless i (Turkish)
366_8	Lowercase ij digraph (Dutch)
367_8	Lowercase l with middle dot (Catalan)
370_8	Lowercase l with stroke (Polish)
371_8	Lowercase o with slash (Norwegian, Danish)
372_8	Lowercase oe digraph
373_8	Double s = Ess-zed = sharp s (German)
374_8	Lowercase "Thorn" (Icelandic)
375_8	Lowercase t with stroke (Lapp)
376_8	Lowercase "Eng" (Lapp)

* These diacritics are non-spacing.

3.2.2 Character Set 41_8: JIS symbols 1 – punctuation and symbols not in CS 0

The character codes (low-order byte) *within* Character Set 41_8 (see reference chart in Appendix B):

42_8	Japanese comma = Chinese comma
43_8	Japanese period = Chinese period
53_8	"daku-on" mark
54_8	"han-daku-on" mark
63_8	Repeat katakana
64_8	Repeat katakana with daku-on
65_8	Repeat hiragana
66_8	Repeat hiragana with daku-on
67_8	"reduplicate"
70_8	"reduplicate above item"
71_8	Repeat kanji
72_8	"shime"
73_8	"Kanji zero"
74_8	Long vowel bar (spacing character)
76_8	Hyphen — **NOT** minus ($0_8 \mid 55_8$)
102_8	Parallel sign, type 1 = double vertical bar
104_8	Three-dot leader
105_8	Two-dot leader
114_8	Left broken bracket
115_8	Right broken bracket
126_8	Left Japanese quote = left Chinese quote
127_8	Right Japanese quote = right Chinese quote
130_8	Left Japanese double quote = left Chinese double quote
131_8	Right Japanese double quote = right Chinese double quote
132_8	Left black lenticular bracket
133_8	Right black lenticular bracket
142_8	Does not equal
145_8	"Less than or equal to"
146_8	"Greater than or equal to"
147_8	Infinity
150_8	Therefore
151_8	male = Mars
152_8	female = Venus
154_8	Minutes sign = feet sign = prime sign = phonetic stress sign (spacing character) — **NOT** acute accent ($0_8 \mid 302_8$)
155_8	Seconds sign = double prime = inches sign (spacing character) — **NOT** double acute accent ($0_8 \mid 315_8$)
156_8	Degrees-Celsius symbol
171_8	White star
172_8	Black star
173_8	White circle
174_8	Black circle
175_8	Two concentric white circles = bull's eye
176_8	White diamond

3.2.3 Character Set 42_8: JIS symbols 2 – punctuation and symbols not in CS 0

The character codes (low-order byte) *within* Character Set 42_8 (see reference chart in Appendix B):

41_8	Black diamond
42_8	Ballot box = wave operator = white square
43_8	Black square
44_8	White point-up triangle — **NOT** uppercase Greek "delta" (46_8 | 105_8)
45_8	Black point-up triangle
46_8	White point-down triangle — **NOT** nabla (357_8 | 271_8)
47_8	Black point-down triangle
50_8	"kome" symbol
51_8	"post office" symbol
56_8	Japanese "geta"

3.2.4 Character Set 43_8: extended Latin alphabet

Character Set 43_8 contains characters defined in ISO 5426, "Extension of the Latin alphabet coded character set for bibliographic information interchange" [8]. They constitute a character set for the interchange of bibliographic citations, including their annotations, in the Latin alphabet. *The characters selected are retained in ISO order and are assigned to code positions on the "right" side of the ISO 8-bit code table.*

The character codes (low-order byte) *within* Character Set 43_8 (see reference chart in Appendix B):

254_8	Musical flat
256_8	Sound recording copyright statement
260_8	Ayn — **NOT** single open quote (0_8 | 251_8)
261_8	Alif/Hamzah — **NOT** single open quote (0_8 | 251_8)
262_8	Lowered left single quote — **NOT** single open quote (0 | 251_8), prints low on line. Used in German, etc.
274_8	Musical sharp
275_8	Mjagkij znak (transliterated Cyrillic alphabet languages)
276_8	Tverdyj znak (transliterated Cyrillic alphabet languages)
300_8*	Low rising tone mark (Vietnamese)
311_8*	Umlaut (German, etc.) — Diaeresis (0_8 | 310_8) is preferred in a single language application and umlaut should only be used where text operations are performed on several languages.
313_8*	High comma off center (Czech, Slovak, etc.)
314_8*	High inverted comma centered (Latvian)
316_8*	Horn (Vietnamese and transliterated Thai)
321_8*	Rude (transliterated Thai)
322_8*	Hook to the left (Latvian, Romanian)
324_8*	Circle below (transliterated Bengali, Hindi, etc.)
325_8*	Half circle below (transliterated Semitic languages, Sanskrit)
326_8*	Dot below (transliterated Bengali, Hindi, etc.)
327_8*	Double dot below (transliterated Urdu)
331_8*	Double underline (transliterated Hindi, Sindhi)

* These diacritics are non-spacing.

332_8*	Vertical bar (African languages)
333_8*	Circumflex undermark (African languages)
335_8*	Left half of ligature sign and of double tilde (Both ligature and double tilde are divided into two parts, the first parts of each being identical. Used in transliterations.)
336_8*	Right half of ligature sign — **see 335_8 above**
337_8*	Right half of double tilde (Tagalog) — **see 335_8 above**

* These diacritics are non-spacing.

3.2.5 Character Set 44_8: Japanese hiragana syllabary

Character Set 44_8 contains hiragana characters used to write about 60% of typical Japanese text.

The character codes (low-order byte) *within* Character Set 44_8 (see reference chart in Appendix B):

41_8	Hiragana small "ah"
...	...
163_8	Hiragana "n"

3.2.6 Character Set 45_8: Japanese katakana syllabary

Character Set 45_8 contains katakana characters used to write about 10% of typical Japanese text.

The character codes (low-order byte) *within* Character Set 45_8 (see reference chart in Appendix B):

41_8	Katakana small "ah"
...	...
163_8	Katakana "n"
164_8	Katakana "vu"
165_8	Katakana small "ka"
166_8	Katakana small "ke"

3.2.7 Character Set 46_8: Greek alphabet

Character Set 46_8 contains characters defined in ISO 5428, "Greek alphabet coded character set for bibliographic information interchange" [9]. *The characters selected are retained in ISO order and in ISO code positions.*

The character codes (low-order byte) *within* Character Set 46_8 (see reference chart in Appendix B):

45_8	Smooth breathing (non-spacing) — Prints over a small letter, before a capital letter.
46_8	Rough breathing (non-spacing) — Prints over a small letter, before a capital letter.
47_8	Iota subscript (non-spacing) — Prints under small letter
64_8	Upper prime — Follow letters used as numbers under 1000

65_8	Lower prime — Follow letters used as numbers from 1000
73_8	Raised full stop (period)
101_8	Capital letter Alpha
102_8	Capital letter Beta
104_8	Capital letter Gamma
105_8	Capital letter Delta
106_8	Capital letter Epsilon
107_8	Capital letter Stigma — Obsolete letter used as 6
110_8	Capital letter Digamma — Obsolete letter used as 6
111_8	Capital letter Zeta
112_8	Capital letter Eta
113_8	Capital letter Theta
114_8	Capital letter Iota
115_8	Capital letter Kappa
116_8	Capital letter Lambda
117_8	Capital letter Mu
120_8	Capital letter Nu
121_8	Capital letter Xi
122_8	Capital letter Omicron
123_8	Capital letter Pi
124_8	Capital letter Koppa — Obsolete letter used as 90 (also written K')
125_8	Capital letter Rho
126_8	Capital letter Sigma
130_8	Capital letter Tau
131_8	Capital letter Upsilon
132_8	Capital letter Phi
133_8	Capital letter Chi
134_8	Capital letter Psi
135_8	Capital letter Omega
136_8	Capital letter Sampi — Obsolete letter used as 900
141_8	Small letter Alpha
142_8	Small letter Beta beginning the word
143_8	Small letter Beta found at middle of the word
144_8	Small letter Gamma
145_8	Small letter Delta
146_8	Small letter Epsilon
147_8	Small letter Stigma — Obsolete letter used as 6
150_8	Small letter Digamma — Obsolete letter used as 6
151_8	Small letter Zeta
152_8	Small letter Eta
153_8	Small letter Theta
154_8	Small letter Iota
155_8	Small letter Kappa
156_8	Small letter Lambda
157_8	Small letter Mu
160_8	Small letter Nu
161_8	Small letter Xi
162_8	Small letter Omicron
163_8	Small letter Pi
164_8	Small letter Koppa — Obsolete letter used as 90
165_8	Small letter Rho
166_8	Small letter Sigma form found at beginning or middle of words
167_8	Small letter Sigma form found at end of words
170_8	Small letter Tau
171_8	Small letter Upsilon
172_8	Small letter Phi

173_8	Small letter Chi
174_8	Small letter Psi
175_8	Small letter Omega
176_8	Small letter Sampi — Obsolete letter used as 900

3.2.8 Character Set 47_8: Cyrillic alphabet

Character Set 47_8 contains the Russian alphabet.

The character codes (low-order byte) *within* Character Set 47_8 (see reference chart in Appendix B):

41_8	Russian capital letter "A"
42_8	Russian capital letter BE
43_8	Russian capital letter VE
44_8	Russian capital letter GHE
45_8	Russian capital letter DE
46_8	Russian capital letter E
47_8	Russian capital letter YO
50_8	Russian capital letter ZHE
51_8	Russian capital letter ZE
52_8	Russian capital letter I
53_8	Russian capital letter SHORT I
54_8	Russian capital letter KA
55_8	Russian capital letter EL
56_8	Russian capital letter EM
57_8	Russian capital letter EN
60_8	Russian capital letter O
61_8	Russian capital letter PE
62_8	Russian capital letter ER
63_8	Russian capital letter ES
64_8	Russian capital letter TE
65_8	Russian capital letter U
66_8	Russian capital letter EF
67_8	Russian capital letter HA
70_8	Russian capital letter TSE
71_8	Russian capital letter CHE
72_8	Russian capital letter SHA
73_8	Russian capital letter SHCHA
74_8	Russian capital letter ER (also hard sign)
75_8	Russian capital letter ERY
76_8	Russian capital letter SOFT SIGN
77_8	Russian capital letter REVERSE E
100_8	Russian capital letter YU
101_8	Russian capital letter YA
121_8	Russian small letter a
...	...
161_8	Russian small letter ya

3.2.9 Character Sets 60_8 through 117_8: JIS LEVEL-I Japanese kanji

Japanese kanji are Chinese-style characters used to write about 35% of average Japanese text. Character Sets 60_8 through 117_8 contain the characters defined as JIS Level-I kanji by the Japanese Industrial Standard. They contain 2,965 Japanese kanji characters. *The characters are retained in JIS order and in JIS code positions.*

3.2.10 Character Sets 120_8 through 163_8: JIS LEVEL-II Japanese kanji

Character Sets 120_8 through 163_8 contain the characters defined as JIS Level-II kanji by the Japanese Industrial Standard. They contain 3,384 Japanese kanji characters. *The characters are retained in JIS order and in JIS code positions.*

3.2.11 Character Set 164_8: symbols 3 – miscellaneous Japanese symbols

Character Set 164_8 contains miscellaneous Japanese symbols.

The character codes (low-order byte) *within* Character Set 164_8 (see reference chart in Appendix B):

41_8	Japanese "kabu"
42_8	Circled katakana "ah"
...	...
57_8	Circled katakana "nu"

3.2.12 Character Set 356_8: general and technical symbols 2

Character Set 356_8 ends a series of consecutive character sets (currently two of them) which contain "symbols" that are not traditionally considered part of linguistic punctuation.

The character codes (low-order byte) *within* Character Set 356_8 (see reference chart in Appendix B):

41_8	Thick space = 3-em space (normally nonprinting)
42_8	4-em space (normally nonprinting)
43_8	Hair space (fixed and normally nonprinting)
44_8	Punctuation space (fixed, device dependent, and normally nonprinting)
56_8	Decimal point = radix point — **NOT** period ($0_8 \mid 56_8$)
174_8	Absolute value = such that, type 2
176_8	Similar to (geometry) = equivalent to

3.2.13 Character Set 357_8: general and technical symbols 1

Character Set 357_8 begins a series of consecutive character sets (currently two of them) which contain "symbols" that are not traditionally considered part of linguistic punctuation.

Some of these symbols, for instance the circle, have a large number of different applications or interpretations (see guides such as *Shepherd's Glossary of Graphic Signs and Symbols* [13]). Only the most common applications are mentioned below.

The character codes (low-order byte) *within* Character Set 357_8 (see reference chart in Appendix B):

41_8	Non-breaking space (normally nonprinting)
42_8	Non-breaking hyphen
43_8	Discretionary hyphen
44_8	En dash
45_8	Em dash
46_8	Figure dash
47_8	Neutral single quote
50_8	Lowered left double quote (European usage)
51_8	German right double quote
52_8	Single "guillemet" left quote — **NOT** "less than" ($0_8 \mid 74_8$)
53_8	Single "guillemet" right quote — **NOT** "greater than" ($0_8 \mid 76_8$)
54_8	En quad
55_8	Em quad
56_8	Figure space = numeric space (normally nonprinting)
57_8	Thin space = 5-em space (normally nonprinting)
60_8	Dagger
61_8	Double dagger
62_8	Bra
63_8	Ket
64_8	Right-pointing index
65_8	Left-pointing index
66_8	Left perpendicular (perp)
67_8	Right perpendicular (perp)
70_8	Left 2 perpendicular (perp)
71_8	Right 2 perpendicular (perp)
72_8	Left white lenticular bracket (Chinese)
73_8	Right white lenticular bracket (Chinese)
74_8	NorthWest arrow
75_8	SouthEast arrow
76_8	NorthEast arrow
77_8	SouthWest arrow
100_8	Care of
101_8	Per thousand = per mil
102_8	Much less than — **NOT** left guillemet ($0_8 \mid 253_8$)
103_8	Much greater than — **NOT** right guillemet ($0_8 \mid 273_8$)
104_8	Not less than
105_8	Not greater than
106_8	Divides — **NOT** vertical bar ($0_8 \mid 174_8$)
107_8	Does not divide
110_8	Double slash = parallel sign, type 2
111_8	Not parallel
112_8	Is a member of — **NOT** Greek "epsilon" ($46_8 \mid 146_8$)
113_8	Is not a member of
114_8	Such that, type 1 = contains as a member
115_8	Double back arrow = is implied by

116_8 Double double arrow = iff
117_8 Double right arrow = implies
120_8 Reversible reaction, type 2
121_8 Reversible reaction, type 1 = electric current
122_8 Double arrow
123_8 Curly arrow
124_8 Contains, type 1 [A contains every element of B, but A does not equal B]
125_8 Contained in, type 1 [Every element of B belongs to A, but B is not equal to A]
126_8 Intersection [The set of all elements which belong to both A and B] Also: Product of intersection of classes (math logic) or sets (algebra)
127_8 Union [The set of all elements which belong to A or to B or to A and B] = sum or union of classes (math logic) or sets (algebra)
130_8 "Contains or equals" [A contains every element of B]
131_8 "Contained in or equals" [Every element of B belongs to A] Also: Identity or inclusion in the set
132_8 Properly includes in set [A contains every element of B, but A does not equal B] = contains, type 2 (This meaning is $(357_8 \mid 130_8)$ and requires use of $(357_8 \mid 124_8)$ to indicate A does not equal B, per ISO 31/X1-1978.)
133_8 Proper inclusion in set [Every element of B belongs to A, but B is not equal to A] = Contained in, type 2 (This meaning is $(357_8 \mid 131_8)$ and requires use of $(357_8 \mid 125_8)$ to indicate B does not equal A, per ISO 31/X1-1978.)
134_8 Neither contains nor is equal to = does not contain as a subset
135_8 Neither contained in nor is equal to
136_8 Does not contain = does not properly include in set
137_8 Is not contained in = non-proper inclusion in set
140_8 Checked ballot box
141_8 Null set — **NOT** Norwegian/Danish o with slash $(0_8 \mid 371_8)$
142_8 Abstract + = Earth = Sign of composition
143_8 Abstract -
144_8 Abstract x
145_8 Abstract /
146_8 Centered bullet — **NOT** black circle $(41_8 \mid 174_8)$
147_8 Centered ring — **NOT** white circle $(41_8 \mid 173_8)$
150_8 Planck's constant
151_8 Liter
152_8 Not
153_8 Broken vertical bar
154_8 Angle — **NOT** "less than" $(0_8 \mid 74_8)$
155_8 Spherical angle
156_8 "Identifier"
157_8 Because
160_8 Perpendicular
161_8 Is proportional to
162_8 Identically equal = equivalent
163_8 Equal by definition
164_8 Questioned equality
165_8 Integral
166_8 Contour integral
167_8 Asymptotically equal to = Approximately equal, type 1
170_8 Isomorphic = congruent

171_8	Asymptotic to = approximately equal, type 2
172_8	Summation — **NOT** uppercase Greek "sigma" (46_8 | 126_8)
173_8	Product — **NOT** uppercase Greek "pi" (46_8 | 123_8)
174_8	Radical = root
175_8	"Minus or plus"
176_8	Shade
241_8	Cruzeiro (Brazilian)
242_8	Florin = Guilder (Dutch)
243_8	Francs
244_8	Pesetas (Spanish)
245_8	European currency symbol
246_8	Milreis = Escudo (Portuguese)
247_8	Generic Infinity Sign — **NOT** Hebrew aleph
250_8	Number
251_8	Take
252_8	TEL (telephone)
253_8	Yogh (Old English) = dram = IPA "zh" sound
254_8	Complex number
255_8	Natural number
256_8	Real number = reluctance
257_8	Integer
260_8	Left ceiling
261_8	Right ceiling
262_8	Left floor
263_8	Right floor
264_8	There exists
265_8	For all
266_8	And
267_8	Or
270_8	QED
271_8	Nabla = del = differential operator
272_8	Partial derivative
273_8	OCR hook
274_8	OCR fork
275_8	OCR chair
276_8	Alternating current
277_8	Double low bar (spacing character)
300_8	Arc
301_8	Fixed-pitch Roman numeral I as independent from letter I
...	...
312_8	Fixed-pitch Roman numeral X as independent from letter X
313_8	Spades
314_8	Hearts
315_8	Diamonds
316_8	Clubs
317_8	Check mark = yes
320_8	X mark = no
321_8	Circled number 1
...	...
332_8	Circled number 10
333_8	Circled East (right) arrow
334_8	Circled East-then-South (right-then-down) arrow

335_8	Circled South-then-West (down-then-left) arrow
336_8	Peace symbol
337_8	Smile face = "have a nice day!"
340_8	Skull & crossbones
341_8	Thick vertical line
342_8	Thick horizontal line
343_8	Thick intersecting lines
344_8	Thin vertical line
345_8	Thin horizontal line
346_8	Thin intersecting lines
347_8	Sun = abstract multiplication
350_8	First quarter moon
351_8	Third quarter moon
352_8	Mercury
353_8	Jupiter
354_8	Saturn
355_8	Uranus
356_8	Neptune
357_8	Pluto
360_8	Aquarius
361_8	Pisces
362_8	Aries
363_8	Taurus
364_8	Gemini
365_8	Cancer
366_8	Leo
367_8	Virgo
370_8	Libra
371_8	Scorpius = minim
372_8	Sagittarius
373_8	Capricorn
374_8	Telephone symbol
375_8	Fraction one third as independent character
376_8	Fraction two thirds as independent character

4

Rendering character codes

4.1 Rendering entity definition

Recall from 1.2 that the standard assigns codes to rendering characters, a character other than a graphic character or control character, which can include any of the following:

- a non-conventional representation of a control code
- a sequence of graphic characters, i.e., ligature or accented character
- a contextually-dependent alternate representation for a graphic character, i.e., initial, medial, or final form for an alphabet such as Arabic
- a "variant" representation for a graphic character, e.g., the rendering character Ā instead of the graphic character A

Recall further that rendering entities which may be associated with text characters (e.g., logos, signatures, etc.) are not addressed in this document, but that a portion of the rendering code space is reserved for private use for this purpose. The intent is to enable the incorporation of logos, signatures, and specialized graphics within strings destined for printers.

Given that there are several classes of rendering characters, it is important that rendering entities be defined as precisely as possible. Most important are those rendering character codes which can be algorithmically substituted for graphic character codes. In this standard, rendering characters and their numerical codes are defined to be *A single character or collection of characters which can be algorithmically substituted for a graphic character, or characters, to alter the displayed or printed appearance of the normal graphic character, or graphic characters, in conformance with predetermined typographic, linguistic, or formatting rules.*

The definition of rendering character includes "variant" representations for a graphic character; however, some "variant" representations for a graphic character are excluded by the formal definition of a rendering character. An example of a "variant" that fails to qualify as a rendering character is the *non-general "look"* known as Swash, there is no consistency among typeface suppliers as to what graphic characters may have the Swash

appearance. As a consequence, automatic substitution cannot be successfully accomplished.

In the case of the non-general "look" known as Old Style, selected entities qualify as rendering character candidates. Old Style includes certain numerals which when used within text improves readability; however, they are not recommended when figures appear in tables. Since there is universal agreement on which numerals may have the Old Style "look," these numerals qualify as rendering characters—the key phrase in the definition of a rendering character is *algorithmically substituted.*

The goal of formal definition is to allow for automatic substitution where necessary and specification by formatters when desired.

4.2 Rendering entity code space

4.2.1 Rendering characters

Rendering character codes can always be distinguished from graphic character codes, as the high order byte is always larger than the higher order byte for graphic character codes. Rendering Code Sets 360_8 through 375_8 are reserved for rendering characters as defined in 4.1. Using this definition the codespace allocation is primarily for ligatures and accented characters. A reasonable estimate of future assignments is as follows: Arabic - 450, Devanagari - 300, Korean - 400, Roman - 350, and other languages - approximately 500. For other classes of renderings, requirements are less than 200 for each category.

4.2.2 Private use rendering entities

The Rendering Code Set 376_8 is unique in that entities assigned to this set do not satisfy the definition of a rendering code. The assignment within this character set of logos, signatures, and certain "variant" representations, those having non-general "looks," is for local use. As such, the communicant is responsible for administration and utilization of this code space.

There is no reason to assume that entities in this character set on a given printer or display will match, in any way, with entities on another printer or display for identical codes. The intent is to allow for code assignments that are unique to a device. This permits clients to conveniently incorporate specialized or non-standard entities within their systems or printers.

Entities assigned within the private use code space cannot be used in some forms of communication interchange. See 1.3, Utilization, for communication of non-rendering character codes.

4.3 The rendering character code sets

For us to talk about one rendering character as distinct from another, we can refer to it by its traditional name(s), by its role(s) in a traditional alphabet or symbol system, or by the context(s) in which it is traditionally used. This collection of traditional usage information which identifies the rendering character can be referred to in brief as its *semantic.*

To indicate a rendering it also helps to show a picture of it, but in some cases this can be misleading since different renderings can look very similar or even identical. In the listing which follows, sometimes a picture is shown, but it is the *semantic* information which takes precedence in defining the identity of the rendering character.

In some cases we go so far as to specify what the rendering character is **NOT**, in order to distinguish it from other rendering characters which look similar or identical. In other cases we use an "=" sign to indicate either synonymous names or alternative applications for the same Xerox character.

We use a notation such as (360_8 | 45_8) to depict the two-byte structure of a rendering character code. For example, (360_8 | 45_8) refers to the rendering character in Character Set 360_8, which has the 8-bit code 45_8 within that character set.

4.3.1 Character Set 360_8: ligatures and field format symbols

The rendering codes (low-order byte) *within* Character Set 360_8 (see reference chart in Appendix B):

41_8	Ligature ff
42_8	Ligature ffi
43_8	Ligature ffl
44_8	Ligature fi
45_8	Ligature fl
46_8	Ligature st (Quaint character)
271_8	Graphic entity representative of HORIZONTAL TABULATION
272_8	Graphic entity representative of LINE FEED (LF)
275_8	Graphic entity representative of NEW LINE (NL)
302_8	Graphic entity representative of PAGE FORMAT CHARACTER (PFC)
303_8	Graphic entity representative of START OF DOCUMENT (SOD)
304_8	Graphic entity representative of STOP CHARACTER
312_8	Graphic entity representative of "SUBSTITUTE" character
331_8	Graphic entity representative of PARAGRAPH-TAB
335_8	Graphic entity representative of NEW PARAGRAPH (NP)
341_8	Graphic for Field Format: Match special character
...	...
357_8	Graphic for Field Format: Match special character
366_8	Graphic for Field Format: Any Greek Letter
367_8	Graphic for Field Format: Any Russian Letter
370_8	Graphic for Field Format: Any Hiragana character
371_8	Graphic for Field Format: Any Katakana character
372_8	Graphic for Field Format: Any Kanji character
373_8	Graphic for Field Format: Any Japanese character
374_8	Graphic entity representative of Space, Type 1
375_8	Graphic entity representative of Space, Type 2

4.3.2 Character Set 361_8: accented Latin characters

The character codes (low-order byte) *within* Character Set 361_8 (see reference chart in Appendix B):

41_8	Grave A
42_8	Acute A
43_8	Circumflex A
44_8	Tilde A
45_8	Macron A
46_8	Breve A
47_8	Diaeresis A = umlaut A
50_8	Ring A = angstrom A
51_8	Ogonek A
52_8	Acute C
53_8	Circumflex C
54_8	High dot C
55_8	Cedilla C
56_8	Hachek C = caron C
57_8	Hachek D = caron D
60_8	Grave E
61_8	Acute E
62_8	Circumflex E
63_8	Macron E
64_8	High dot E
65_8	Diaeresis E = umlaut E
66_8	Ogonek E
67_8	Hachek E = caron E
71_8	Circumflex G
72_8	Breve G
73_8	High dot G
74_8	Cedilla G
75_8	Circumflex H
76_8	Grave I
77_8	Acute I
100_8	Circumflex I
101_8	Tilde I
102_8	Macron I
103_8	High dot I
104_8	Diaeresis I = umlaut I
105_8	Ogonek I
106_8	Circumflex J
107_8	Cedilla K
110_8	Acute L
111_8	Cedilla L
112_8	Hachek L = caron L
113_8	Acute N
114_8	Tilde N
115_8	Cedilla N
116_8	Hachek N = caron N
117_8	Grave O
120_8	Acute O
121_8	Circumflex O
122_8	Tilde O
123_8	Macron O

124_8	Diaeresis O = umlaut O
125_8	Double acute O
126_8	Acute R
127_8	Cedilla R
130_8	Hachek R = caron R
131_8	Acute S
132_8	Circumflex S
133_8	Cedilla S
134_8	Hachek S = caron S
135_8	Cedilla T
136_8	Hachek T = caron T
137_8	Grave U
140_8	Acute U
141_8	Circumflex U
142_8	Tilde U
143_8	Macron U
144_8	Breve U
145_8	Diaeresis U = umlaut U
146_8	Ring U
147_8	Double acute U
150_8	Ogonek U
151_8	Circumflex W
152_8	Grave Y
153_8	Acute Y
154_8	Circumflex Y
155_8	Diaeresis Y = umlaut Y
156_8	Acute Z
157_8	High dot Z
160_8	Hachek Z = caron Z

241_8	Grave a
242_8	Acute a
243_8	Circumflex a
244_8	Tilde a
245_8	Macron a
246_8	Breve a
247_8	Diaeresis a = umlaut a
250_8	Ring a
251_8	Ogonek a
252_8	Acute c
253_8	Circumflex c
254_8	High dot c
255_8	Cedilla c
256_8	Hachek c = caron c
257_8	Hachek d = caron d
260_8	Grave e
261_8	Acute e
262_8	Circumflex e
263_8	Macron e
264_8	High dot e
265_8	Diaeresis e = umlaut e
266_8	Ogonek e
267_8	Hachek e = caron e
270_8	Acute g
271_8	Circumflex g

272_8	Breve g
273_8	High dot g
275_8	Circumflex h
276_8	Grave i
277_8	Acute i
300_8	Circumflex i
301_8	Tilde i
302_8	Macron i
304_8	Diaeresis i = umlaut i
305_8	Ogonek i
306_8	Circumflex j
307_8	Cedilla k
310_8	Acute l
311_8	Cedilla l
312_8	Hachek l = caron l
313_8	Acute n
314_8	Tilde n
315_8	Cedilla n
316_8	Hachek n = caron n
317_8	Grave o
320_8	Acute o
321_8	Circumflex o
322_8	Tilde o
323_8	Macron o
324_8	Diaeresis o = umlaut o
325_8	Double acute o
326_8	Acute r
327_8	Cedilla r
330_8	Hachek r = caron r
331_8	Acute s
332_8	Circumflex s
333_8	Cedilla s
334_8	Hachek s = caron s
335_8	Cedilla t
336_8	Hachek t = caron t
337_8	Grave u
340_8	Acute u
341_8	Circumflex u
342_8	Tilde u
343_8	Macron u
344_8	Breve u
345_8	Diaeresis u = umlaut u
346_8	Ring u
347_8	Double acute u
350_8	Ogonek u
351_8	Circumflex w
352_8	Grave y
353_8	Acute y
354_8	Circumflex y
355_8	Diaeresis y = umlaut y
356_8	Acute z
357_8	High dot z
360_8	Hachek z = caron z

5

Control character codes

At present there are no control character code assignments. Additions to this section, the assignment of control characters to the reserved control code spaces, will be made in the near future.

6

String encoding

6.1 Scope

This document presumes that textual information is to be stored and transmitted in the form of a sequence of numerical codes. This section specifies the forms in which these sequences of numerical codes can be represented and how they are typically used to transmit fragments of text such as names and messages. These fragments are referred to as *strings*, and a *Xerox string* means a string consisting of character codes defined by this standard.

There are many situations in which more complex and sophisticated models of text are required which might serve functions greater than the simple transmission of a sequence of characters. To distinguish between these more complex situations and the simple view of text here, we refer to the format given here as plain text. This standard makes no attempt to define formats for any other types of text.

Character codes are normally numerical codes in the range 0 to 65,535. Not all values in this range are valid, and the use of higher numbers is not precluded from later versions of this standard. It is useful to regard these numbers as 16-bit entities. The Character Sequence Encoding defines a string as a sequence of *stringlets*, each stringlet consisting of a declaration followed by a body.

There are two types of stringlets: 1-byte, which uses one byte per character, and 2-byte, which uses two bytes per character. The two types of stringlet can be mixed in any combination except that there is an implicit 1-byte stringlet declaration at the beginning of every string. This means that every string *starts* with a 1-byte stringlet body, which may be null. In order to conform to this standard, a device must support both types of stringlet.

The 1-byte stringlet body is defined in such a manner that:

1) a sequence of characters is compressed into strings of 8-bit bytes on a one-for-one basis, i.e., one 8-bit byte for each 16-bit entity.

2) all sequences of 8-bit ISO 646 characters of high frequency usage constitute valid Xerox strings.

The 2-byte stringlet body consists of a sequence of 16-bit codes.

6.2 Review of definitions

Recall from 2.2 that the total range of numbers available for character codes, namely [0 .. 177777_8], is partitioned into 256 blocks of 256 codes each, called *character sets*. Each 16-bit character code can be viewed as consisting of two 8-bit bytes, where the high-order byte is the character set code and the low-order byte is the character's code *within* the character set, as shown in Figure 6.1. We shall designate these bytes by the terminology **CharSet** and **Char8Code**, respectively.

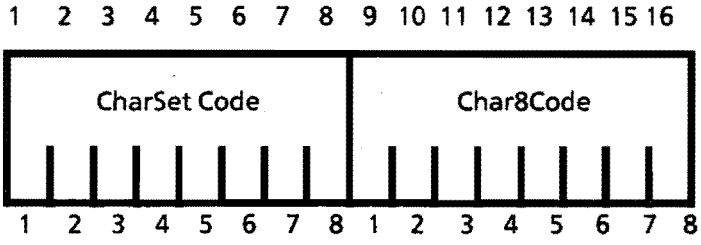


Figure 6.1 16-bit Xerox character code

Recall further that the character codes are assigned so that characters within a single character set tend to be related to each other by traditional usage. In normal text, successive characters tend to come from the same character set, usually letters of the same alphabet.

Recall finally that the 8-bit byte 377_8, called the *character set select code*, is forbidden to occur as the high- or low-order byte of any character codes.

Given this structure on the character codes, the character sequence coding is a simple run-encoding of the **CharSet** bytes.

The same character set code space structure is maintained within the Xerox standard to maintain ease of conversion between the Xerox standard and existing international standards. Assignment of graphic characters outside of the outlined structure is avoided to minimize the problems associated with interfacing at the terminal device level. The IRV set, designated in the absence of a working agreement between a sender and recipient of the data, is also used.

6.3 Simple encoding examples

Below are a few simple examples of sequences of characters that have been encoded using character sequence encoding. These should be helpful in understanding the formal definitions that follow. The numbers are the successive bytes of the encoded sequence given in octal. The "**377**" is the character set select, with the character set number following it in italics.

Examples:

ASCII based

A	S	C	I	I	<sp>	b	a	s	e	d
101	123	103	111	111	040	142	141	163	145	144

†footnote

	†			f	o	o	t	n	o	t	e
377	*357*	060	**377**	*000*	146	157	157	164	156	157	164	145

a ≠ α — as a 2-byte stringlet

			a	<sp>	≠	<sp>	α
377	***377***	*000*	000 141	000 040	041 142	000 040	046 141

6.4 Syntax

The Xerox String Encoding provides a method for encoding any 16-bit (two-byte) number that does not contain the value 377_8 (decimal value 255) in either its left-hand or right-hand bytes. Thus, with this encoding it is possible to encode any number in the range [0...65278] that does not leave a remainder of 255 when divided by 256. All legitimate Xerox Character Codes lie within this range. This encoding provides the ability to encode many numbers that cannot be legitimate Xerox Character Codes, hence this encoding system encompasses a broader class of 16-bit codes than is required for the encoding of Xerox Character Codes.

The following provides a definition of the encoding expressed in a Backus-Naur format.

CSselect:: = 377_8
As above.

CharSet8 :: = [0...376_8]
Any 8-bit number other than 377_8 (CSselect). **CharSet8** designates a legal character set (as listed in 2.2). Note that any legal character set designator must be in this range, but that not all numbers in this range may be legitimate values for **CharSet8** in the Xerox Character Code Standard. Legal values for **CharSet8** in the Xerox Character Code Standard are in the set **[0, 41_8 .. 176_8, 241_8 .. 376_8]**. Note also that not all legitimate **CharSet8** values in the Xerox Character Code Standard in this range have yet been designated as legal values for **CharSet8**. Note also that this range includes the value 0.

CharSet16 :: = [0...376_8]
An 8-bit number designating a legal 16-bit character set. In this version of the standard only 0 is valid. Allowing non-zero values of **CharSet16** would allow addressing of 24-bit characters—in this version only 0 is legal.

CS8Declaration :: = CSselect CharSet8
A 2-byte sequence to declare a new 188 character code set. If this occurs within a 2-byte encoded sequence, then the encoding is changed to 1-byte. If this occurs within a 1-byte encoded sequence, then it causes the character set to be changed as defined in 6.4. Note

that there are 189 possible character sets, of 188 characters each, designated by the numbers in the set [0, 41_8 .. 176_8, 241_8 .. 376_8].

CS16Declaration ::= CSselect CSselect CharSet16
A 3-byte sequence preceding a sequence of 2-byte encoded characters. In this version of the standard the only legal value of **CharSet16** is 0. Therefore **CS16Declaration** is always (in octal) "377 377 000."
[Allowing non-zero values of **CharSet16** would allow addressing of 24-bit characters—in this version only 0 is legal]

Char8Code::= [0...376_8]
Any 8-bit number other than 377_8 (CSselect). **Char8Code** is a character's code within its character set (as listed in 2.2). Note that any legal character code within its character set must be in this range, but that not all numbers in this range are necessarily legal values for **Char8Code**. Legal values for **Char8Code** are in the set [40_8 .. 176_8, 241_8 .. 376_8]. Further, legal values for **Char8Code** are a function of the value for the associated **CharSet** byte. Note that the above range does not include the value 0, but does include 40_8.

Char16Code::= CharSet8 Char8Code
A two-byte 16-bit character code. Note that not all values of **Char8Code** are necessarily legal. Legal values for **Char8Code** are a function of the value for the associated **CharSet8** byte.

Stringlet8 ::= ∅ | Stringlet8 Char8Code
By default, a **Stringlet8** is interpreted as character codes lying within Character Set 0. This is the 1-byte stringlet body which occurs at the beginning of a **String**.

DeclaredStringlet8 ::= CS8declaration Char8Code |
DeclaredStringlet8 Char8Code
A 1-byte stringlet complete with a specific code space declaration at its head and a body consisting of a sequence of **Char8Codes**.

DeclaredStringlet16 ::= CS16declaration Char16Code |
DeclaredStringlet16 Char16Code
A 2-byte stringlet with an explicit **CS16declaration** at its head and a body consisting of a sequence of **Char16Codes**.

String ::= Stringlet8 |
DeclaredStringlet8 |
DeclaredStringlet16 |
String DeclaredStringlet8|
String DeclaredStringlet16
A sequence of DeclaredStringlets, except that the first one may be undeclared (but this may be null; if it is not null then it must be a **Stringlet8**). DeclaredStringlets may be mixed in any combination (apart from the first **Stringlet8**).

6.5 Interpretation of the encoding

The syntax above defines both the 1-byte and 2-byte stringlets, but some comments on the process of interpreting the 1-byte stringlet body to get **Char8Code** bytes back into logical 16-bit character codes are in order.

- In interpreting a 1-byte stringlet body, the high-order byte (**CharSet**) for each character code can be considered to come from an 8-bit finite-state machine which tracks the sequence byte-by-byte. The **CharSet** state is initialized to 0 at the beginning of the interpretation. During interpretation the finite-state machine is alerted by a **CS8Declaration** to switch state to the indicated CharSet.

- The 1-byte stringlet implies that the number of logical characters in a sequence cannot be assumed to be equal to the number of bytes. Likewise, any process which wants to step through a sequence of characters encoded by means of the 1-byte stringlet in a-character-at-a-time fashion cannot assume that each byte corresponds to a single character.

As stated before, the default state is initialized to 1-byte stringlet body within character set 0 Character Code Standard at the beginning of the interpretation process.

This definition has impact on any processes which read a sequence of characters "backwards," from tail-to-head. Processes which read a sequence of characters encoded by means of the character sequence encoding "backwards," from tail-to-head, must locate the beginnings of **DeclaredStringlet8** or **DeclaredStringlet16** segments in order to know how to interpret each character. This is facilitated by the fact that the **CSselect** byte 377_8 is forbidden as the high- or low-order byte of character codes, so the beginnings of **DeclaredStringlet8** or **DeclaredStringlet16** segments may be found simply by scanning for 377_8 bytes–i.e., an isolated sequence of the form 377_8 or 377_8 377_8. Since Character Set 377_8 is forbidden, a sequence of more than two 377_8 bytes in succession within a sequence of characters is impossible.

Since a **CS8Declaration** requires 2 bytes, a 1-byte stringlet does not result in a compression of a string if the average run-length (the number of characters between **CharSet** changes) is less than 2.0 characters. For ordinary European-language text this not the case and 1-byte stringlet is an economic representation, but for the Japanese and Chinese languages, with their thousands of kanji characters, the situation is different. Here the use of a 2-byte stringlet is more economical. This is the main rationale for allowing 2-byte stringlets: so that Kanji-based text can be represented and stored with reasonable efficiency.

In order to use a 2-byte stringlet, because the default is a 1-byte stringlet body, it must be prefixed by a **CS16Declaration**, which functions as an annunciator for the 2-byte stringlet body.

Return from a 2-byte stringlet body is effected by a single "377_8" character followed by a single byte that designates the high-order byte (**CharSet**) of the subsequent characters. Therefore no 16-bit 2-byte encoded character code can contain the value 377_8 in any of its byte positions.

6.6 Relationship to existing 8-bit encodings

The character sequence encoding specifies the default convention that the bytes are interpreted as **Char8Codes,** i.e., a 1-byte stringlet body, within Character Set 0 until a **CS8Declaration** or **CS16Declaration** is encountered. This means that if the entire Xerox string content consists solely of characters from Character Set 0 as a 1-byte stringlet body, then the Xerox string is nothing more than a sequence of 8-bit **Char8Code** bytes taken from Character Set 0. (See the first example in 6.3).

Since Character Set 0 is directly taken from the ASCII/Teletex ISO-based standards, it follows that *the overwhelming majority of ASCII, Teletex, and ISO strings are bit-for-bit identical to the Xerox Character Sequence Encoding with 1-byte stringlet body.* (The only qualifications relate to "$" in ASCII and "#" and "general currency sign" in CCITT recommendations S.61 and S.100 when compared to ISO 6937.)

6.7 European diacritical marks in strings

A European diacritical mark is a small mark which is traditionally thought of as being applied to some other character; the mark never appears alone by itself in ordinary text. However, such marks are assigned their own separate character codes. Examples include the "accent" and "undermark" characters 301_8 through 317_8 of Character Set 0 and characters 300_8 through 337_8 of Character Set 43. Excluded from this discussion are marks in non-European languages such as Arabic, Hebrew, Hindi, Thai, etc.

ISO specifies that in a string body European diacritics always precede the character which they modify. Multiple diacritics are to be entered in the order in which they appear reading left to right or top to bottom. The BACKSPACE character (0/8 of ISO 646, a control character in that standard) is not to be used to identify diacritics.

Characters composed of two or more components, of which one is a qualifying mark, are encoded as follows:

(1) Characters in which a qualifying mark is superimposed upon the basic character are encoded as a single character (for example, uppercase D with stroke, 342_8 in Character Set 0).

(2) Characters in which the qualifying mark is separated from, or only touches, the basic character are encoded as two characters (for example, "ç" = "cedilla" and c or "ç" = $313_8 + 143_8$ of Character Set 0).

6.8 General encoding example

Below are a few examples of sequences that have been encoded using the full capability of the Xerox Character Encoding. The declarations are shown in italics. For ease of reference, the examples of 6.3 are repeated here.

Examples:

(1-byte stringlet body only)
ASCII based

A	S	C	I	I	<sp>	b	a	s	e	d
101	123	103	111	111	040	142	141	163	145	144

(1-byte stringlets only)
†footnote

	†			f	o	o	t	n	o	t	e
377	*357*	060	***377***	*000*	146	157	157	164	156	157	164	145

a ≠ α

a	<sp>			≠			<sp>			α
141	040	***377***	*041*	142	***377***	*000*	040	***377***	*046*	141

(2-byte stringlet only)
†footnote

			†		f		o		o		t
377	***377***	*000*	357	060	000	146	000	157	000	157	000	164
	n		o		t		e				
000	156	000	157	000	164	000	145					

a ≠ α

		a	<sp>	≠	<sp>	α
377	***377*** *000*	000 141	000 040	041 142	000 040	046 141

(1-byte stringlet followed by 2-byte)
cat

c				a		t	
103	***377***	***377***	*000*	000	101	000	124

(2-byte stringlet followed by 1-byte)
cat

			c			a	t
377	***377***	*000*	000	103	***377***	*000*	101	124

Appendix A
References

[1] American National Standards Institute. American National Standard Code for Information Interchange. X3.4-1977.

[2] Consultative Committee International Telegraph and Telephone. CCITT Draft Recommendations S.61. Character Repertoire and Coded Character Sets for International Teletex Service. (Geneva, 1980).

[3] Consultative Committee International Telegraph and Telephone. CCITT Draft Recommendation S.100. International Information Exchange for Interactive Videotex. (Geneva,1980).

[4] International Business Machines Corporation. 6670 Information Distribution Keyboards and Fonts, Office System 6. First edition. (1981)

[5] International Organization for Standardization. 7-Bit Coded Character Set for Information Processing Interchange. ISO 646-1973 (E).

[6] International Standards Organization. Information Processing–ISO 7-Bit and 8-Bit Coded Character Sets–Code Extension Techniques. Submitted on 02-04-1982. ISO/DIS 2022.2. ISO/TC 97.

[7] International Standards Organization. Code Extension Techniques for Use with the ISO 7-Bit Coded Character Set. ISO 2022-1973 (E).

[8] International Standards Organization. Extension of the Latin alphabet coded character set for bibliographic information interchange. ISO 5426-1980 (E).

[9] International Standards Organization. Greek alphabet coded character set for bibliographic information interchange. ISO 5428-1980 (E).

[10] International Standards Organization. Information Processing–Coded Character Sets for Text Communication–Part 1: General Introduction. Part 2: Latin Alphabetic and Non-Alphabetic Graphic Characters. ISO/DIS 6937/2. ISO/TC 97. Submitted on 02-25-1982.

[11] Japanese Industrial Standard. Code of the Japanese Graphic Character Set for Information Interchange. JIS C 6226-1978.

[12] Japanese Industrial Standard. Code Extension Techniques for Use with the Code for Information Interchange. JIS C 6228-1975.

[13] Shepherd, Walter. Shepherd's Glossary of Graphic Signs and Symbols. Dover Publications; New York; 1971.

[14] Xerox Corporation. Character Grouping Standard. Xerox System Integration Standard. El Segundo, California; (in preparation).

[15] Xerox Corporation. General Character Looks Standard. Xerox System Integration Standard. El Segundo, California; (in preparation).

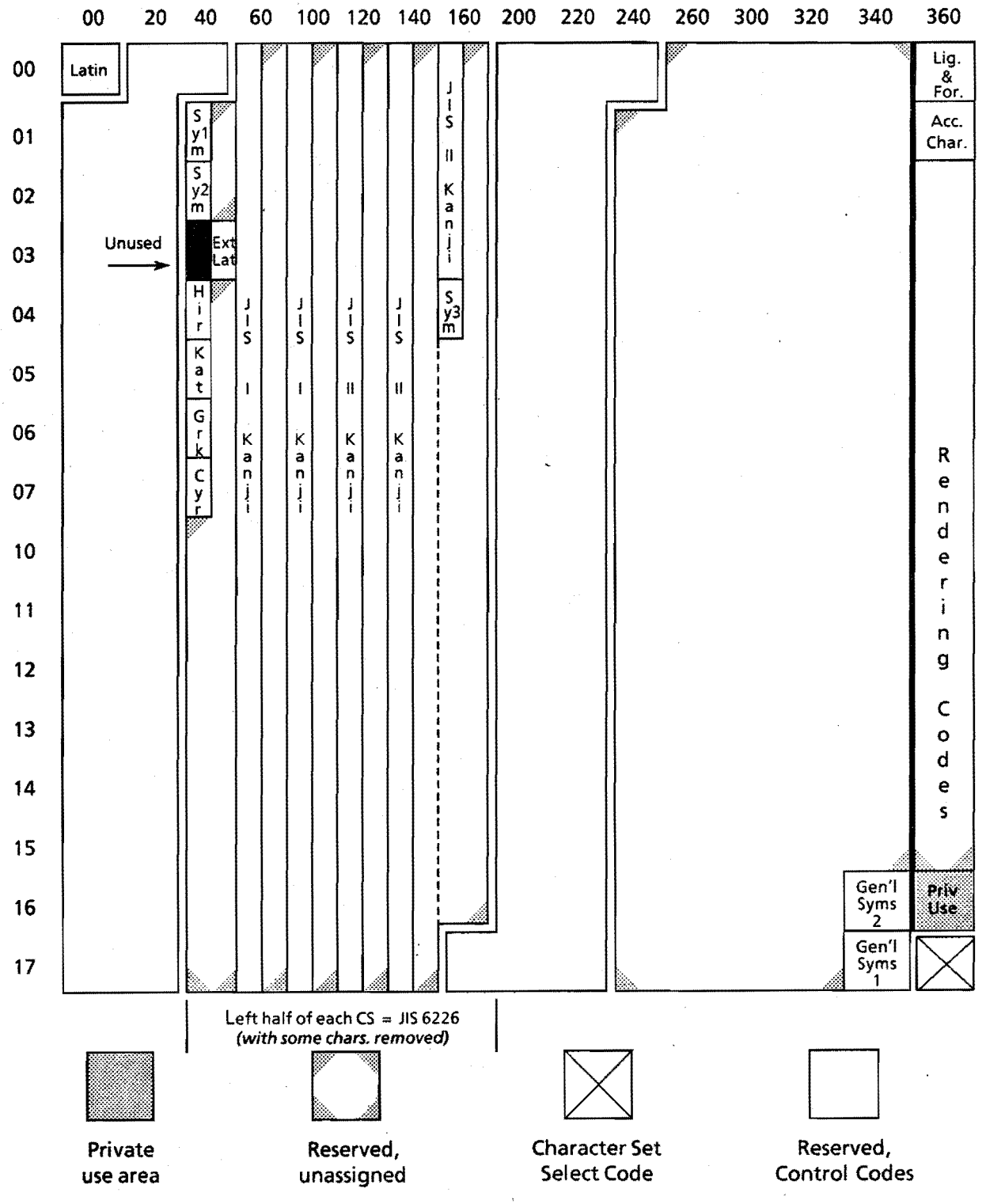
[16] Xerox Corporation. Information Processing System (IPS) Performance Specification. Xerox 860 Information Processing System. S804.

Appendix B
Character code charts

This Appendix contains the reference charts of the symbols for the non-kanji graphic character codes described in Section 3 and the rendering character codes described in Section 4.

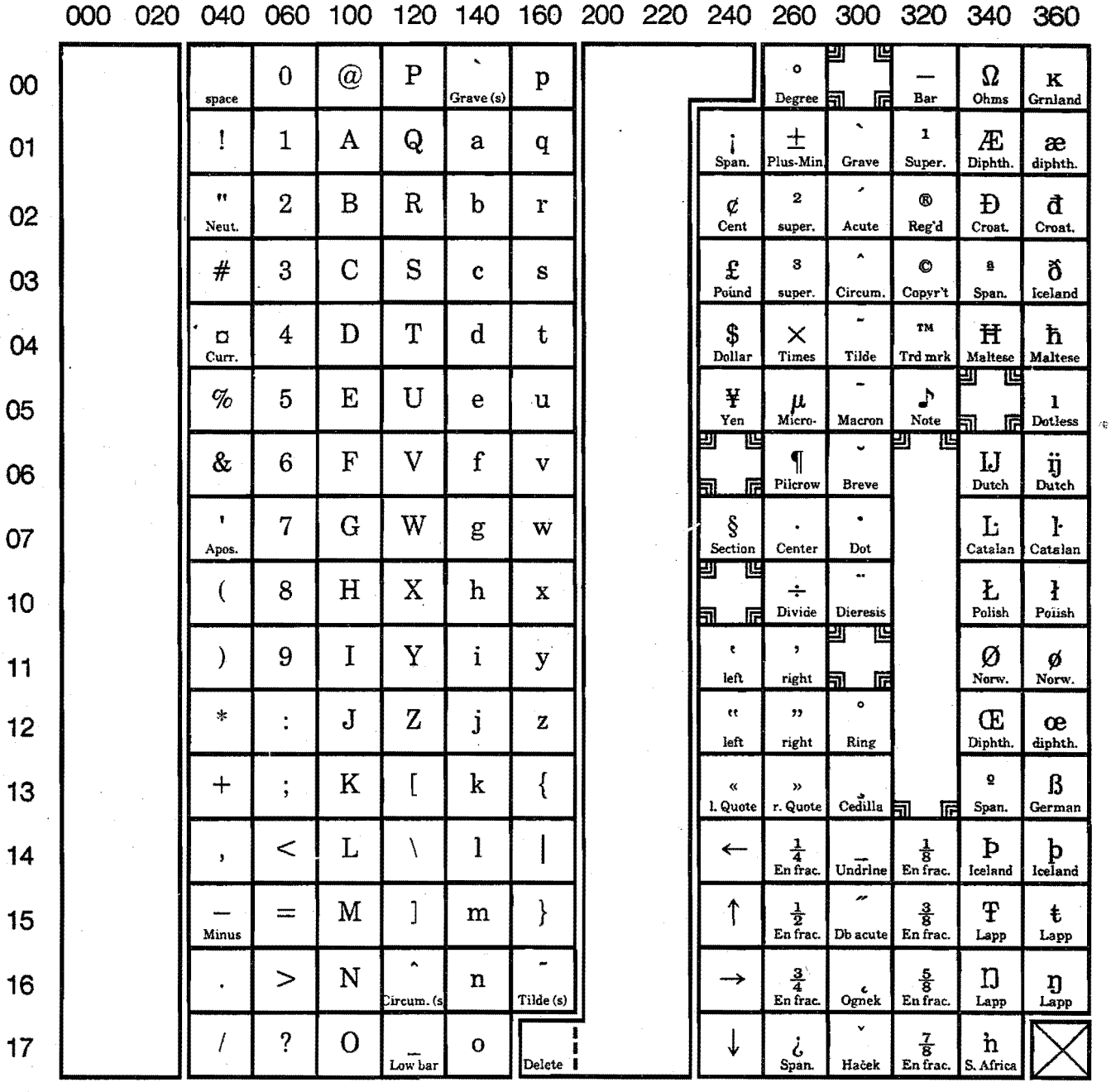
Xerox Character Set Allocation

Each square represents one Character Set (HIGH - order character code byte)



Xerox Character Set 0

ASCII/ISO/CCITT Roman Alphabet and Punctuation



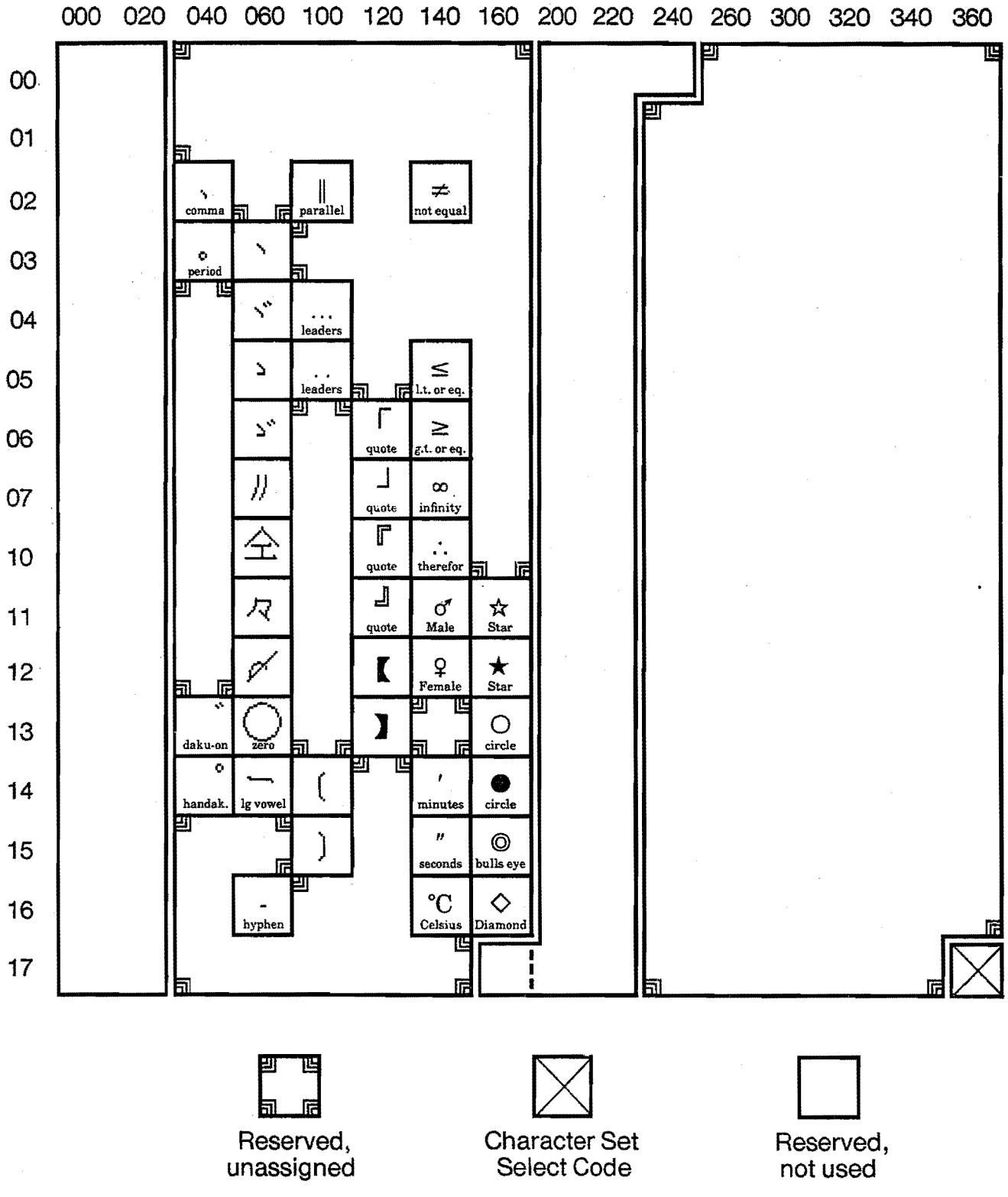
Reserved, unassigned

Character Set Select Code

Reserved, not used

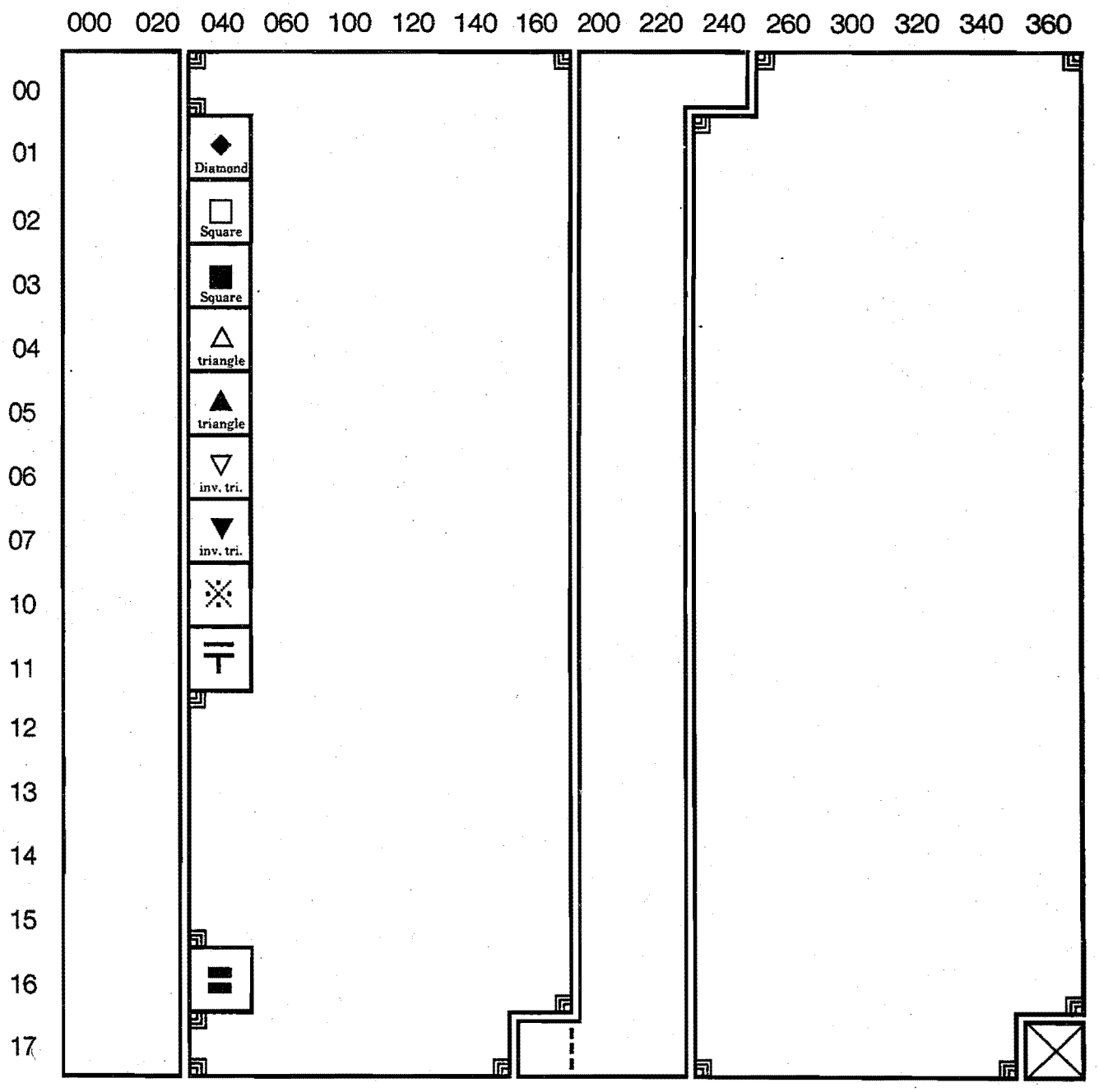
Xerox Character Set 41$_8$

JIS Symbols 1—Punctuation and Symbols not in CS 0



Xerox Character Set 42_8

JIS Symbols 2—Punctuation and Symbols not in CS 0



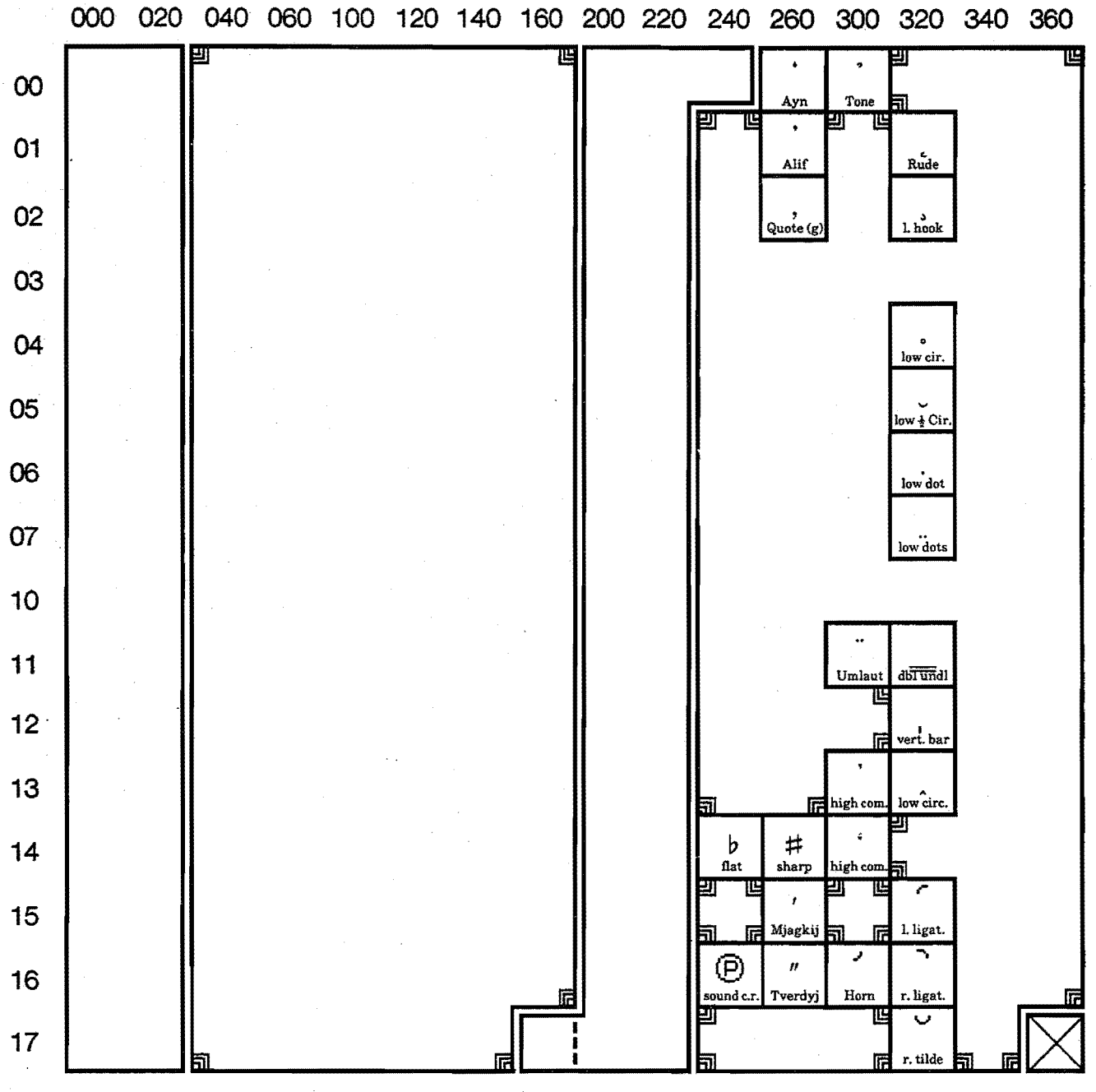
Reserved, unassigned

Character Set Select Code

Reserved, not used

Xerox Character Set 43_8

Extended Latin



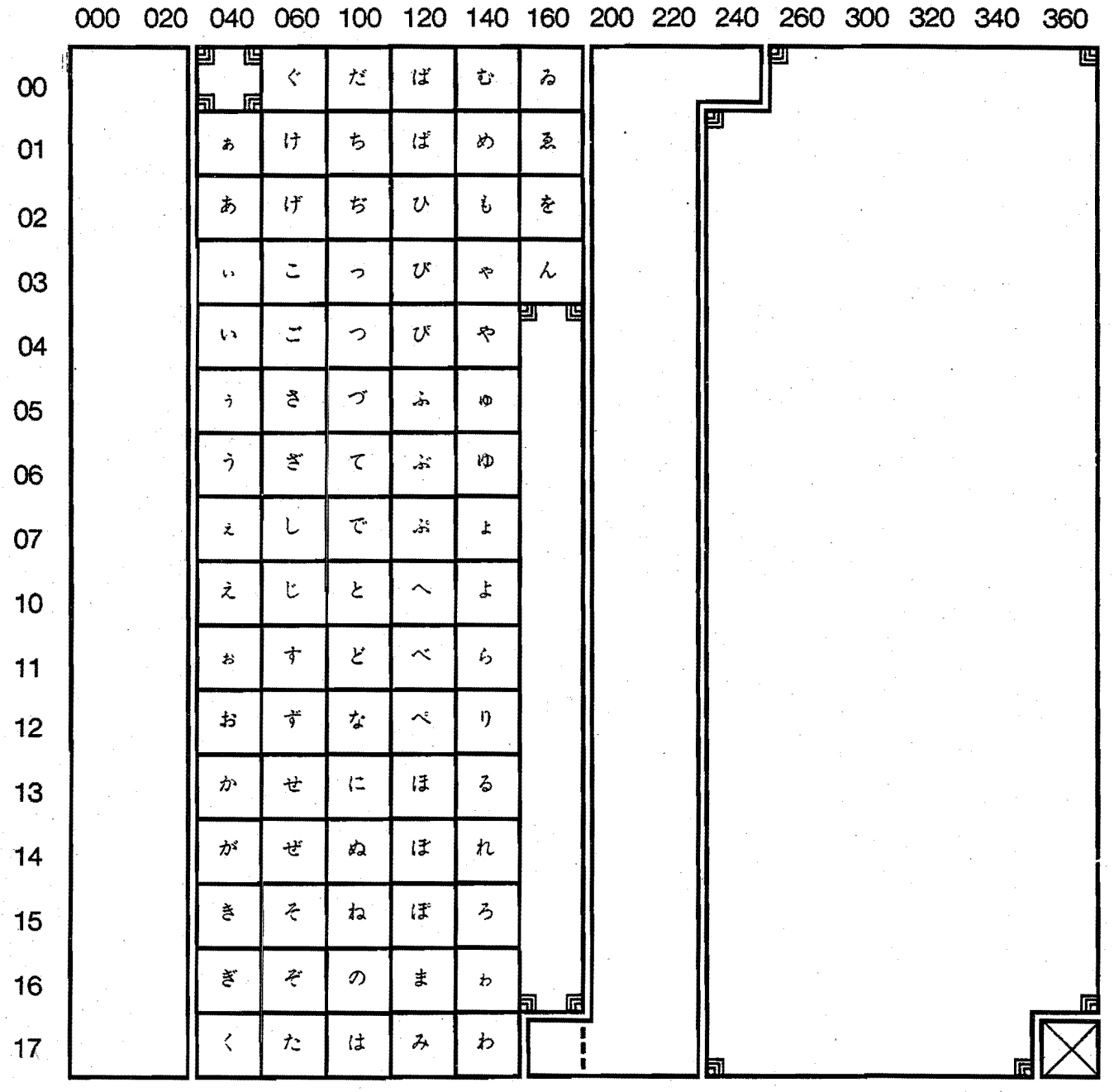
Reserved, unassigned

Character Set Select Code

Reserved, not used

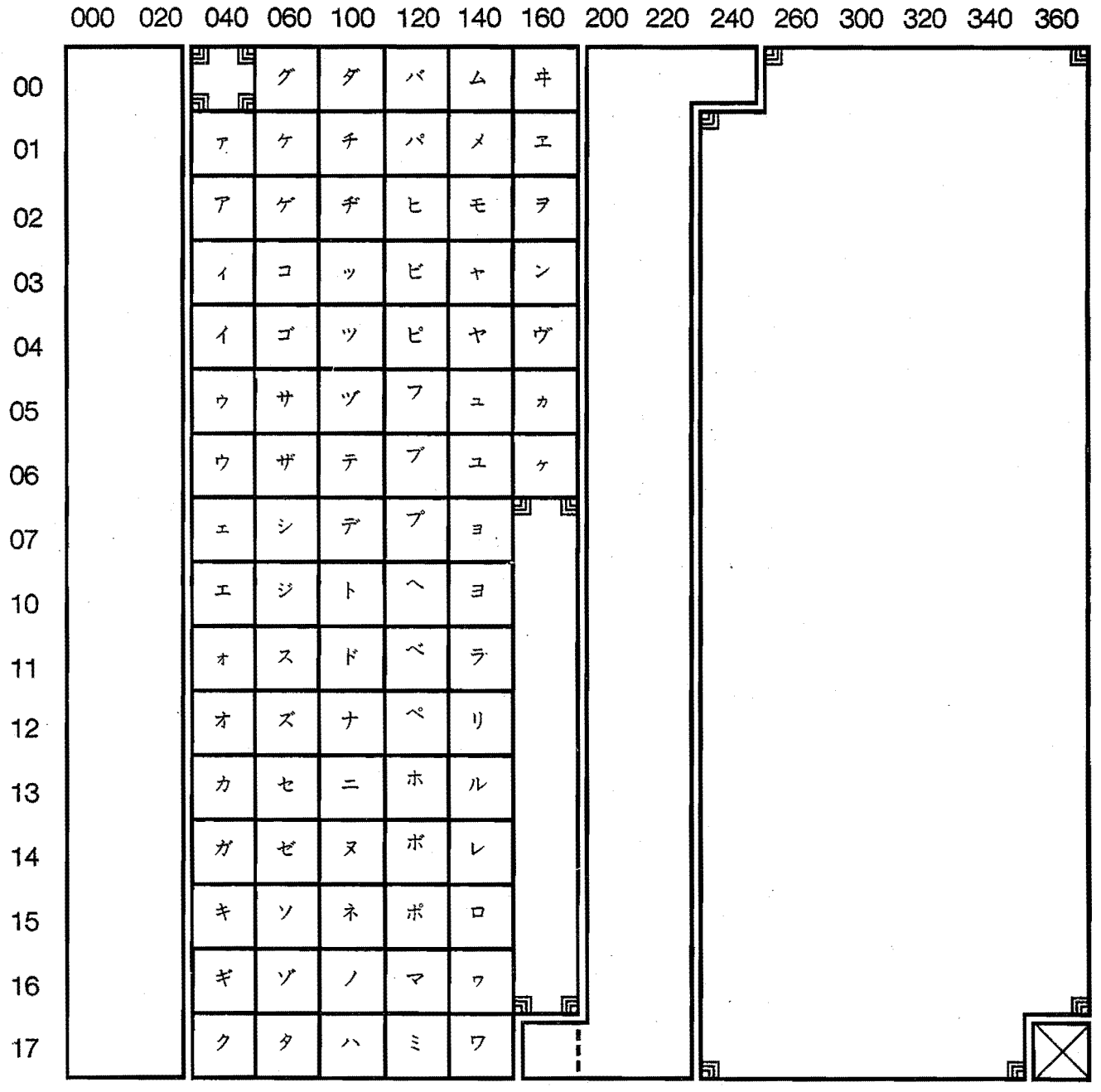
Xerox Character Set 44$_8$

JIS Hiragana



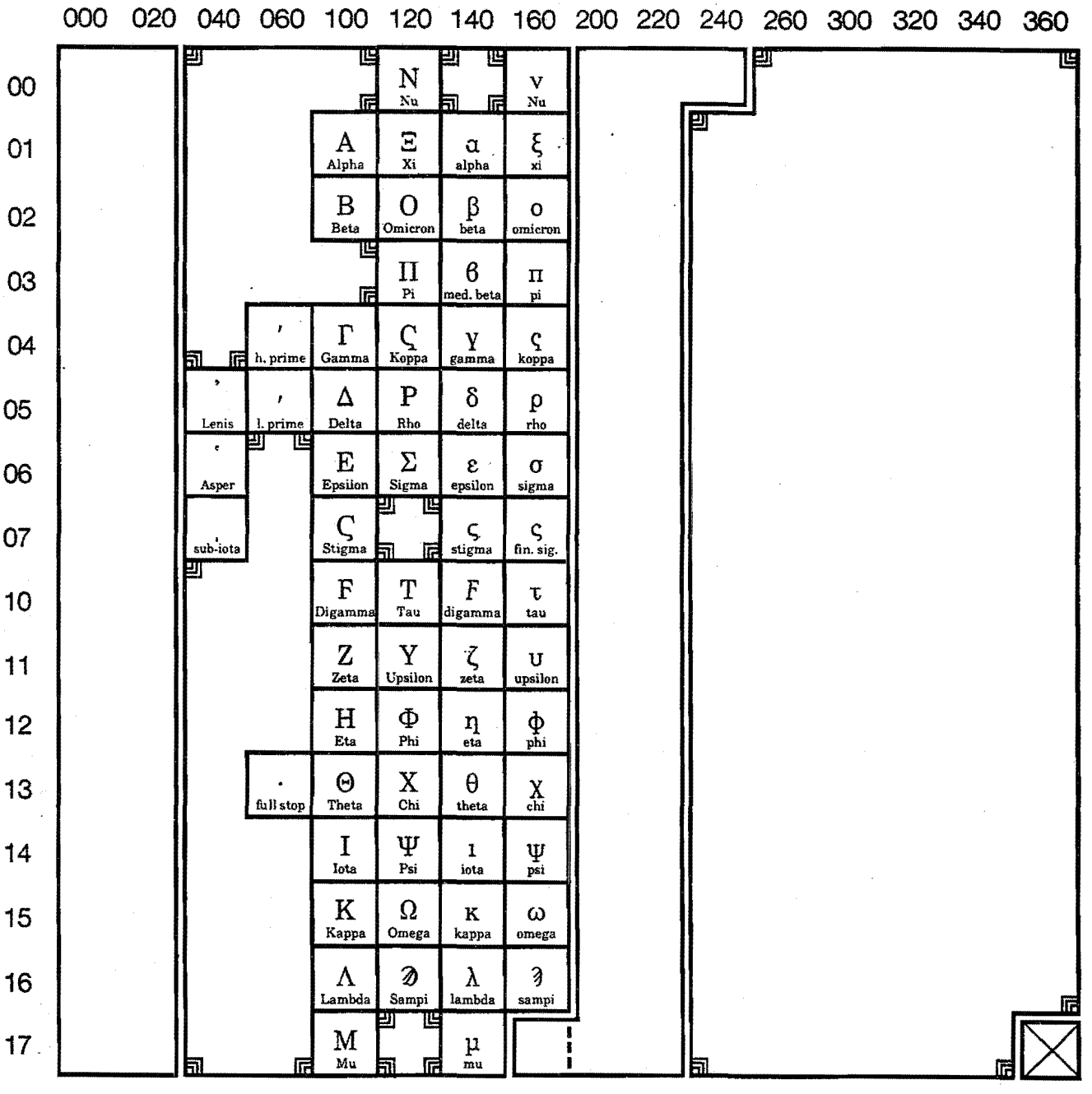
Xerox Character Set 45$_8$

JIS Katakana



Xerox Character Set 46_8

Greek



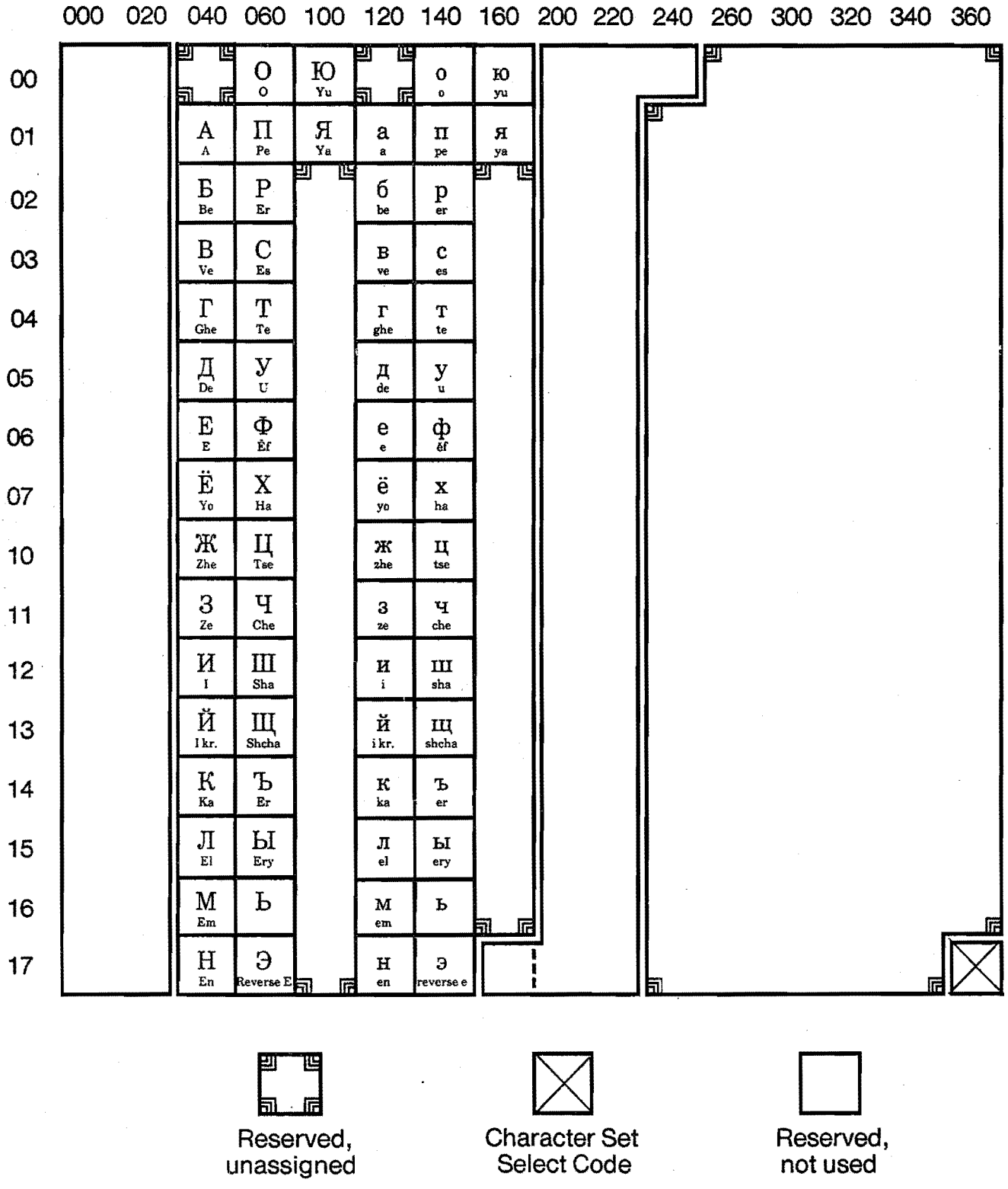
Reserved, unassigned

Character Set Select Code

Reserved, not used

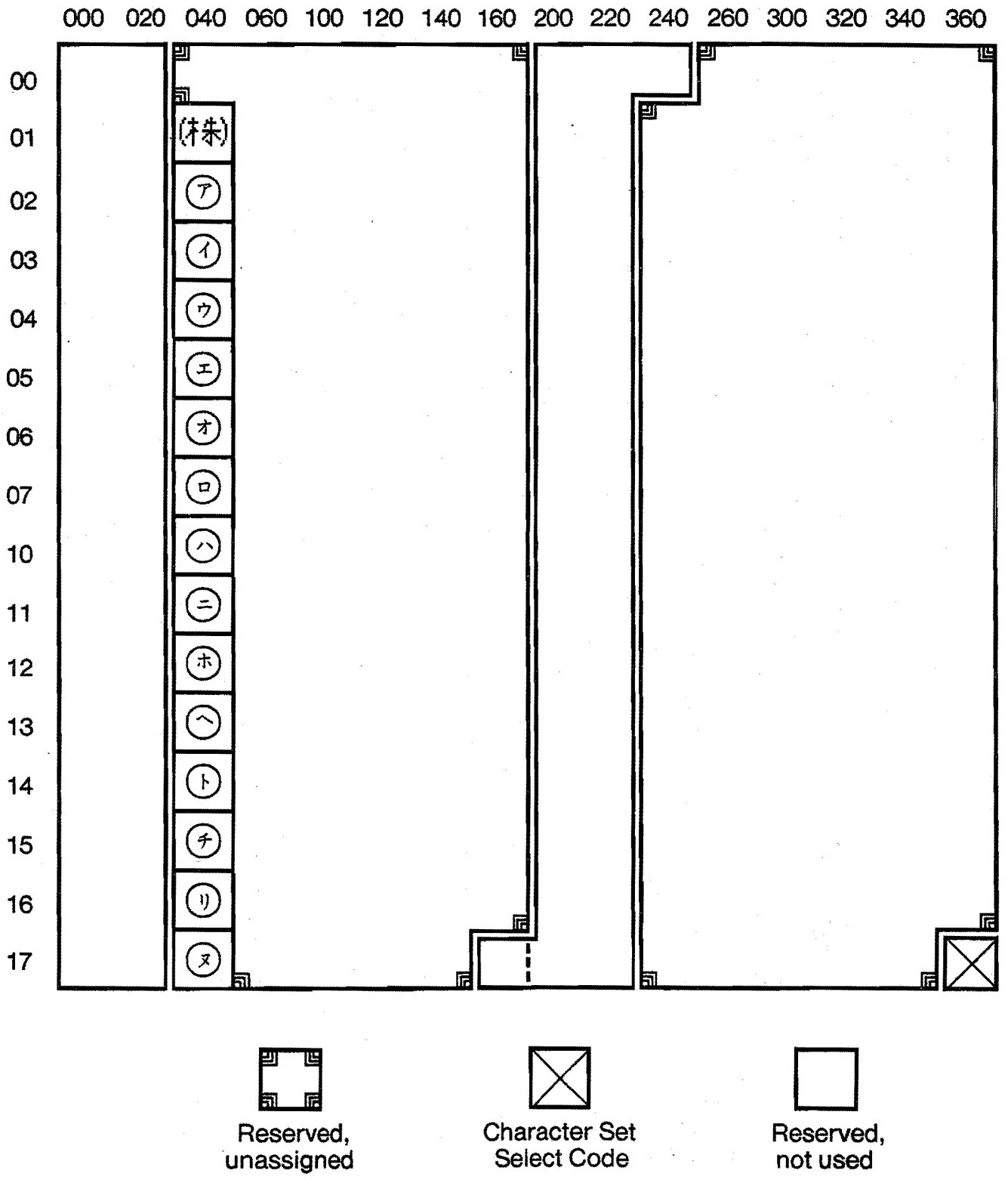
Xerox Character Set 47$_8$

Cyrillic



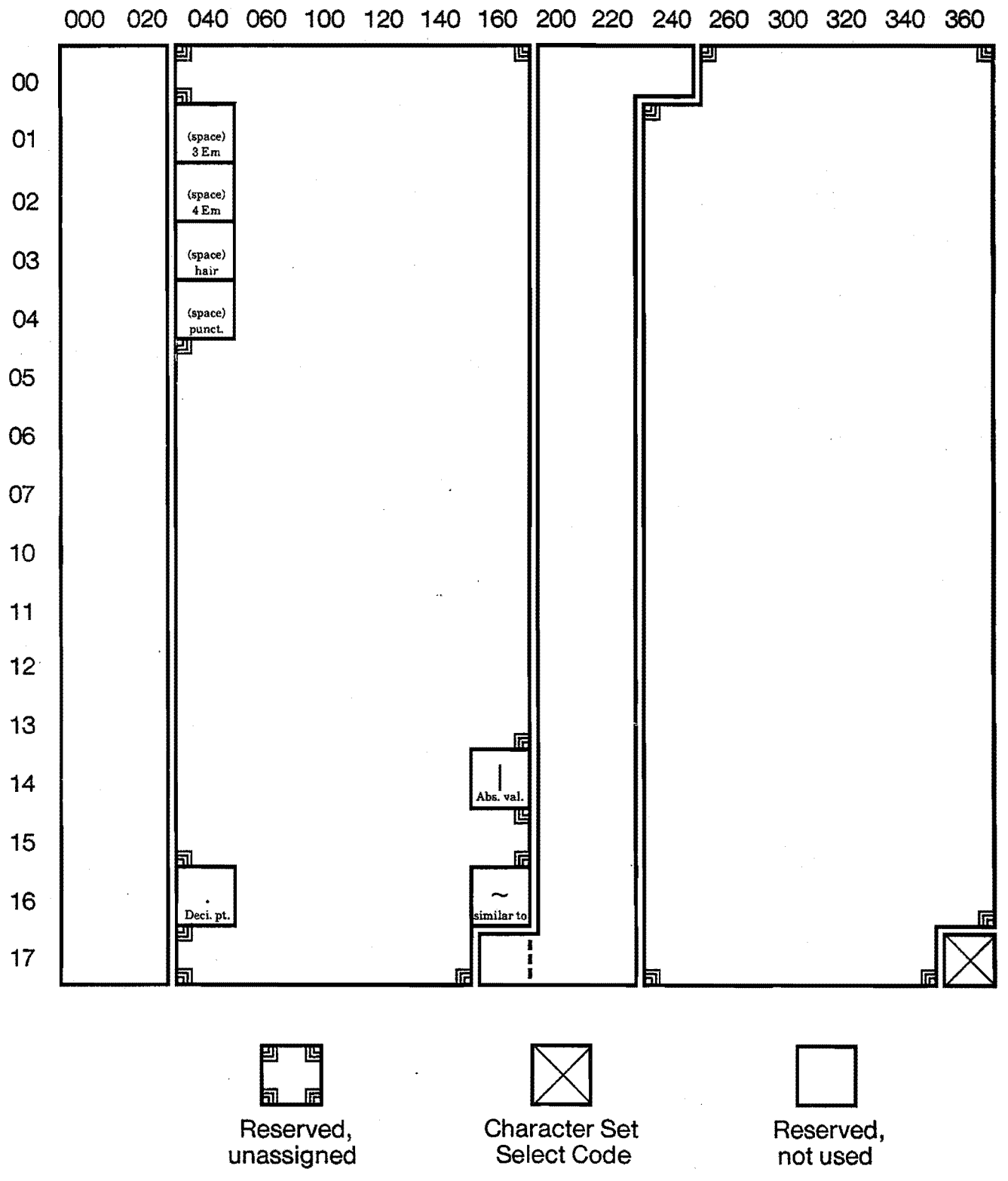
Xerox Character Set 164_8

Symbols 3—Miscellaneous Japanese Symbols



Xerox Character Set 356$_8$

General and Technical Symbols 2



Xerox Character Set 357$_8$

General and Technical Symbols 1

	000	020	040	060	100	120	140	160	200	220	240	260	300	320	340	360
00				† Dagger	% care of	⇌ rev. rea.	☒ ballot	⊥ perp.				⌜ l. ceil.	⌒ arc	✕ x-out	☠ poison	♒ Aquarius
01			(space) non-brk	‡ dbl Dag.	‰ per mil	⇆ rev. rea.	∅ null set	∝ prop.			₢ Brazil	⌝ r. ceil.	Ⅰ roman 1	①	| (form)	♓ Pisces
02			- non-brk	⟨ bra	⋘ much l.t.	↔	⊕ abs. +	≡ identical			ƒ Dutch	⌞ l. floor	Ⅱ roman 2	②	— (form)	♈ Aries
03			- discre.	⟩ ket	⋙ much g.t.	↝	⊖ abs. –	≐ def. =			₣ French	⌟ r. floor	Ⅲ roman 3	③	+ (form)	♉ Taurus
04			– en dash	☞ r. index	⊀ not l.t.	⊇	⊗ abs. ×	≟ ques. =			₱ Spanish	∃ exists	Ⅳ roman 4	④	| (form)	♊ Gemini
05			— em dash	☜ l. index	⊁ not g.t.	⊊	⊘ abs. ÷	∫ integral			₤ Cur. Comn.	∀ for all	Ⅴ roman 5	⑤	— (form)	♋ Cancer
06			‒ fig dash	⊢ l. perp	| divide	∩ inter.	● bullet	∮ cont. int.			$ Portug.	∧ and	Ⅵ roman 6	⑥	+ (form)	♌ Leo
07			' neut.	⊣ r. perp	∤ not div.	∪ union	∘ c. ring	≃ approx. =			ℵ gen. inf.	∨ or	Ⅶ roman 7	⑦	☉ Sun	♍ Virgo
10			” Germ. l.	⊨ l. 2 perp	// para.	⊇	ℏ Planck	≅ congr.			№ Number	▸ QED	Ⅷ roman 8	⑧	☽ ½ moon	♎ Libra
11			“ Germ. r.	⫤ r. 2 perp	∦ not par.	⊆	ℓ liter	≈ approx. =			℞ Take	∇ nabla	Ⅸ roman 9	⑨	☾ ¾ moon	♏ Scorpio
12			‹ l. qoute	⟦	∈ mem. of	⊃ contains	¬ not	Σ sum			℡ Teleph.	∂ deriv.	Ⅹ roman 10	⑩	☿ Mercury	♐ Sagitt.
13			› r. quote	⟧	∉ not mem.	⊂ cont. in	∣	∏ product			℥ dram	♪ hook	♠ spade	⑪	♃ Jupiter	♑ Capric.
14			(space) en quad	↖	∋ such that	⊉	∠ angle	√ root			ℂ cmplx	ϒ fork	♡ heart	⑫	♄ Saturn	☎ Teleph.
15			(space) em quad	↘	⇐ imp. by	⊄	∡ sph. ang.	∓ min.-plus			ℕ natrl	⊦ chair	◇ diamond	⑬	♅ Uranus	⅓ En frac.
16			(space) figure	↗	⇔ iff	⊅	∷ iden. .	shading			ℝ real	~ AC	♣ club	☮ Peace	♆ Neptune	⅔ En frac.
17			(space) thin	↙	⇒ implies	⊄	∵ because				ℤ integer	low bars	✓ check	☺ Smile	♇ Pluto	

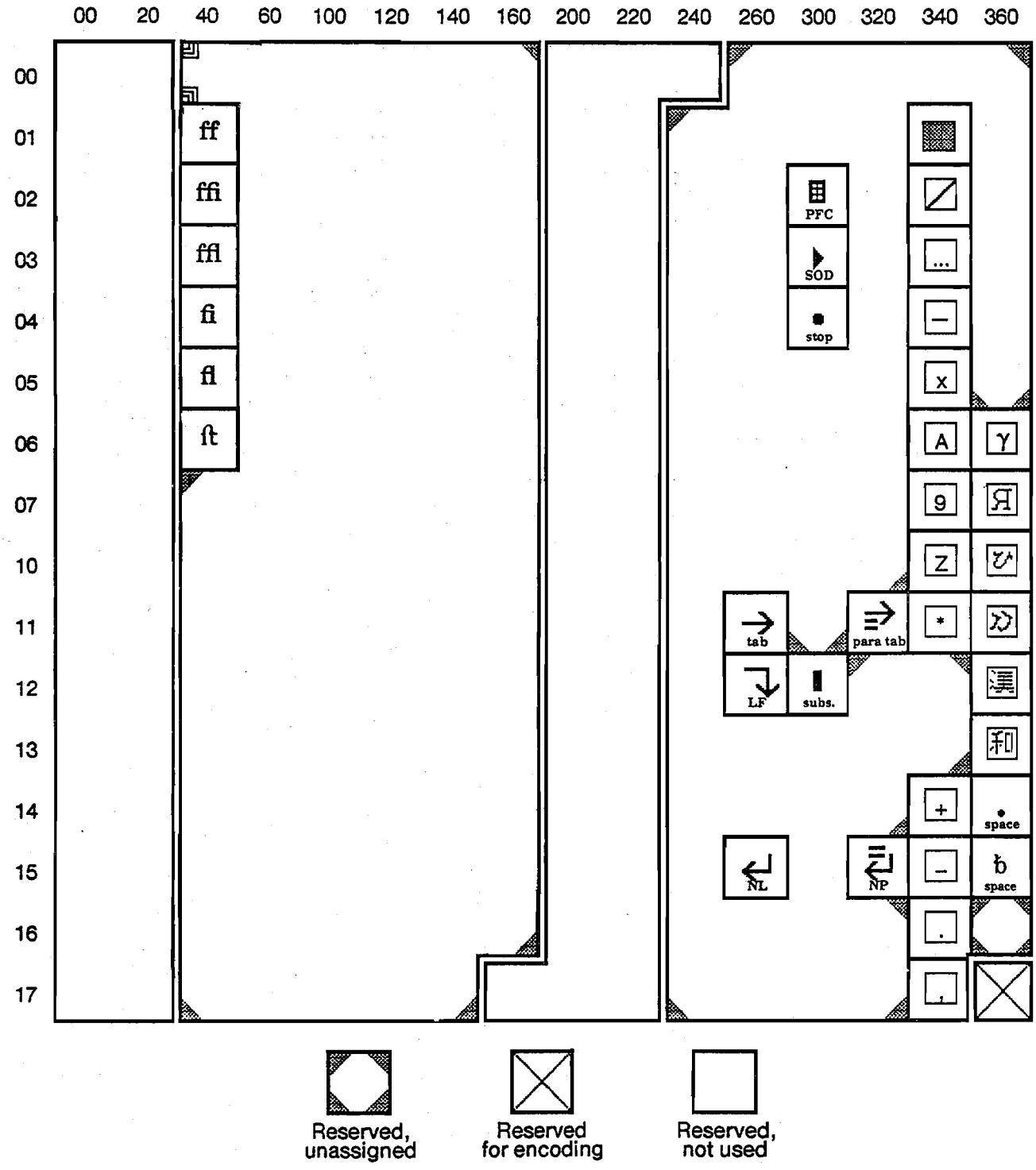
Reserved, unassigned

Character Set Select Code

Reserved, not used

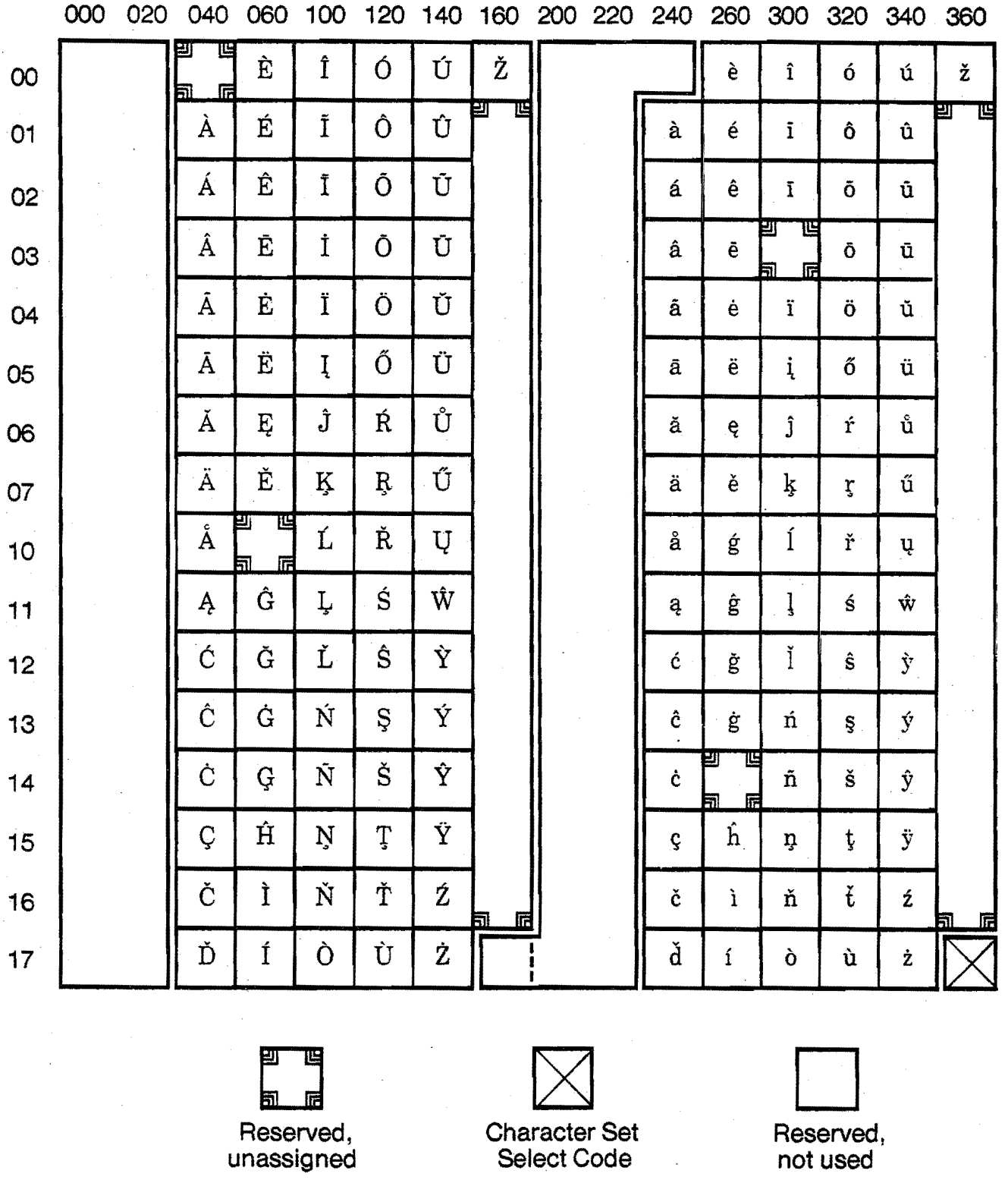
Xerox Rendering Set 360$_8$

Ligatures, Graphical Entities, and Field Format Symbols



Xerox Rendering Set 361$_8$

Accented Characters



Reserved, unassigned

Character Set Select Code

Reserved, not used

Appendix C
Technical questions

This appendix addresses a number of technical questions concerning the Xerox character codes listed in Section 3. Since the topics below are not necessarily related to each other, there may not be continuity in reading from one section of this appendix to the next.

The discussions below rely heavily on the two distinctions made in 1.5 and 1.6:

- Static text *representation* is distinguished from active text-manipulation *processes*.
- The *content* of text (*identity* of a character) is distinguished from its *appearance*.

The impact of these distinctions is often that questions which are raised in the context of character codes turn out to be answered by mechanisms which lie *outside* the realm of character codes, namely either text-manipulation processes or "*looks*" information that specifies how text will appear when rendered visible. A full answer to such questions would require a detailed description of those mechanisms, which lies outside the scope of the present standard.

C.1 Basic *looks*

Section 1.6 gave examples of different *looks* for the letter "C." Some of the appearance factors taken into account were: character size, character form (serif vs. sans-serif), character style (regular or roman vs. italic), character weight (light vs. bold), character baseline (normal vs. raised), and underlining. These are some of the most basic *looks*. Each can take on a number or *range* of values (e.g., there is a whole range of character weights, not merely light and bold.)

C.2 Superscripts and subscripts

Character baseline is a *looks* consideration. In the expression"$2x^2$," the two occurrences of the character "2" certainly represent the same numeral with different *looks* (size and baseline). This is especially obvious since computation might be performed involving these numbers. Therefore superscripted numerals should not have independent character codes.

The graphic character codes do include independent superscript "2" and "3" characters, but only to maintain Teletex coverage for Character Set 0. These codes should not be used

in a sophisticated text-processing system, and no such Xerox codes are provided for the other digits.

C.3 Emphasis

Various *looks* are used to emphasize a phrase of text: *italicizing*, **emboldening**, underlining, and (in some European languages) increased character s p a c i n g. These visual devices do not change the content of the emphasized text, only its appearance. The same could be said of emphasis via CAPITALIZATION as well (see the following).

C.4 Case

Historically, the lowercase Latin letters such as "a" developed as mere stylistic variants of the standard Latin capitals such as "A," in the same way that italic-style "*A*" did. Thus "case" is merely a difference of visual style, so it *should* be handled the same as any other *looks*.

Specifically, the lowercase "a" *ought to* be assigned the same "101_8" code as uppercase "A." Likewise for the even more troublesome "small-cap A." All are different appearances of the same character. In other words, a part of the code-vs.-*looks* chart *should* show:

Code(s)	*looks* #1	#2	#3	#4	#5	#6
101_8	A	a	A	*A*	a	A
102_8	B	b	B	*B*	b	B
103_8	C	c	C	*C*	c	C
...						
141_8	*<not used>*					
142_8	*<not used>*					
143_8	*<not used>*					

Treating case as a *looks* difference and not a code difference would yield more benefits than mere historical veracity. The word "cat" would have one unique internal representation instead of 8 case-dependent spellings. A single **Substitute** command that replaced the word "cat" with "dog" would also automatically replace "Cat" with "Dog." And the letter "a" would automatically sort together with "A" and before "Z," instead of requiring special treatment in the collating procedure.

Because of the untraditional nature of this analysis, the Xerox character codes do retain the ASCII standard assignment of 141_8 for "a," and a similar distinction for the Greek and Cyrillic alphabets. A sophisticated text-processing system would simply not use those

codes internally, but would convert to the existing interchange standards at its communication interfaces.

If one chooses to remain with the traditional problems of representing lowercase "a" internally as code 141_8, then small-cap "A" is best handled as a *looks* variant of "a." In other words, a part of the code-vs.-*looks* chart would show:

Code(s)	*looks* #1	#2	#3
141_8	a	a	A
142_8	b	b	B
143_8	c	c	C

A situation parallel to "case" occurs with the Japanese syllabaries *hiragana* and *katakana*, which ought to be represented as two different *looks* for a single set of *kana* symbols.

C.5 Ligatures

In standard printing it is common to express certain letter combinations by a single typographic object called a *ligature*, for example to render "f i" with "**fi**." This is modeled by designing the rendering process to detect the occurrence of the characters "f i" in sequence and render them visible with the "**fi**" ligature if appropriate. In other words, the code-vs.-*looks* chart may allow multiple codes per line, such as:

Code(s)	*looks* #1	#2	#3	#4	#5
146_8	f	f	f	f	f
$146_8 + 151_8$	fi	fi	fi	fi	fi

What is important here is that the ligature itself, i.e., the typographic object "**fi**," *is not a text character and does not have a graphic character code*. This means that the text *content* of a string like "fish" does not vary; it remains four characters, whether it is *rendered* with a ligature or not.

Although the ligature entity "**fi**" is not a text character, in practice it is convenient to be able to refer to it by a numerical code. For this sort of use, the block of numbers assigned to alternate representations, Character Sets 360_8 through 376_8, are excluded from use as graphic character codes. See Section 4 for the definition of rendering characters and code assignments.

C.6 European marked characters rendered via ligatures

The Xerox standard follows Teletex in representing European marked characters by *two* successive codes: the "mark" character followed by the "base" character. But good

typography demands that the common European marked characters such as c-cedilla "ç" be *rendered* by a single specially-drawn typographic object.

This situation is handled in precisely the same way as ligatures. The rendering process detects the occurrence of the characters "cedilla c" in sequence and renders them visible with the graphic "ç." In other words, a part of the code-vs.-*looks* chart should show:

Code(s)	*looks* #1
143_8	c
$313_8 + 143_8$	ç

What is important here is that the European marked character itself, i.e., the typographic object "ç," *is not a text character and does not have a graphic character code.* It is, however, a rendering character; see Section 4 of this standard. This permits adherence to the Teletex representation standard without loss of rendering quality.

This approach also solves the problem of whether the "marks" are visible or not when attached to uppercase letters. This is a matter of rendering style, not a difference of content. The marked uppercase letters can be rendered *either way*; but in any case their *content* is represented as mark plus base. So, a part of the code-vs.-*looks* chart should show:

Code(s)	*looks* #1	#2
141_8	a	a
$310_8 + 141_8$	ä	ä
101_8	A	A
$310_8 + 101_8$	Ä	A

C.7 Generalized *n*-ary ligatures

The process which renders text visible may be designed to parse it into substrings of arbitrary length, and then emit a single visible graphic corresponding to each substring. Typographic objects representing n successive characters are a simple generalization of ligatures.

The English alphabet requires at most two simple ternary ligatures ("ffi" and "ffl"), but in other languages generalized ligatures can play an invaluable role:

- The Arabic-based scripts are modeled on cursive handwriting, and as such there are numerous combinations of letters which are better represented as nicely-drawn ligatures than by composition out of components. The more ligatures used, the more legible the printing. High-quality printing may use up to 900 ligatures for an alphabet containing only 29 distinct characters! One of the ligatures is obligatory.

- Devanagari-based scripts such as Hindi have dozens of ligatures which are either obligatory or strongly preferred to the sequence of individual characters.

- Some scripts such as Hindi and Thai write certain vowels *before* the consonants that they phonetically *follow*. For example, the Hindi word pronounced "*hindu*," would in effect be written "*ihndu*." This behavior is best handled by having a ligature for the pair of characters "*h*" + "*i*" which visually appears as the form "*ih*."

- Korean writes letters in small 2-dimensional syllabic clumps, although the underlying phonetic string is of course linear. The total number of possible syllables is on the order of 2,200, each of which could be either represented by its own generalized ligature or composed from an alphabet of about 400 small syllable fragments.

C.8 Context-dependent letterforms

Closely related to *n*-ary ligatures are cases where the form of each letter depends on the identity of the letters surrounding it. These cases also require a parsing of the text by the rendering process, but unlike ligatures, the text characters are singly mapped into visible forms.

The most noteworthy case of context-dependent letterforms is normal English text, where the first character of each sentence is rendered with uppercase *looks* (see C.4). Another important application is Arabic-script languages, whose printing absolutely requires this capability. Other languages have a few context-dependent characters (Hindi, Greek, Hebrew).

C.9 Miscellaneous digraphs

A *digraph* is a typographic object that appears to be composed out of two letters. Aside from ligatures, there are other digraphs whose treatment is quite different.

Certain languages have digraph letters: Spanish has "ch" and "ll," and Hungarian has several. These objects *collate* as though they were single characters, yet otherwise appear to be treated as pairs of separate characters. It seems best to represent such digraphs as pairs of separate characters and let the collating process parse the text to find them. The Xerox character codes do include independent Dutch "ij," "IJ," and South African "ĥ" characters, primarily to maintain Teletex coverage for Character Set 0. The use of these codes is not necessarily recommended.

Other digraphs, notably "ae," "oe," and German "*ess-zed*" *do* differ graphically and otherwise from a pair of letters. The use of these individual codes in Character Set 0 is recommended.

C.10 Fractions

Fractions must be handled as unitary objects of some sort, since the formatting of the fraction applies to the fraction as a whole and not only to the individual digits within it. But fractions cannot possibly be given individual character codes, since they are infinite in number.

Therefore, fractions must necessarily be handled as non-textual objects of some kind. The Xerox character codes do include independent "1/2," "1/3," "2/3," "1/4," and "3/4" characters, but only to maintain coverage of Teletex and other code sets. These codes

should not be used in a sophisticated text-processing system, and very few such codes are provided for other fractions.

C.11 Logos, signatures, and other non-textual objects

There is no clear borderline between "text" and "graphics." The Xerox character codes arbitrarily include a few "pictorial" symbols, but policy is to exclude these in general. Pictorial entities which are not commonly used in running text should be handled by the machinery of graphics rather than the machinery of text. It is possible to construct an architecture where sequences of text may be interrupted by interceding non-textual objects such as "frames" for graphics.

The preference for not assigning character codes to arbitrary pictorial symbols applies to company logos and the signatures of individuals (see 4.2.2). Moreover, such items should be accompanied by security information (password, etc.) which places them outside the pale of ordinary text. On the other hand, if a company name *can* be written in ordinary letters, such as

XEROX,

then there is certainly no harm in doing so.

C.12 Rendering of "normally nonprinting" characters

The Xerox standard follows ISO in defining code (0 | 40_8) as "Space, (normally nonprinting)." To render the character 40_8 visible, perhaps with one of several alternate graphical entities, is a different *look* of the Space character, not a change in its identity. So, a code-vs.-*looks* chart for the Space character could show:

Code(s)	*looks* #1	#2	#3	#4
0_8 | 40_8		•	␣	▮

In the above visualization, either the Space character or another nonprinting character, the *picture* in each *looks* column is not itself a Space or even a graphic character at all. However, in practice it is convenient to be able to refer to each *picture* by a numerical code. Therefore each *picture* is assigned a code in Section 4.

To distinguish between a numeric code which identifies a substitute *picture* from a normal character code, the higher order byte of the substitute code is always larger than the higher order byte of the replaced character code. Substitute codes are defined as rendering character codes (4.1) when they can be algorithmically substituted for normal character codes. Rendering character codes use the block of numbers corresponding to Character Sets 360_8 through 375_8.

In addition to rendering character codes always having a larger high order byte, any given instance of a rendering type is assigned within a specified numeric code range. For rendering characters which are non-conventional representations of normally nonprinting characters, Character Set 360_8 Right contains all of the *looks* for making control codes and graphic characters such as Space visible.

By definition from 4.1, substitution is always in conformance with predetermined typographic, linguistic, or formatting rules. For rendering nonprinting characters visible

the set of <character code, character *looks*> pairs models these rules and always contains Character Set 360_8 as its high order byte. A code-vs-*looks* chart of the set of pairs for rendering Space visible would show:

Code(s)	looks #1	#2	#3	#4
$0_8 \mid 40_8$				
$360_8 \mid 374_8$		•		
$360_8 \mid 375_8$			♿	
$360_8 \mid 312_8$				▮

The graphical *picture* "▮" is defined in 4.3.1 to be the graphic entity representative of "SUBSTITUTE" character, while the other pictures "•" and "♿" are graphic entities representative of Space, type 1 and Space, type 2. Each picture in the chart provides an illustration of algorithmic substitution of a <character code, character *looks*> pair for the $<0_8 \mid 40_8$, Space> pair in accordance with a previously assumed set of rules. The example only illustrates the rendering of nonprinting characters with substitute pairs from Character Set 360_8 Right and does not imply nor recommend any particular set of rules.

C.13 Quads and spaces

In any given point-size of type, space is necessary in the line itself, in order to provide white space between letters, words, or sentences; for indentions; for centering of words in a measure; or spacing out partially filled lines of matter. Common widths for spaces are 2, 3, 4, 6, 12, and 18 points, with a 24-point unit in the larger body sizes. *En* and *em quads* are included in the assortment, but not universally. The Xerox standard follows established practice and provides a range of widths to achieve the closeness essential for pleasant reading.

The *em quad* and the *en quad* (which is half the width of the em) are standard spacing units, the em quad being a square space of any given point size of type. For example, a 12-point en quad is 12-point body-size and 6-point set size. Some printers in speaking of the en quad refer to it as the "nut quad" because of the similarity in name between the em and en quad.

Spaces smaller than an en quad are given in relation to the em quad. The *3-em space* is one-third the width of the em quad; the *4-em space* is one-fourth; and the *5-em space* one-fifth. A 3-em space, 3-to-the-em to be precise, is sometimes referred to as a *thick space*. Similarly, a 5-em or 5-to-the-em space is referred to as a *thin space*.

In sizes up to 18-point a space thinner than 5-em is called a *hairspace*. A *hairspace* is a fixed space having an assigned width of one point; however, a hairspace of two points is not uncommon.

The Xerox standard also provides for a fixed, but *device dependent space* which is equal in width to the comma, period, and semicolon. Being of the same width, this space is given the identity "punctuation space" and is assigned the code $(356_8 \mid 44_8)$.

C.14 Printwheels

Printwheel printing is somewhat at a loss to cope with the Xerox character world, with over 700 non-*kanji* characters, each of which can have an arbitrary number of *looks*. The key word here is *creative coping*: a division sign can be simulated as a colon overstruck with minus sign.

The process which maps text into printwheel spokes should therefore be able to map an arbitrary combination of characters plus *looks* into an arbitrary combination of printing actions. For example, a u-umlaut "ü" can be simulated as an umlaut mark overstruck with a "u." If the wheel does not have the mark character, then the "ü" can be alluded to via "ue" or even just plain "u." Again, a certain printwheel may have a large integral sign on one spoke and a small integral sign on another, and the mapping process must attend to the character *looks* information in order to pick the integral-sign spoke of the desired size.

Clearly there are no fixed rules to this game. What is important is that the mapping process have as much flexibility as possible—it cannot possibly be a simple one-to-one mapping between character codes and wheel spokes.

C.15 Kanji variations

Chinese and Japanese kanji characters are subject to special *looks* variations, above and beyond the usual parameters of size and style:

- Vertical vs. horizontal text
  (A few characters change their appearance, such as parentheses.)
- "Variant" character forms
  (Affects a few hundred characters.)
- Traditional vs. simplified character forms
  (Affects thousands of characters.)

These are all questions of *appearance* rather than *content*. Indeed, if we can say that one character is a "variant" or "simplified" form of another, then we have explicitly declared them to be *the same character*, and they should be given the same character code but different *looks*. Unfortunately, the information interchange code standards in Asia sometimes handle these distinctions by assigning separate character codes to the different forms.

In the Xerox character codes there are thousands of Japanese kanji characters which have the same appearance as Chinese kanji. For example, the Japanese character "*ichi*" is identical semantically, graphically, and historically to the Chinese character "*yi*." These two kanji are considered to be different for the same reason that letters like "A" are repeated in the Latin, Greek, and Cyrillic alphabets: they are different by virtue of context. Furthermore, it would be chaos to attempt to manage a system in which Japanese and Chinese kanji codes were shared, not to mention the effect on run-length for text in both languages. Note that this version of the standard does not make any code assignments for Chinese kanji.

C.16 Find and Substitute

Find and **Substitute** are not merely useful editing commands; they are also the most direct means by which a system user matches his understanding of text representation against the system's encoding. For example:

> The English and Russian alphabets both contain letters which have the visual appearance of "ABC," but in the case of Russian these letters correspond phonetically to "AVS." A user with a bilingual English-Russian document would hardly expect the English string "ABC" to **Find** the Russian string of the same appearance, since in the user's mind there is no relationship between the two.

In other words, when the user applies a **Find** command, his *expectation* of what is a reasonable match is a *de facto* definition of text identity.

C.17 Character identity

Despite the helpfulness of the **Find** command in defining a notion of text identity, it still offers only a *gedankenexperiment*, not a definitive answer to questions of the sort, "Should these two symbols be two different character codes, or the same character code with different *looks*?".

In fact, there *is* no source of a definitive answer to such questions; there are only carefully considered judgments of appropriateness and expediency. The listing of Xerox character codes in 3.2 is ultimately a record of such considered judgments. For example:

> It is clear enough that "hyphen" and "minus sign" are *not* the same semantic, hence these are assigned to two different Xerox character codes. The ASCII and ISO 55_8 was given the identity "minus" simply because that is more compatible with existing standards; "hyphen" is assigned the code (41_8 | 75_8).

In cases where the Xerox character code listing does not distinguish separate characters, it is worth considering the possibility that they are *looks* variants of each other. For example:

> It is clear enough that a "less-than-or-equal-to" sign that has one line on the bottom is just a *looks* variant of one that has two lines on the bottom. Hence, there is only one Xerox character code for the "less-than-or-equal-to" sign.

In cases where the Xerox character code listing does distinguish separate characters, a type designation is provided when the same semantic can be implied. For example:

> It is clear that the approximate signs "≃" and "≈" have similar meaning, but are assigned separate Xerox character codes. The code (357_8 | 167_8) for "≃" has been given the identity "approximately equal, type 1" and the code (357_8 | 171_8) has the identity "approximately equal, type 2."

The user of this standard must often apply his or her own judgment in finding the right combination of character identity and *looks* to meet the needs of a particular situation.

C.18 Keyboard input

The method of keyboard input should have no particular relationship to the assignment of character codes, but in the past there has been a tendency to assume that any entity that appears on a standard typewriter keyboard must be represented as a single character (for example, the Dutch "ij" digraph).

In fact the mapping of keystrokes to characters may be many-to-one or one-to-many. A many-to-one mapping is found in phonetic-based typing methods for Japanese and Chinese. A one-to-many mapping is found in Swedish, whose standard keyboard contains several European marked characters such as "ä" which are to be represented internally as two separate codes.

C.19 Collating

The term "collating" here simply means "sorting into alphabetical order." The numerical Xerox character codes obviously constitute a *de facto* linear ordering on the characters, and the alphabetic Xerox characters are of course placed in standard dictionary order insofar as possible.

However, the numerical Xerox code order simply cannot serve as a universal standard collating order. Even among European Latin-alphabet languages, certain characters occupy different alphabetical places in different languages. For example, the character o-umlaut ("ö") has different standard positions:

- In German: Mixed in with "o."
- In Hungarian: Between "o" and "p."
- In Swedish: At the end after "z."

It also turns out that most alphabetical orderings are not based on simple numerical single-character comparisons. For example, in Spanish "ch" is a letter coming between "c" and "d," in French the "oe" digraph is a single character but it sorts between "od" and "of," and in all European languages the uppercase and lowercase characters are sorted together.

Likewise, the phonetic ordering used in Japanese dictionaries is not a simple linear ordering of the *kana* characters, but instead involves sorting certain sets of *kana* together in a first pass and then discriminating them in a second pass (just as in collating English, if "A" and "a" are given separate codes, then they must first be sorted together and then later discriminated).

In the case of Japanese and Chinese *kanji*, there is no standard dictionary order at all.

The inescapable conclusion is:

- The *de facto* linear character ordering implied by the numerical Xerox character codes *cannot* be designed so as to carry the burden of the collating process. That being the case, there must exist a separate collating procedure for each language, which may in general be an arbitrarily complex algorithm.

C.20 Information interchange

The present standard applies directly to textual information interchange only insofar as it defines character codes which constitute a representation of text content. It does not define or imply any transmission, error-correction, or other communications protocols.

The process of information interchange at an interface with an external system generally involves a format conversion of some sort, which often includes code-shuffling as one aspect of the conversion (for example, between ASCII and EBCDIC). The character codes obviously cannot be chosen so as to eliminate all such code shuffling, especially for a multinational system that must interface with various conflicting information interchange standards.

This situation is quite analogous to the conclusion with regard to collating (see above):

> The numerical Xerox character codes *cannot* be designed so as to completely eliminate code shuffling as part of all format conversion processes. That being the case, there must exist a separate character conversion procedure for each information interchange standard, which may in general be an arbitrarily complex algorithm.

The goal, of course, is to *minimize* interface code shuffling, which is the reason that the Xerox character codes adhere to existing standards wherever possible, even at the cost of a very sparse utilization of the available code space.

C.21 Number of available codes

The basic Xerox code space is 16 bits, i.e., $[0 .. 177777_8]$, which in theory allows for 65,536 different characters; however, this space is very considerably diluted by the ISO-based format which excludes "control codes" $[1 .. 37_8]$ and $[177_8 .. 240_8]$ from being used in any byte.

For Japanese characters, the Xerox codes follow the JIS standard, which adopts the ISO format and further prohibits use of the high-order bit in any byte (except that the Xerox codes impose the Character Set Escape code 377_8). Thus, JIS limits itself to 94*94 = 8,836 total characters. There appear to remain 1,033 code spaces available to JIS for additional Japanese kanji under the current self-imposed regime. The current JIS codes cover all but the most obscure Japanese text, and only a highly-trained scholar could recognize all of the currently-specified JIS kanji characters. However, there are many thousands of obscure characters used in the names of people and places or archaic text, that are not now covered by JIS.

The Xerox code design makes no attempt to resolve this problem on behalf of JIS. JIS will undoubtedly address it in the future, and the Xerox codes will attempt to remain as consistent as possible with JIS' future handling of kanji code assignments. Certainly the most attractive code-extension possibility is an escape to a multi-byte encoding (see 6.4) for ultra-obscure kanji characters. In any case, the current limitations of the Xerox characters with regard to Japanese kanji are precisely those which are presently in force throughout Japan.

The situation for Chinese kanji characters is similar, but confused by the lack of a generally accepted character set. The total number of Chinese kanji characters is in excess of 50,000. However, most of these are extremely rare, and between 6,000 and 12,000 kanji characters will suffice for all but the most obscure Chinese text.

As for non-kanji characters, all but the most technical European-language text is already covered by the current Xerox character codes, and there are only a few significant non-European alphabets remaining to be added. Most European-language typewriting and information processing has survived so far with no more than a *dozen* or so "symbol" characters.

A reference such as *Shepherd's Glossary of Graphic Signs and Symbols* [13] lists only 6,188 symbol *applications*, but this represents a very much smaller number of distinct character codes, because many of these "symbols" are actually letters, and there are often many different applications of the same symbol (e.g., a circle). Likewise the number of different visual *appearances* of symbols is in no way constrained by the number of distinct symbol codes.

The Xerox code space range of character sets [241_8 .. 355_8] currently allows for the addition of approximately 14,500 new symbols and/or characters. This provision should be adequate for the foreseeable future without going to a code-extension scheme such as described in 6.4.

Appendix D
Differences between the Xerox Character Code and JIS 6226 standards

Deletions and additions have been made to the *Code of the Japanese Graphic Character Set for Information Interchange* standard, JIS C 6226-1978 (Japanese Industrial Standard) [11], to create the Xerox Character Code Standard. The differences are listed below for reference:

Deletions to the JIS Standard

Character	Xerox Code in Character Set0	JIS Code Set Row(1st)–Col.(2nd)
Space	40	1-1
!	41	1-10
#	43	1-84
%	45	1-83
&	46	1-85
' (apos)	47	1-39
(	50	1-42
)	51	1-43
*	52	1-86
+	53	1-60
,	54	1-4
- (minus)	55	1-61
.	56	1-5
/	57	1-31
0	60	3-16
1	61	3-17
2	62	3-18
3	63	3-19
4	64	3-20
5	65	3-21
6	66	3-22
7	67	3-23
8	70	3-24
9	71	3-25
:	72	1-7
;	73	1-8
<	74	1-67

Character	Xerox Code in Character Set 0	JIS Code Set Row(1st)–Col.(2nd)
=	75	1-65
>	76	1-68
?	77	1-9
@	100	1-87
A	101	3-33
B	102	3-34
C	103	3-35
D	104	3-36
E	105	3-37
F	106	3-38
G	107	3-39
H	110	3-40
I	111	3-41
J	112	3-42
K	113	3-43
L	114	3-44
M	115	3-45
N	116	3-46
O	117	3-47
P	120	3-48
Q	121	3-49
R	122	3-50
S	123	3-51
T	124	3-52
U	125	3-53
V	126	3-54
W	127	3-55
X	130	3-56
Y	131	3-57
Z	132	3-58
a	141	3-65
b	142	3-66
c	143	3-67
d	144	3-68
e	145	3-69
f	146	3-70
g	147	3-71
h	150	3-72
i	151	3-73
j	152	3-74
k	153	3-75
l	154	3-76
m	155	3-77
n	156	3-78
o	157	3-79
p	160	3-80
q	161	3-81

Character		Xerox Code in Character Set 0	JIS Code Set Row(1st)–Col.(2nd)
r		162	3-82
s		163	3-83
t		164	3-84
u		165	3-85
v		166	3-86
w		167	3-87
x		170	3-88
y		171	3-89
z		172	3-90
{		173	1-48
|	(bar)	174	1-35
}		175	1-49
~	(tilde)	176	1-33
¢	(cents)	242	1-81
£	(Pound)	243	1-82
$		244	1-80
¥	(yen)	245	1-79
§		247	1-88
‘	(left)	251	1-38
“	(left)	252	1-40
«		253	1-52
←	(left arrow)	254	2-11
↑	(up arrow)	255	2-12
→	(right arrow)	256	2-10
↓	(down arrow)	257	2-13
°	(degree)	260	1-75
±	(plus minus)	261	1-62
×	(mult.)	264	1-63
•	(centered dot)	267	1-6
÷	(div)	270	1-64
’	(right)	271	1-39
”	(right)	272	1-41
»		273	1-53
`	(grave)	301	1-14
´	(acute)	302	1-13
^	(circumflex)	303	1-16
¨	(diaeresis)	310	1-15
___	(underline)	314	1-18

Substitution within the JIS Standard
(Greek)

ISO 5428-1980 (E), Greek alphabet coded character set for bibliographic information has been substituted for the JIS C 6226-1978 Greek character set.

Addition to the JIS Standard
(Miscellaneous Japanese Symbols)

Character	Xerox Code in Character Set 164	JIS Code Set Row(1st)–Col.(2nd)
	41	84-1
	42	84-2
	43	84-3
	44	84-4
	45	84-5
	46	84-6
	47	84-7
	50	84-8
	51	84-9
	52	84-10
	53	84-11
	54	84-12
	55	84-13
	56	84-14
	57	84-15

Appendix E
Cross reference

Alphabet, English	See Alphabet, Roman
Alphabet, Greek	Character Set 46_8
Alphabet, Latin	See Alphabet, Roman
Alphabet, Roman	Character Set 0_8
Alternating current	$357_8 \mid 276_8$
Ampersand	$0_8 \mid 46_8$
And	$357_8 \mid 266_8$
Angle	$357_8 \mid 154_8$
Angle, Spherical	$357_8 \mid 155_8$
Apostrophe	$0_8 \mid 47_8$
Approximately equal, Type 1	$357_8 \mid 167_8$
Approximately equal, Type 2	$357_8 \mid 171_8$
Aquarius	$357_8 \mid 360_8$
Arc	$357_8 \mid 300_8$
Aries	$357_8 \mid 362_8$
Arrow, Curly	$357_8 \mid 123_8$
Arrow, Double	$357_8 \mid 122_8$
Arrow, Double, Back	$357_8 \mid 115_8$
Arrow, Double, Double	$357_8 \mid 116_8$
Arrow, Double, Left	See Arrow, Double, Back
Arrow, Double, Right	$357_8 \mid 117_8$
Arrow, East	$0_8 \mid 256_8$
Arrow, East, Circled	$357_8 \mid 333_8$
Arrow, East-then-South, Circled	$357_8 \mid 334_8$
Arrow, North	$0_8 \mid 255_8$
Arrow, NorthEast	$357_8 \mid 76_8$
Arrow, NorthWest	$357_8 \mid 74_8$
Arrow, South	$0_8 \mid 257_8$
Arrow, South-then-West, Circled	$357_8 \mid 335_8$
Arrow, SouthEast	$357_8 \mid 75_8$
Arrow, SouthWest	$357_8 \mid 77_8$
Arrow, West	$0_8 \mid 254_8$
Asterisk	$0_8 \mid 52_8$
At sign (Commercial at sign)	$0_8 \mid 100_8$
Backslash	See Slant, Reverse
Ballot box	$42_8 \mid 42_8$
Ballot box, Checked	$357_8 \mid 140_8$
Bar, Low	$0_8 \mid 137_8$
Bar, Low, Double	$357_8 \mid 277_8$
Bar, Vertical	$0_8 \mid 174_8$
Bar, Vertical, Broken	$357_8 \mid 153_8$
Bar, Vertical, Double	See Parallel sign
Because	$357_8 \mid 157_8$
Bra	$357_8 \mid 62_8$
Brace, Closing (Right)	$0_8 \mid 175_8$
Brace, Opening (Left)	$0_8 \mid 173_8$
Bracket, Closing (Right)	$0_8 \mid 135_8$
Bracket, Lenticular, Left, Black (Japanese)	$41_8 \mid 132_8$
Bracket, Lenticular, Left, White (Chinese)	$357_8 \mid 72_8$
Bracket, Lenticular, Right, Black (Japanese)	$41_8 \mid 133_8$
Bracket, Lenticular, Right, White (Chinese)	$357_8 \mid 73_8$

Bracket, Opening (Left)	$0_8 \mid 133_8$
Breve (accent)	$0_8 \mid 306_8$
Bullet, Centered	$357_8 \mid 146_8$
Bull's eye	See Circle,White, Two, Concentric
Cancer	$357_8 \mid 365_8$
Capricorn	$357_8 \mid 373_8$
Care of	$357_8 \mid 100_8$
Caron	See Hachek
Cedilla (Undermark)	$0_8 \mid 313_8$
Ceiling, Left	$357_8 \mid 260_8$
Ceiling, Right	$357_8 \mid 261_8$
Cent sign	$0_8 \mid 242_8$
Check mark	$357_8 \mid 337$
Circle, Black	$41_8 \mid 174_8$
Circle, White	$41_8 \mid 173_8$
Circle, White, Two, Concentric	$41_8 \mid 175_8$
Circled number	See Number, Circled
Circumflex Accent (spacing)	$0_8 \mid 136_8$
Circumflex (accent)	$0_8 \mid 303_8$
Circumflex Undermark (African languages)	$41_8 \mid 333_8$
Clubs	$357_8 \mid 316_8$
Colon	$0_8 \mid 72_8$
Comma	$0_8 \mid 54_8$
Complex number	See Number, Complex
Congruent	See Isomorphic
Contain, Does not	See Does not contain
Contained in, Type 1	$337_8 \mid 125_8$
Contained in, Type 2	$357_8 \mid 133_8$
Contained in or equals	$357_8 \mid 131_8$
Contained in, Not	See Is not contained in
Contained in or equals, Not	See Neither contained in nor is equal to
Contains or equals, Not	See Neither contains nor is equal to
Contains, Type 1	$357_8 \mid 124_8$
Contains, Type 2	$357_8 \mid 132_8$
Contains as a member	See Such that, Type 1
Contains or equals	$357_8 \mid 130_8$
Copyright sign	$0_8 \mid 323_8$
Copyright statement, Sound recording	$43_8 \mid 256_8$
Cruzeiro (Brazilian)	$357_8 \mid 241_8$
Currency sign, European	$357_8 \mid 245_8$
Currency symbol, Brazilian	See Cruzeiro
Currency symbol, Dutch	See Florin
Currency symbol, General (International)	$0_8 \mid 44_8$
Currency symbol, Portugese	See Escudo
Currency symbol, Spanish	See Pesetas
Currency symbol, Yen (Japanese)	$0_8 \mid 245_8$
Cyrillic	See Alphabet, Cyrillic

Face, Smile	See Smile Face
Feet sign	See Minutes sign
Female	41_8 | 152_8
Figure dash	See Dash, Figure
Figure space	See Space, Figure
Five	0_8 | 65_8
Flat, Musical	43_8 | 254_8
Floor, Left	357_8 | 262_8
Floor, Right	357_8 | 263_8
Florin	357_8 | 242_8
For all	357_8 | 265_8
Four	0_8 | 64_8
Fraction, Five eighths	0_8 | 336_8
Fraction, One eighth	0_8 | 334_8
Fraction, One half	0_8 | 275_8
Fraction, One third	357_8 | 375_8
Fraction, One quarter	0_8 | 274_8
Fraction, Seven eighths	0_8 | 337_8
Fraction, Three eighths	0_8 | 335_8
Fraction, Three quarters	0_8 | 276_8
Fraction, Two thirds	357_8 | 376_8
Francs	357_8 | 243_8
Gemini	357_8 | 364_8
General currency symbol	0_8 | 250_8
Grave Accent (spacing)	0_8 | 140_8
Grave (accent)	0_8 | 301_8
Greater than	0_8 | 76_8
Greater than, Much	357_8 | 103_8
Greater than, Not	357_8 | 105_8
Greater than or equal to	41_8 | 103_8
Greek	See Alphabet, Greek
Guilder	See Florin
Guillemet, Left (European quotation mark)	See Quote, Double, Left, European
Guillemet, Right (European quotation mark)	See Quote, Double, Right, European
Guillemet, Single...	See Quote, Single, ..., European
h with stroke, Lowercase (Maltese)	0_8 | 364_8
H with stroke, Uppercase (Maltese)	0_8 | 344_8
Hachek (accent)	0_8 | 317_8
Happy Face	See Smile face
Hat	See Circumflex
Hearts	357_8 | 314_8
Hiragana	44_8 | starting at 41_8
Hyphen	41_8 | 76_8
Hyphen, Discretionary	357_8 | 43_8
Hyphen, Non-breaking	357_8 | 42_8
i, dotless, Lowercase (Turkish)	0_8 | 365_8
Identically equal	357_8 | 162_8
Identifier	357_8 | 156_8

Iff	See Arrow, Double, Double
ij digraph, Lowercase (Dutch)	$0_8 \mid 366_8$
IJ digraph , Uppercase (Dutch)	$0_8 \mid 346_8$
Implies	See Arrow, Double, Right
Inches sign	See Seconds sign
Index, Left-pointing	$357_8 \mid 65_8$
Index, Right-pointing	$357_8 \mid 64_8$
Infinity	$41_8 \mid 147_8$
Infinity, Generic	$357_8 \mid 247_8$
Integer	$357_8 \mid 257_8$
Integral	$357_8 \mid 165_8$
Integral, Contour	$357_8 \mid 166_8$
Intersection	$357_8 \mid 126_8$
Is a member of	$357_8 \mid 112_8$
Is implied by	See Arrow, Double, Back
Is not a member of	$357_8 \mid 112_8$
Is not contained in	$357_8 \mid 137_8$
Isomorphic	$357_8 \mid 170_8$
Is proportional to	$357_8 \mid 161_8$
Jupiter	$357_8 \mid 353_8$
k, Lowercase (Greenlandic)	$0_8 \mid 360_8$
Katakana	Character Set 45_8
Ket	$357_8 \mid 63_8$
l with middle dot, Lowercase (Catalan)	$0_8 \mid 367_8$
L with middle dot, Uppercase (Catalan)	$0_8 \mid 347_8$
l with stroke, Lowercase (Polish)	$0_8 \mid 370_8$
L with stroke, Uppercase (Polish)	$0_8 \mid 350_8$
Latin	See Alphabet, Roman
Leader, two-dot	$41_8 \mid 105_8$
Leader, three-dot	$41_8 \mid 104_8$
Leo	$357_8 \mid 366_8$
Less than	$0_8 \mid 74_8$
Less than, Much	$357_8 \mid 102_8$
Less than, Not	$357_8 \mid 104_8$
Less than or equal to	$41_8 \mid 145_8$
Letters	See Alphabets
Libra	$357_8 \mid 370_8$
Line, Horizontal, Thick	$357_8 \mid 342_8$
Line, Horizontal, Thin	$357_8 \mid 345_8$
Line, Intersecting, Thick	$357_8 \mid 343_8$
Line, Intersecting, Thin	$357_8 \mid 346_8$
Line, Vertical, Thick	$357_8 \mid 341_8$
Line, Vertical, Thin	$357_8 \mid 344_8$
Liter	$357_8 \mid 151_8$
Macron (accent)	$0_8 \mid 305_8$
Male	$41_8 \mid 151_8$
Mars	See Male
Member	See Is a member of

Member, contains as a	See Such that
Member, Not	See Is not a member of
Micro sign	$0_8 \mid 265_8$
Milreis	See Currency sign
Minim	See Scorpius
Minus or plus	$357_8 \mid 175_8$
Minus sign	$0_8 \mid 55_8$
Minutes sign	$41_8 \mid 154_8$
Moon, First quarter	$357_8 \mid 350_8$
Moon, Third quarter	$357_8 \mid 351_8$
Much greater than	See Greater than, Much
Much less than	See Less than, Much
Multiply sign	$0_8 \mid 264_8$
Musical Flat	$43_8 \mid 254_8$
Musical Note	$0_8 \mid 325_8$
Musical Sharp	$43_8 \mid 274_8$
Mercury	$357_8 \mid 352_8$
n with apostrophe, Lowercase (South African)	$0_8 \mid 357_8$
Nabla	$357_8 \mid 271_8$
Natural Number	See Number, Natural
Neither contained in nor is equal to	$357_8 \mid 135_8$
Neither contains nor is equal to	$357_8 \mid 134_8$
Neptune	$357_8 \mid 356_8$
Nine	$0_8 \mid 71_8$
Not	$357_8 \mid 152_8$
Not sign	See Approximately
Not greater than	See Greater than, Not
Not less than	See Less than, Not
Not parallel	$357_8 \mid 111_8$
Note, Music	$0_8 \mid 325_8$
Null set	$357_8 \mid 141_8$
Number	$357_8 \mid 250_8$
Number, Circled	357_8 | starting at 321_8
Number, Complex	$357_8 \mid 254_8$
Number, Natural	$357_8 \mid 255_8$
Number, Real	$357_8 \mid 256_8$
Number sign	$0_8 \mid 43_8$
Numeral, Roman, Fixed-pitch	357_8 | starting at 301_8
o with slash, Lowercase (Norwegian, Danish)	$0_8 \mid 371_8$
O with slash, Uppercase (Norwegian, Danish)	$0_8 \mid 351_8$
OCR chair	$357_8 \mid 275_8$
OCR fork	$357_8 \mid 274_8$
OCR hook	$357_8 \mid 273_8$
oe digraph, Lowercase	$0_8 \mid 372_8$
OE digraph, Uppercase	$0_8 \mid 352_8$
Ogonek undermark = Polish hook	$0_8 \mid 316_8$
Ohm sign	$0_8 \mid 340_8$
One	$0_8 \mid 61_8$
Or	$357_8 \mid 267_8$
Ordinal indicator, Feminine (Spanish)	$0_8 \mid 343_8$

Ordinal indicator, Masculine (Spanish)	$0_8 \mid 353_8$
Over-dot (accent)	$0_8 \mid 307_8$
Over-ring (accent)	$0_8 \mid 312_8$
Paragraph sign	$0_8 \mid 266_8$
Parallel, Not	See Not parallel
Parallel sign, Type 1	$41_8 \mid 102_8$
Parallel sign, Type 2	See Slant, Double
Parenthesis, Closing	$0_8 \mid 51_8$
Parenthesis, Opening	$0_8 \mid 50_8$
Partial derivative	See Derivative, Partial
Peace symbol	$357_8 \mid 336_8$
Percent sign	$0_8 \mid 45_8$
Per mill sign	See Per thousand sign
Per thousand sign	$357_8 \mid 101_8$
Period	$0_8 \mid 56$
Perp, Left	See Perpendicular, Left
Perp, Left, Type 2	See Perpendicular, Left, Double
Perp, Right	See Perpendicular, Right
Perp, Right, Type 2	See Perpendicular, Right, Double
Perpendicular	$357_8 \mid 160_8$
Perpendicular, Left	$357_8 \mid 66_8$
Perpendicular, Left, Double	$357_8 \mid 70_8$
Perpendicular, Right	$357_8 \mid 67_8$
Perpendicular, Right, Double	$357_8 \mid 71_8$
Pesetas (Spanish)	$357_8 \mid 244_8$
Pilcrow	See Paragraph
Pisces	$357_8 \mid 361_8$
Planck's constant	$357_8 \mid 150_8$
Plus sign	$0_8 \mid 53_8$
Plus or minus sign	$0_8 \mid 261_8$
Product	$357_8 \mid 173_8$
Pound-Sterling sign	$0_8 \mid 243_8$
Pluto	$357_8 \mid 357_8$
QED	$357_8 \mid 270_8$
Quad, Em	$357_8 \mid 55_8$
Quad, En	$357_8 \mid 54_8$
Questioned equality	$357_8 \mid 164_8$
Question mark	$0_8 \mid 77_8$
Question mark, Inverted (Spanish)	$0_8 \mid 277_8$
Quote, Double, Neutral	$0_8 \mid 42_8$
Quote, Double, Left, European	$0_8 \mid 253_8$
Quote, Double, Left, Guillemet	See Quote, Double, Left, European
Quote, Double, Left, Lowered	$357_8 \mid 50_8$
Quote, Double, Right, European	$0_8 \mid 273_8$
Quote, Double, Right, Guillemet	See Quote, Double, Right, European
Quote, Double, Right, German	$357_8 \mid 51_8$
Quote, Single, Left	$0_8 \mid 251_8$
Quote, Single, Left, Guillemet	$357_8 \mid 52_8$
Quote, Single, Left, Lowered	$43_8 \mid 262_8$

Quote, Single, Neutral	$357_8 \mid 47_8$
Quote, Single, Right	$0_8 \mid 271_8$
Quote, Single, Right, Guillemet	$357_8 \mid 53_8$
Radical	$357_8 \mid 174_8$
Real number	See Number, Real
Registered sign	$0_8 \mid 322_8$
Reluctance	See Number, Real
Reversible reaction, Type 1	$357_8 \mid 121_8$
Reversible reaction, Type 2	$357_8 \mid 120_8$
Ring, Centered	$357_8 \mid 147_8$
Roman	See Alphabet, Roman
Root	See Radical
s, Double	$0_8 \mid 373_8$
Sagittarius	$357_8 \mid 372_8$
Saturn	$357_8 \mid 354_8$
Scorpius	$357_8 \mid 371_8$
Seconds sign	$41_8 \mid 155_8$
Section sign	$0_8 \mid 247_8$
Semicolon	$0_8 \mid 73_8$
Seven	$0_8 \mid 67_8$
Shade	$357_8 \mid 176_8$
Sharp, Musical	$43_8 \mid 274_8$
Sharp s	See s, Double
Similar to	$356_8 \mid 176_8$
Six	$0_8 \mid 66_8$
Skull & crossbones	$357_8 \mid 340_8$
Slant	$0_8 \mid 57_8$
Slant, Double	$357_8 \mid 110_8$
Slant, Reverse	$0_8 \mid 134_8$
Slant, Reverse, High	$0_8 \mid 140_8$
Slash, Double	See Slant, Double
Smile face	$357_8 \mid 337_8$
Solidus	See Slant
Space (Normally nonprinting)	$0_8 \mid 40_8$
Space, Figure (Normally nonprinting)	$357_8 \mid 56_8$
Space, Hair (Normally nonprinting)	$356_8 \mid 43_8$
Space, Non-breaking (Normally nonprinting)	$357_8 \mid 41_8$
Space, Numeric (Normally nonprinting)	See Space, Figure
Space, Punctuation (Normally nonprinting)	$356_8 \mid 44_8$
Space, Thick	(Normally nonprinting) $356_8 \mid 41_8$
Space, Thin (Normally nonprinting)	$357_8 \mid 57_8$
Spades	$357_8 \mid 313_8$
Square, Black	$42_8 \mid 43_8$
Square, White	See Ballot box
Star, Black	$41_8 \mid 172_8$
Star, White	$41_8 \mid 171_8$
Stop, Full (Period)	$0_8 \mid 56_8$
Stop, Full, Raised (Greek)	$46_8 \mid 73_8$
Stress mark, Phonetic (Non-spacing character)	See Acute

Stress mark, Phonetic (Spacing character)	See Minutes sign
Such that, Type 1	$357_8 \mid 114_8$
Such that, Type 2	$356_8 \mid 174_8$
Summation	$357_8 \mid 172_8$
Sun	$357_8 \mid 347_8$
Superscript 2	$0_8 \mid 262_8$
Superscript 3	$0_8 \mid 263_8$
t with stroke, Lowercase (Lapp)	$0_8 \mid 375_8$
T with stroke, Uppercase (Lapp)	$0_8 \mid 355_8$
Take	$357_8 \mid 251_8$
Taurus	$357_8 \mid 363_8$
TEL (Telephone)	$357_8 \mid 252_8$
Telephone symbol	$357_8 \mid 374_8$
There exists	$357_8 \mid 264_8$
Therefore	$41_8 \mid 150_8$
Thin space	See Space, Thin
Thorn, Lowercase (Icelandic)	$0_8 \mid 374_8$
Thorn, Uppercase (Icelandic)	$0_8 \mid 354_8$
Three-dot leader	$41_8 \mid 104_8$
Three, Superscript	See Superscript 3
Tilde (spacing)	$0_8 \mid 176_8$
Tilde (accent)	$0_8 \mid 304_8$
Trademark sign	$0_8 \mid 324_8$
Triangle, Black, Point-down	$42_8 \mid 47_8$
Triangle, Black, Point-up	$42_8 \mid 45_8$
Triangle, White, Point-down	$42_8 \mid 46_8$
Triangle, White, Point-up	$42_8 \mid 44_8$
Two	$0_8 \mid 62_8$
Two-dot leader	$41_8 \mid 105_8$
Two, Superscript	See Superscript 2
Three, Superscript	See Superscript 3
Umlaut (accent)	See Diaeresis
Underline (Non-spacing undermark)	$0_8 \mid 314_8$
Undermark, Cedilla	See Cedilla
Undermark, Ogonek	See Ogonek
Undermark, Underline	See Underline
Union	$357_8 \mid 127_8$
Uranus	$357_8 \mid 355_8$
Venus	See Female
Virgo	$357_8 \mid 367_8$
Virgule	See Slant
Vowel mark, Long	See Macron
Vowel mark, Short	See Breve
Wave operator	See Ballot box
X mark	$357_8 \mid 320_8$

Yen currency symbol (Japanese)	$0_8 \mid 245_8$
Yogh (Old English)	$357_8 \mid 253_8$
Zero	$0_8 \mid 60_8$

Appendix F
Glossary

An underlined word in a definition is defined in this glossary.

character: A graphic shape that is used for representation of visual information or a syntactic entity which lacks a physical representation. In electronic printing, a character is represented in the form of a spatial arrangement of adjacent pixels.

character appearance: A function of two variables: character codes which express the identity of characters and character "looks" information which models the different appearances that a shape can assume.

character code: Any code representing a graphic character, rendering character, or control character, usually unique within the set, most commonly represented as a cardinal number. In a particular code collection, i.e., the Xerox Character Code Standard, a character code is represented as a 16-bit non-negative integer.

character code collection: A listing of specific cardinal numbers from a character code standard.

character collection: A particular selection of the individual characters appearing in a font. The instance of a character collection having a fixed number of characters is known as a character set in the graphic arts—a character collection has no such limit.

character grouping: One or more NAMED character collections in union with a core character collection (also NAMED, but a member of a nested set of character collections).

character set: An instance of a character collection having a fixed number of characters. For purposes of this document, a block of 256 contiguous numerical codes, of which 188 may be assigned to graphical characters.

control character: A character, other than a graphic character or rendering character, whose occurrence in a particular context initiates, modifies, or stops a control operation. A control character, while not a graphic character or rendering character, may have a non-conventional representation in some circumstances.

core character collection: One of a nested family of NAMED character collections. In the graphic arts industry, an instance of a core character collection is called a universal character set.

Core Font Library: A font library having the essential typefaces and character collections for a particular targeted market available on all products.

display font: A particular variation of a type face, or unique design, where type is set larger than that used in text to attract a readers attention and where distinctiveness of design may take precedence over readability factors.

file: A set of related records treated as a unit.

file format: The arrangement and structure of data or words in a file, including the order and size of the components of the file.

font: A particular collection of characters of a typeface with unique parameters in the Variation vector, i.e., a particular instance of values for orientation, size, posture, weight, etc., attributes.
(general definition): The word font or fount is derived from the word "foundry" which is where, originally, type was cast. It has come to mean the vehicle which holds the typeface character collection. A font can be metal, photographic film or electronic media (cartridge, tape, disc).

font family: A particular collection of font progressions of a typeface where the Variation vector with given orientations and sizes can differ in posture attribute value and weight attribute values. By current practice a font family can include font progressions in posture attribute values of roman and italic and weight attribute values of light, medium, and bold, i.e., 40 X 2 X 3 = 240 fonts.

font file: A set of font records including a digital representation of a set or collection of graphic symbols and/or characters and control information for some level(s) of processing.

Font Library: A repository for fonts and font metric information on any number of typefaces, e.g., the Times Roman Family, the Helvetica Family, etc.

font progression: A particular size sequence of font rotations of a typeface, i.e., the Variation vector with a given orientation value can differ only in size attribute value. A typical progression is ten sizes for the typical four orientations, resulting in 40 fonts for a typical font progression.
(general definition): Sequential sizes of one typeface. The number of fonts which comprise a progression can be influenced by needs of a typical market.

font rotation: A set of fonts of a typeface where only the parameter of the Variation vector affecting the rotational orientation of character placement differs between fonts. Typical orientation values are portrait, landscape, inverse portrait, and inverse landscape.
(general definition): A set of fonts of one typeface which print vertically or horizontally on a page or their inverse.

general "*looks*": The finite set of basic character "looks" (see Appendix C, C.1) upon which agreement has been reached and which can be broadly applied to all characters in a designated code space.

graphic: A symbol produced by a process such as handwriting, drawing, or printing.

graphic character: A character, other than a control character or rendering character, that is normally represented by a graphic.

graphic shape: The physical form of letters, accent marks, numeric figures, fractions, symbols, and constructions (forms characters, mosaics, etc.).

kern: (noun): That portion of a letter which extends beyond its width, i.e., the letter shapes that overhang—the projection of a character beyond its side bearings.
(verb): The function of adjusting the intercharacter spacing in character groups (words) to improve their appearance.

library font file format: The arrangement and structure of Font Library information in a Font Library font file. The internal format within the library and not a printer or display device specific format.

logotype (or logo): A symbol, image, or complex character, composed of multiple entities which, when assembled as a unit, provide for a given graphic shape, i.e., the name of a company or product in a special design used as an identity mark.

NAMED character collection: The identity given by a registration service, and recorded within a Name Registry, to a particular selection of the characters appearing in deliverable from a Font Library.

Name Registry: The official record book within the registry for recording unique identifiers. Within a Font Library, the record book to record typeface and character code identifiers for fonts, e.g., Font Name Registry.

non-general "*looks*":
First kind: The finite set of character "looks" upon which agreement has been reached and which cannot be broadly applied to all characters in the designated code space.
Second kind: The thousands of non-interesting sets of character "looks" which may or may not be broadly applied to all characters in the designated code space.
Non-general "looks" of the first kind, those upon which agreement has been reached includes Old Style numerals, long descenders, small caps, etc.

registry: A dynamic (and sometimes complex) service whose purpose is to record information, assign and register unique identifiers, and respond to requests for information about identifiers.

rendering: A copy or version of a symbol produced by a process such as printing.

rendering character: A character, other than a control character or graphic character, that fits in one of four classes:
1st - a non-conventional representation of a control character.
2nd - a sequence of graphic characters (ligature or accented character).
3rd - contextually-dependent alternate representation for a graphic character (initial, medial, or final).
4th - a "variant" representation for a graphic character.

screen display font: A particular rendition of a typeface used exclusively to create character images on cathode ray tube screens, usually at low to very low resolution.

typeface: The features by which a character's design is recognized, hence the word "face." Within the Latin language group of graphic shapes are the following forms: Uncial, Blackletter, Serif, Sans Serif, Scripts, and Decorative. Each form characterizes one or more designs.
(Example: Serif form contains four designs called Old Style, Transitional, Modern, and Slab Serif designs. The typeface called Bodoni is a Modern design, while Times Roman is a Transitional design.)

variation: The extent to which a typographic object varies.

Variation vector: An ordered set of attributes and attribute values which models general character "looks" in the character "looks" informationspecification. An ordered set includes the following attributes: orientation, size, posture, weight, setwidth, stroke, measure, kerning, reading (right or wrong), and escapement (right to left playing and left to right playing).

XEROX

Xerox Corporation
Stamford, Connecticut 06904

XEROX® is a trademark of
XEROX CORPORATION.

Printed in U.S.A.

610P72583