APPENDIX F
Vendor Encoding Methods

This appendix is not included in the printed version of this book, and is instead available as a downloadable and printable PDF file.[*] As new material becomes available, the PDF file will be updated accordingly.

The material currently included is directly excerpted from the first edition, and an updated appendix for the second edition will be made available as that material becomes available.

* *http://examples.oreilly.com/9780596514471/cjkvip2e-appF.pdf*

D

Vendor Encoding Methods

The material covered in this appendix supplements Chapter 4, *Encoding Methods*, and Appendix C, *Vendor Character Set Standards*. Like Appendix C, it is intended as reference material in the event that you need to work with a particular vendor character set. The material here should provide enough information.

Most of the vendor encoding methods share similar encodings with the national character set encodings. This is appropriate since, as you have already learned, most vendor character set standards share many of the same characters with only slight variations. Table D-1 lists these vendor character sets, along with the encodings that support them. Some of the character sets described in Appendix C are not described in this appendix simply because they share the same encoding as described in Chapter 4 (or I couldn't find any encoding information for them).

Table D-1: CJKV Vendor Encoding Method Overview

Encoding	Supported Character Sets
ISO-2022	NEC Kanji, NTT Kanji
EUC	MacOS-S, DEC Korean, IKIS, NEC Kanji
EUC extension	DEC Hanyu, DEC Hanzi, DEC Kanji, HP-16 (Japanese), HangulTalk, Super DEC Kanji, Unified Hangul Code
Big Five	ETen, MacOS-T, DynaLab Hong Kong, Monotype Hong Kong
Shift-JIS	Enfour Gaiji, FMR Kanji, MacOS-J, Microsoft Japanese, NEC Kanji
Shift-JIS extension	HP-15 (Japanese)
IBM DBCS-PC	All IBM
IBM DBCS-Host	All IBM
IBM DBCS-EUC	All IBM
IBM TBCS-EUC	IBM Traditional Chinese
Other	JEF, KEIS78, KEIS83, TRON

At appropriate times in this appendix, comparisons are drawn between vendor encoding methods and those covered in Chapter 4.

Brief Overview of IBM Encodings

IBM has defined four basic multiple-byte language-independent encoding methods: DBCS-PC, DBCS-EUC, TBCS-EUC, and DBCS-Host (the abbreviation DBCS stands for Double-Byte Character Set; likewise, the abbreviation TBCS stands for Triple-Byte Character Set). TBCS-EUC is not used to encode Japanese. IBM manufactures and supports a wide variety of computing environments, such as host computers, PCs, and Unix workstations, and thus requires many encoding methods. While each locale has its own variation for each of these encodings, the tables in this section define these encodings in a generic way.

IBM DBCS-PC Encoding

Table D-2 provides the generic definition for IBM DBCS-PC encoding, which basically uses ASCII or equivalent in the one-byte range and a large and disjoint two-byte range.

Table D-2: IBM DBCS-PC Encoding Specifications—Generic

	Decimal	Hexadecimal
ASCII/CJKV-Roman		
Byte range	33—126	21—7E
Two-byte characters		
First byte range	129—254	81—FE
Second byte ranges	64—126, 128—254	40—7E, 80—FE

Note how the second-byte range is identical to the second-byte range of Shift-JIS encoding.

IBM DBCS-EUC and TBCS-EUC Encodings

IBM has developed EUC encodings that have been implemented identically to EUC encoding as described in Chapter 4. Note that there are two types of IBM EUC encodings, specifically DBCS-EUC and TBCS-EUC. TBCS-EUC refers to characters encoded using three bytes. In IBM terminology, the use of SS2 and SS3 does not count when totalling the number of bytes per character (they are treated as shift characters). So, TBCS-EUC is used only to refer to EUC-TW code set 2 (CNS 11643-1992 Planes 1 through 16).

IBM DBCS-Host Encoding

Table D-3 provides the generic definition for IBM DBCS-Host encoding, which is EBCDIC-based.

Table D-3: IBM DBCS-Host Encoding Specifications—Generic

	Decimal	Hexadecimal
One-byte characters		
Byte range	65–249	41–F9
Full-width space		
First byte	64	40
Second byte	64	40
Two-byte characters		
First byte range	65–254	41–FE
Second byte range	65–254	41–FE
Shifting characters		
One-byte character	15	0F
Two-byte character	14	0E

Note the use of shifting characters, which clearly shows that this encoding is modal. Also, the one-byte range is EBCDIC-based, not ASCII. All multiple-byte encodings that use EBCDIC for the one-byte range turn out to be modal.

Chinese Vendor Encodings—China

While most Chinese vendor character sets follow the GB 2312-80 character set standard and its most widely used encoding (EUC-CN), IBM has developed several other encodings that encapsulate the same character set, along with some IBM extensions.

DEC Hanzi Encoding

DEC Hanzi encoding is identical to EUC-CN encoding except that there is an additional 94×94 region for encoding user-defined characters. Table D-4 illustrates DEC Hanzi encoding.

Table D-4: DEC Hanzi Encoding Specifications

	Decimal	Hexadecimal
ASCII/GB-Roman		
Byte range	33–126	21–7E

Table D-4: DEC Hanzi Encoding Specifications (continued)

	Decimal	Hexadecimal
GB 2312-80		
First byte range	161—254	A1—FE
Second byte range	161—254	A1—FE
User-defined characters		
First byte range	161—254	A1—FE
Second byte range	33—126	21—7E

IBM Simplified Chinese Encodings

IBM has developed a number of encodings for handling Simplified Chinese, specifically for those character sets that are based on GB 2312-80. The following sections describe the Simplified Chinese encodings originally developed by IBM, specifically DBCS-PC and DBCS-Host encodings. IBM has recently adopted GBK (as DBCS-PC), and has their own version of EUC-CN encoding (called DBCS-EUC). Please refer to IBM documentation for details about their EUC-CN and GBK implementations.

IBM Simplified Chinese DBCS-PC encoding

Table D-5 provides the encoding specifications for IBM Simplified Chinese DBCS-PC encoding.

Table D-5: IBM Simplified Chinese DBCS-PC Encoding Specifications

	Decimal	Hexadecimal
ASCII/GB-Roman		
Byte range	33—126	21—7E
GB 2312-80[a]		
First byte range	129—172	81—AC
Second byte ranges	64—126, 128—252	40—7E, 80—FC
User-defined characters		
First byte range	240—249	F0—F9
Second byte ranges	64—126, 128—252	40—7E, 80—FC
IBM Selected Characters		
First byte range	250—251	FA—FB
Second byte ranges	64—126, 128—252	40—7E, 80—FC
Reserved		
First byte ranges	176—239, 252	B0—EF, FC
Second byte ranges	64—126, 128—252	40—7E, 80—FC

[a] The last defined character in this region is 0xAC9E.

IBM Simplified Chinese DBCS-Host encoding

Table D-6 provides the encoding specifications for IBM Simplified Chinese DBCS-Host encoding.

Table D-6: IBM Simplified Chinese DBCS-Host Encoding Specifications

	Decimal	Hexadecimal
One-byte characters		
Byte range	65—249	41—F9
Full-width space		
First byte	64	40
Second byte	64	40
Two-byte characters[a]		
First byte range	65—111	41—6F
Second byte ranges	65—127, 129—253	41—7F, 81—FD
User-defined characters		
First byte range	118—127	76—7F
Second byte ranges	65—127, 129—253	41—7F, 81—FD
Shifting characters		
One-byte character	15	0F
Two-byte character	14	0E
Reserved		
First byte ranges	112—117, 128—254	70—75, 80—FE
Second byte ranges	65—127, 129—253	41—7F, 81—FD

[a] The last defined character in this region is 0x6C9F.

MacOS-S Encoding

The encoding used on MacOS-S is identical to EUC-CN except that it accommodates additional one-byte characters (see Table C-6 on page 558). In fact, the existence of two of these additional one-byte characters, at code points 0xFD and 0xFE, effectively reduces the size of the two-byte encoding region by 188 code points.

Table D-7 illustrates the MacOS-S encoding specifications, which show the reduced two-byte encoding region: 0xA1A1 through 0xFCFE.

Table D-7: MacOS-S Encoding Specifications

	Decimal	Hexadecimal
ASCII/GB-Roman		
Byte ranges	33—126, 128, 253—255	21—7E, 80, FD—FF

Table D-7: MacOS-S Encoding Specifications (continued)

	Decimal	Hexadecimal
GB 2312-80		
First byte range	161—252	A1—FC
Second byte range	161—254	A1—FE

Chinese Vendor Encodings—Taiwan

There are many vendor extensions to Big Five. This is because Big Five is much more widely implemented than CNS 11643-1992. See Table 4-39 on page 172 for a complete description of Big Five encoding. DEC Hanyu seems to be the only vendor implementation that is based on CNS 11643-1992.

DEC Hanyu Encoding

DEC Hanyu encoding consists of a mixed one-, two-, and four-byte encoding. It differs from EUC-TW encoding in two major ways:

- CNS 11643-1992 Planes 1 and 2 are encoded using two bytes
- CNS 11643-1992 Planes 3 and 4 are encoded using four bytes, but the first two bytes are always 0xC2 and 0xCB

Table D-8 provides the DEC Hanyu encoding specifications, illustrating the complete one- through four-byte encoding region.

Table D-8: DEC Hanyu Encoding Specifications

	Decimal	Hexadecimal
ASCII/CNS-Roman		
Byte ranges	33—126	21—7E
CNS 11643-1992 Plane 1		
First byte range	161—254	A1—FE
Second byte range	161—254	A1—FE
CNS 11643-1992 Plane 2		
First byte range	161—254	A1—FE
Second byte range	33—126	21—7E
CNS 11643-1992 Plane 3		
First byte	194	C2
Second byte	203	CB
Third byte range	161—254	A1—FE
Fourth byte range	161—254	A1—FE

Table D-8: DEC Hanyu Encoding Specifications (continued)

	Decimal	Hexadecimal
CNS 11643-1992 Plane 4		
First byte	194	C2
Second byte	203	CB
Third byte range	161—254	A1—FE
Fourth byte range	33—126	21—7E

ETen Encoding

The encoding used by the ETen character set is simply Big Five. The ETen-specific characters are encoded in rows 0xC6 through 0xC8 and 0xF9, which are within the Big Five encoding definition.

IBM Traditional Chinese Encodings

While the most widely used IBM Traditional Chinese encoding is IBM DBCS-Big5, there are a number of encodings for IBM Traditional Chinese, some of which are covered in this section. IBM Traditional Chinese DBCS-EUC and TBCS-EUC encodings support the CNS 11643-1992 character set, at least Planes 1 through 3, and is identical to EUC-TW encoding.

IBM Traditional Chinese DBCS-PC encoding

Table D-9 provides the encoding specifications for IBM Traditional Chinese DBCS-PC encoding.

Table D-9: IBM Traditional Chinese DBCS-PC Encoding Specifications

	Decimal	Hexadecimal
ASCII/CNS-Roman		
Byte range	33—126	21—7E
Two-byte characters[a]		
First byte range	129—209	81—D1
Second byte ranges	64—126, 128—252	40—7E, 80—FC
User-defined characters		
First byte range	219—251	DB—FB
Second byte ranges	64—126, 128—252	40—7E, 80—FC
Reserved		
First byte ranges	210—218, 252	D2—DA, FC
Second byte ranges	64—126, 128—252	40—7E, 80—FC

[a] The last defined character in this region is 0xD1C6.

IBM Traditional Chinese DBCS-Host encoding

Table D-10 provides the encoding specifications for IBM Traditional Chinese DBCS-Host encoding.

Table D-10: IBM Traditional Chinese DBCS-Host Encoding Specifications

	Decimal	Hexadecimal
One-byte characters		
Byte range	65–249	41–F9
Full-width space		
First byte	64	40
Second byte	64	40
Two-byte characters[a]		
First byte range	65–145	41–91
Second byte ranges	65–127, 129–253	41–7F, 81–FD
User-defined characters		
First byte range	194–226	C2–E2
Second byte ranges	65–127, 129–253	41–7F, 81–FD
Shifting characters		
One-byte character	15	0F
Two-byte character	14	0E
Reserved		
First byte ranges	74–75, 146–193, 227–254	4A–4B, 92–C1, E3–FE
Second byte ranges	65–127, 129–253	41–7F, 81–FD

[a] The last defined character in this region is 0x91C7.

MacOS-T Encoding

The encoding used by the MacOS-T character set is slightly reduced Big Five encoding (to accommodate two of its four additional one-byte characters, as illustrated in Table C-14 on page 563). Table D-11 provides a complete description of the Big Five encoding used on MacOS-T.

Table D-11: MacOS-T Encoding Specifications

	Decimal	Hexadecimal
ASCII/CNS-Roman		
Byte ranges	33–126, 128, 253–255	21–7E, 80, FD–FF
Big Five		
First byte range	161–252	A1–FC
Second byte ranges	64–126, 161–254	40–7E, A1–FE

Microsoft Traditional Chinese Encoding

The encoding used by the Microsoft Traditional Chinese character set is simply Big Five. Characters from the ETen character set are encoded in row 0xF9, which is within the Big Five encoding definition (the ETen characters encoded in rows 0xC6 through 0xC8 are not included).

Chinese Vendor Encodings—Hong Kong

All of the encodings that support Hong Kong extensions are based on Big Five encoding. In fact, Big Five is still considered the *de facto* character set and encoding for the Hong Kong locale. While the Hong Kong government has developed its own Hong Kong extension to Big Five (which itself required an extension to Big Five encoding), those Hong Kong extensions developed by vendors keep its characters encoded within the standard Big Five definition. This is important for compatibility with existing operating systems that are based on Big Five.

DynaLab Hong Kong Encoding

The encoding used by DynaLab's Hong Kong extension is simply Big Five. However, when implemented on MacOS-T, 0xFD and 0xFE are unavailable as the first byte of a two-byte character because they are reserved for single-byte characters. Table D-11 on page 612 showed how the MacOS-T implementation of Big Five encoding treats rows 0xFD and 0xFE as part of the single-byte range, not as the first byte of a two-byte character. Under these circumstances, the DynaLab Hong Kong hanzi encoded in rows 0xFD and 0xFE are simply not available.

Monotype Hong Kong Encoding

The encoding used by the Monotype Hong Kong character set is simply Big Five. The Hong Kong characters are encoded in rows 0xFA through 0xFC. This makes it possible to implement this character set on MacOS-T because row 0xFC is still considered to be part of the two-byte encoding region. Table D-11 on page 612 showed how the MacOS-T implementation of Big Five encoding treats rows 0xFD and 0xFE as part of the single-byte range, not as the first byte of a two-byte character.

Japanese Vendor Encodings

All of the major Japanese encodings—ISO-2022-JP, EUC-JP, and Shift-JIS—have been adopted by at least one Japanese vendor for use in their products. There

have also been other encodings developed, some of them pre-dating ISO-2022-JP, EUC-JP, and Shift-JIS encodings.

DEC Kanji Encoding

DEC Kanji encoding is very similar to EUC-JP complete two-byte format. The equivalent of EUC-JP code sets 0, 1, and 3 are supported. Note that the ASCII/JIS-Roman portion of DEC Kanji encoding is identical to EUC-JP packed format (that is, one-byte). Also note that the equivalent of EUC-JP code set 2, specifically half-width katakana, is not supported. Table D-12 shows the encoding.

Table D-12: DEC Kanji Encoding Specifications

	Decimal	Hexadecimal
ASCII/JIS-Roman		
Byte range	33–126	21–7E
JIS X 0208:1997		
First byte range	161–254	A1–FE
Second byte range	161–254	A1–FE
User-defined characters		
First byte range	161–254	A1–FE
Second byte range	33–126	21–7E

Super DEC Kanji encoding expands upon the encoding regions provided by DEC Kanji encoding. Put simply, Super DEC Kanji encoding is DEC Kanji encoding with the rest of EUC-JP encoding thrown in. Table D-13 provides the complete encoding specifications for Super DEC Kanji encoding.

Table D-13: Super DEC Kanji Encoding Specifications

	Decimal	Hexadecimal
ASCII/JIS-Roman		
Byte range	33–126	21–7E
JIS X 0208:1997		
First byte range	161–254	A1–FE
Second byte range	161–254	A1–FE
Half-width katakana		
First byte	142	8E
Second byte range	161–223	A1–DF
JIS X 0212-1990		
First byte	143	8F
Second byte range	161–254	A1–FE
Third byte range	161–254	A1–FE

Table D-13: Super DEC Kanji Encoding Specifications (continued)

	Decimal	Hexadecimal
User-defined characters		
First byte range	161—254	A1—FE
Second byte range	33—126	21—7E

DEC Kanji, Super DEC Kanji, and EUC-JP encodings

DEC Kanji encoding is identical to EUC-JP complete two-byte format without code set 2 (half-width katakana). Also note that the equivalent of EUC-JP code set 3 is not specified to be used for the JIS X 0212-1990 character set.

Super DEC Kanji encoding, however, is a superset of DEC Kanji and EUC-JP encodings.

Fujitsu Japanese Encodings

Fujitsu has developed both mainframe and personal computers, and the character sets and encodings used on each are different. The JEF character set and encoding are used on their FACOM mainframe computers, and the FMR Kanji character set and encoding are used on their personal computers.

JEF encoding

JEF encoding is quite unusual. First, it does not use the ASCII/JIS-Roman character set or encoding—it uses EBCDIC/EBCDIK instead. This allows for a quite different encoding structure, yet you will see some similarities with EUC-JP encoding. The first byte's value spans the seven- and eight-bit range. This does not allow such an encoding to make use of the byte's value to determine whether it represents itself, or is part of an expected two-byte sequence. JEF, instead, uses special characters for performing such shifts of state. The code specifications for JEF are listed in Table D-14.

Table D-14: JEF Encoding Specifications

	Decimal	Hexadecimal
One-byte characters		
Byte range	65—249	41—F9
Full-width space		
First byte	64	40
Second byte	64	40
JIS C 6226-1978[a]		
First byte range	161—254	A1—FE
Second byte range	161—254	A1—FE

Table D-14: JEF Encoding Specifications (continued)

	Decimal	Hexadecimal
JEF Extended characters		
First byte ranges	65—125, 127	41—7D, 7F
Second byte range	161—254	A1—FE
User-defined characters		
First byte range	128—160	80—A0
Second byte range	161—254	A1—FE
Shifting characters		
One-byte character	41	29
Two-byte character	40	28

[a] 0xA1A1 is not a valid code position.

First, the encoding for the JIS C 6226-1978 character set is identical to EUC-JP code set 1. The remainder is unique to JEF, except for the special encoding for the full-width space, which is shared by IBM Japanese DBCS-Host encoding (see page 623).

Figure D-1 illustrates the JEF encoding space. Note that it is similar to EUC-JP encoding (the JIS C 6226-1978 portion shares the same encoding with EUC-JP code set 1), and that the full-width space character is off by itself within the encoding space.

JEF and EUC-JP encodings

The JIS C 6226-1978 portion of JEF encoding is identical to that of EUC-JP code set 1. The rest of JEF encoding, as you saw, is quite different.

FMR Kanji encoding

The FMR Kanji encoding is identical to Shift-JIS encoding. There is not much more to say about it here. See Figure 4-16 on page 178 for an illustration of Shift-JIS encoding, and Table 4-42 on page 176 for a listing of its specifications. Also see Table 4-43 on page 178 for a listing of the user-defined character area. The first byte range for these characters is from 0xF0 to 0xFC.

HP Kanji Encodings

Hewlett-Packard developed two Japanese encoding methods called HP-15 and HP-16. Both encode the same set of characters, but in a different way. HP-15 is a superset of Shift-JIS encoding, and HP-16 is similar to EUC-JP encoding. These encoding methods are fully compatible with one another, and Hewlett-Packard

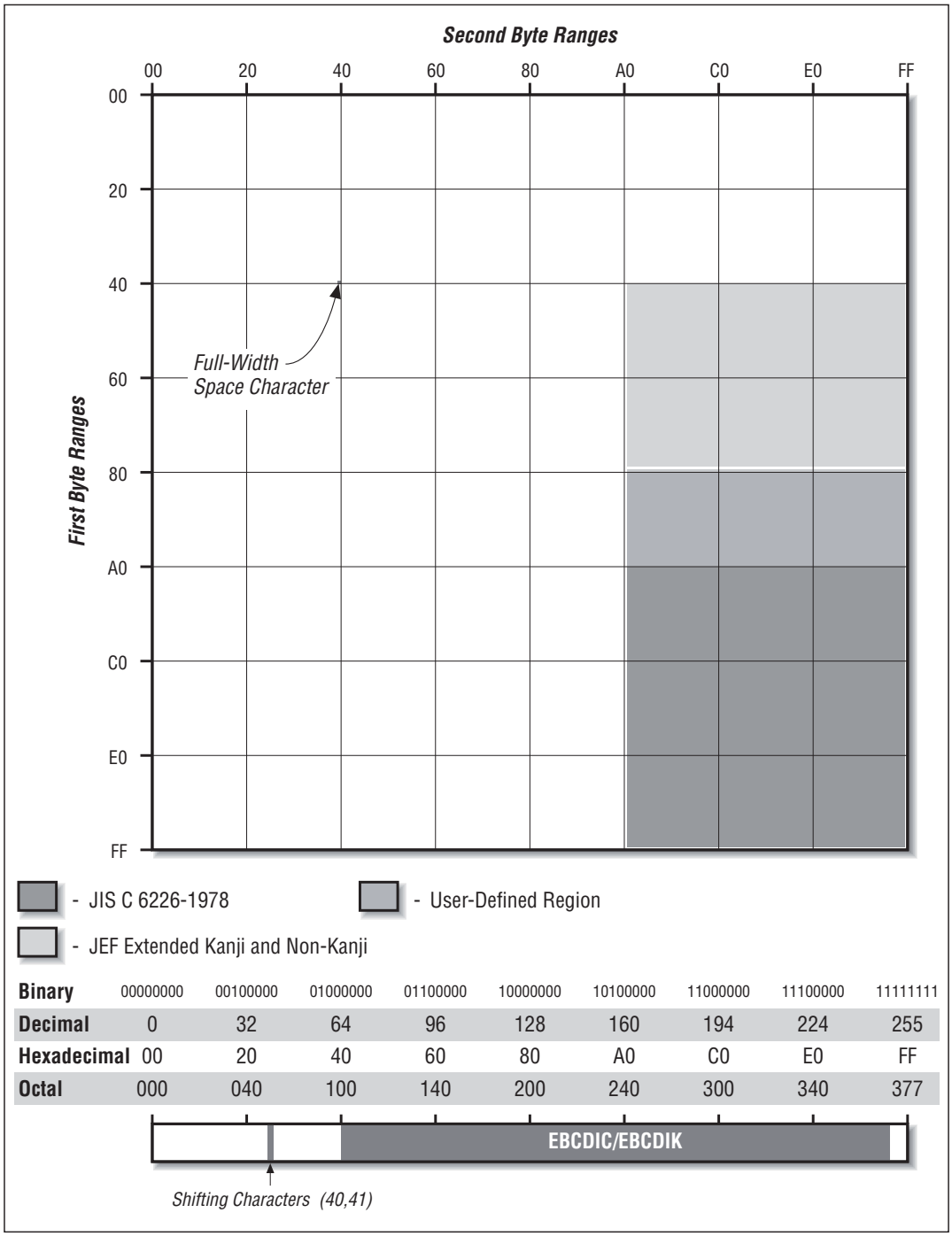


Figure D-1: JEF encoding tables

software is able to convert between them. I have seen the mapping tables, and they are not in a form that can be readily digested.

HP-15 encoding

HP-15 is more or less the same as Shift-JIS—all the standard Shift-JIS code positions fall entirely into the HP-15 code space. Hewlett-Packard defines HP-15 as a series of four encoding blocks, but this can be simplified into two ranges for both bytes. Table D-15 lists the code specifications for HP-15.

Table D-15: HP-15 Encoding Specifications

	Decimal	Hexadecimal
ASCII/JIS-Roman		
Byte range	33—126	21—7E
Two-byte characters		
First byte ranges	128—160, 224—254	80—A0, E0—FE
Second byte ranges	33—126, 128—255	21—7E, 80—FF

These ranges include a user-defined character area that falls at the outskirts of the standard Shift-JIS encoding space. More about this below. Figure D-2 illustrates the encoding space for HP-15.

HP-16 encoding

HP-16—the other Japanese encoding developed by Hewlett-Packard—has an area in its code space that is identical to EUC-JP code set 1. The remaining code space is used for user-defined characters. Hewlett-Packard defines the HP-16 encoding with a series of four encoding blocks. Table D-16 lists the code specifications for HP-16.

HP Kanji versus Shift-JIS and EUC-JP encodings

We really should compare the HP Kanji encodings to Shift-JIS and EUC-JP encodings. HP-15, as mentioned before, is a superset of Shift-JIS encoding, and HP-16 is similar to EUC-JP encoding. Figure D-3 illustrates the encoding space for HP-16.

Figure D-4 illustrates how Shift-JIS encoding is a subset of HP-15—it contains a slightly larger encoding region.

EUC-JP code set 1 is the same as the main portion of HP-16. This is the only similarity between EUC-JP and HP-16. Compare Figures 4-11 on page 166 and D-3 to confirm this.

IBM Japanese Encodings

In this section you will learn about Japanese-specific implementations of IBM's encoding methods.

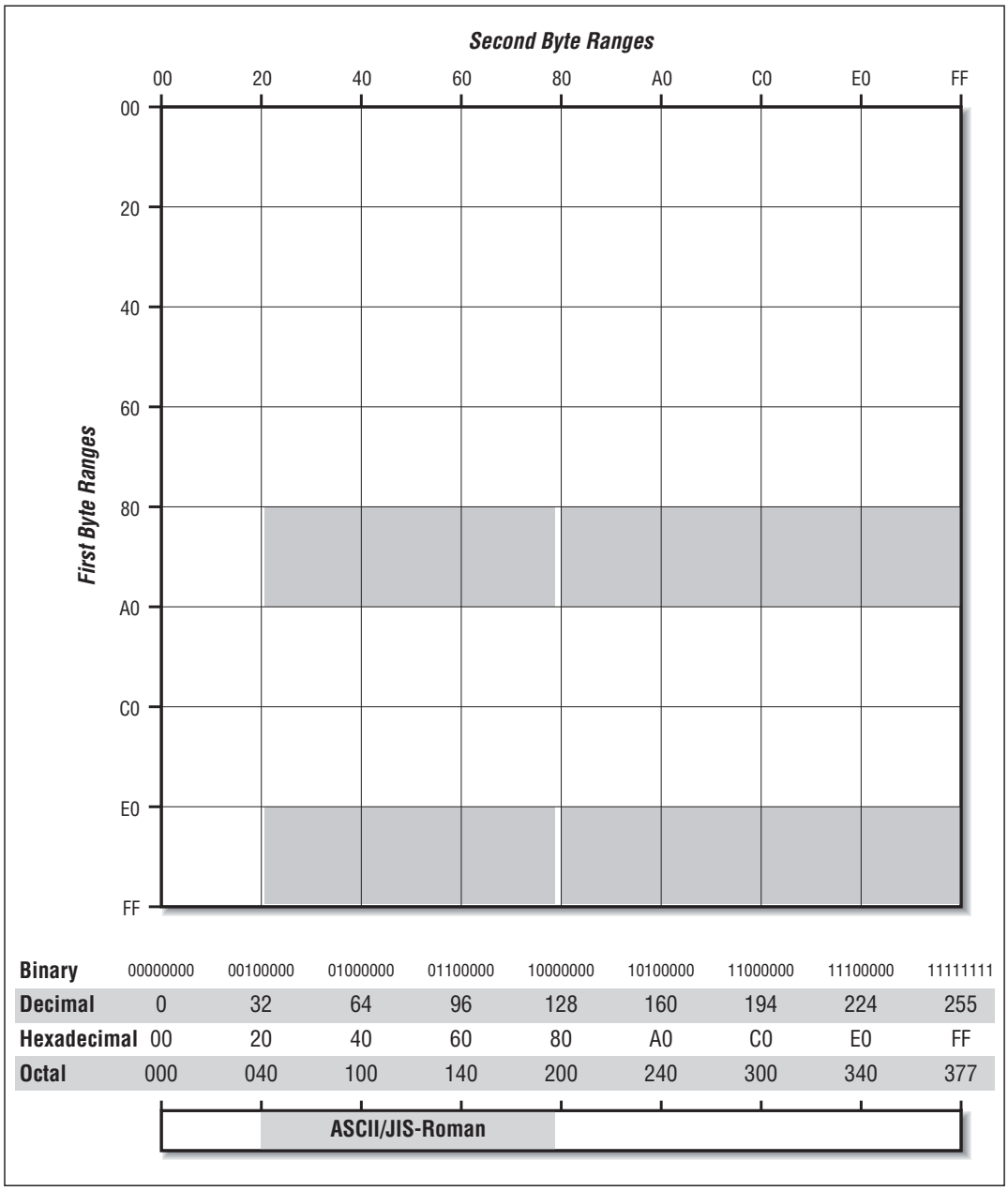


Figure D-2: HP-15 encoding tables

Table D-16: HP-16 Encoding Specifications

	Decimal	Hexadecimal
ASCII/JIS-Roman		
Byte range	33–126	21–7E

Table D-16: HP-16 Encoding Specifications (continued)

	Decimal	Hexadecimal
Two-byte characters		
First byte range	161—254	A1—FE
Second byte range	161—254	A1—FE
User-defined characters		
First byte range	161—194	A1—C2
Second byte range	33—126	21—7E
User-defined characters		
First byte ranges	195—227	C3—E3
Second byte ranges	33—63	21—3F
User-defined characters		
First byte ranges	195—225	C3—E1
Second byte ranges	64—100	40—64

IBM Japanese DBCS-PC encoding

IBM Japanese DBCS-PC encoding is nearly identical to Shift-JIS, contains a user-defined region, and is widely used on IBM PCs. The data found in Table D-17 is also known as IBM Code Page 932, which is a mixture of DBCS-PC, ASCII, and half-width katakana.

See Figure D-5 for an illustration of IBM Japanese DBCS-PC encoding, which illustrates how it is related to Shift-JIS encoding.

The generic definition for DBCS-PC is a bit different. What is listed in Table D-17 is the specific Japanese implementation of DBCS-PC. The main difference is that the first byte range falls between 0x81 and 0xFE. Japanese restricts this range to 0x81–0x9F and 0xE0–0xFC so that half-width katakana, whose encoding range falls in 0xA1–0xDF, could be accommodated. It is important to note that IBM Selected Kanji and Non-kanji fall into a code range that does not correspond to valid code positions in ISO-2022-JP or EUC-JP (code set 1)—they fall well outside the 94×94 matrix when run through the normal conversion algorithms. This becomes an issue when you learn about IBM's implementation of EUC encoding, the next topic of discussion.

IBM Japanese DBCS-EUC encoding

DBCS-EUC encoding is essentially identical to EUC, which was covered in detail in a previous section. All four code sets are implemented. This encoding is primarily found on the AIX environment, and the Japanese implementation is commonly known as IBM-eucJP. TBCS-EUC is defined by IBM for future standardization, and is currently not specified for handling Japanese.

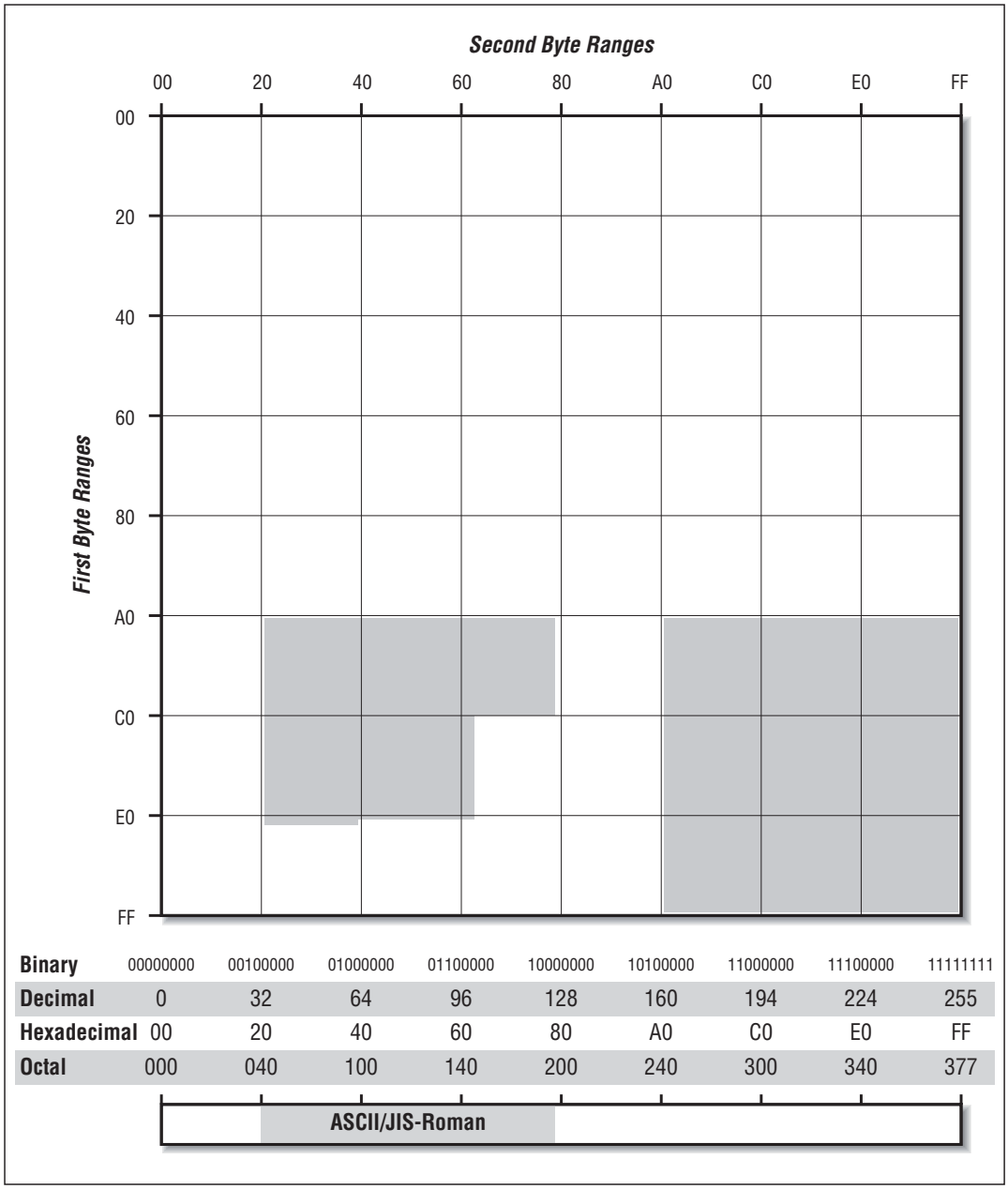


Figure D-3: HP-16 encoding tables

You may be asking how DBCS-EUC handles IBM Selected Kanji and Non-kanji. The answer is that these characters are mapped to JIS X 0208:1997 and JIS X 0212-1990 code positions. Table D-18 summarizes how these characters are mapped.

25 of the 28 IBM Selected Non-kanji are mapped to row 83 of JIS X 0212-1990, and 80 of the 360 IBM Selected Kanji are mapped to row 84 of JIS X 0212-1990. Appendix Q, *Character Lists and Mapping Tables*, shows how the 388 IBM

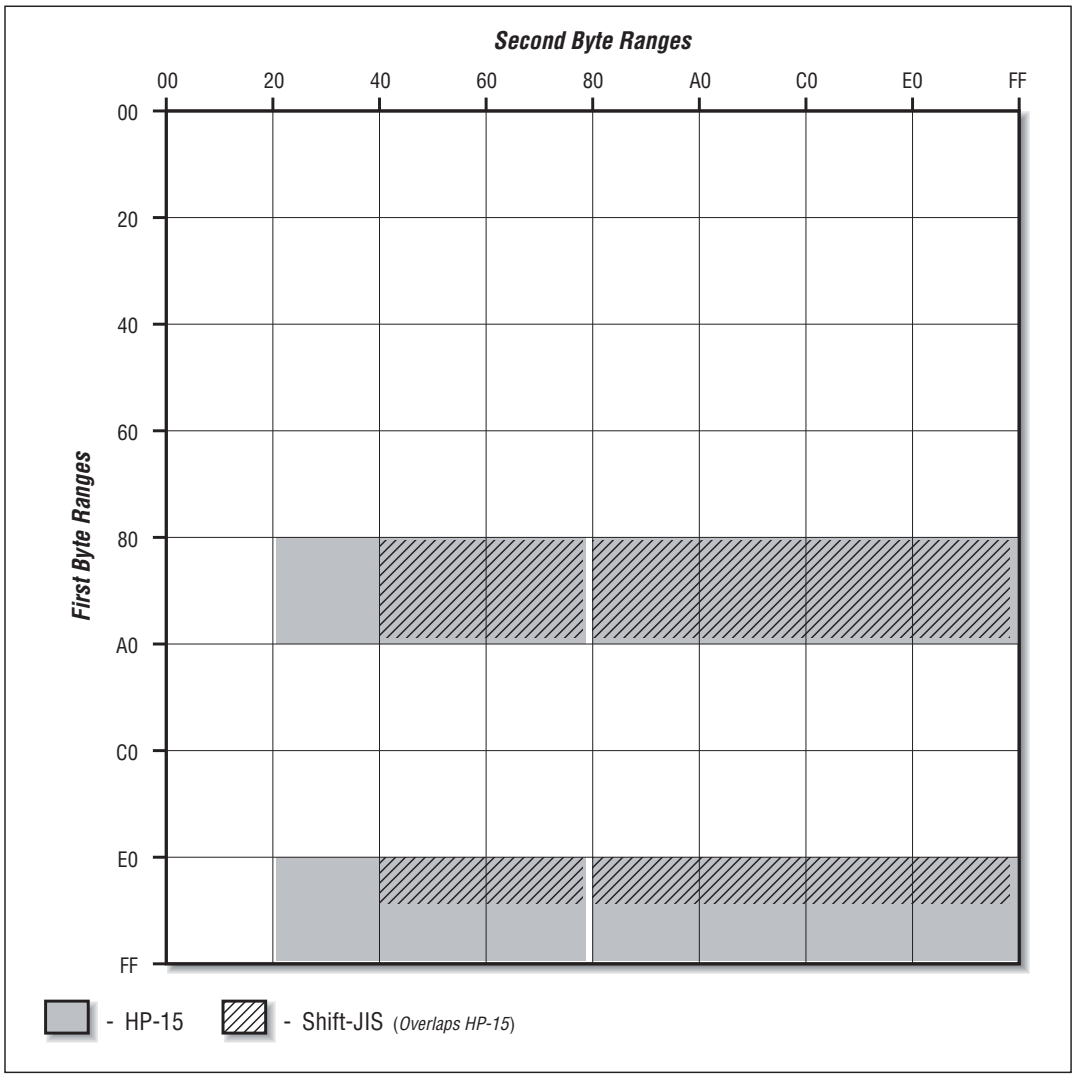


Figure D-4: HP-15 and Shift-JIS encodings—two-byte regions

Table D-17: IBM Japanese DBCS-PC Encoding Specifications

	Decimal	Hexadecimal
ASCII/JIS-Roman		
Byte range	33–126	21–7E
Half-width katakana[a]		
Byte range	161–223	A1–DF
JIS X 0208:1997[b]		
First byte ranges	129–132, 136–159, 224–234	81–84, 88–9F, E0–EA
Second byte ranges	64–126, 128–252	40–7E, 80–FC

Table D-17: IBM Japanese DBCS-PC Encoding Specifications (continued)

	Decimal	Hexadecimal
User-defined characters		
First byte range	240—249	F0—F9
Second byte ranges	64—126, 128—252	40—7E, 80—FC
IBM Selected Characters		
First byte range	250—252	FA—FC
Second byte ranges	64—126, 128—252	40—7E, 80—FC
Reserved[c]		
First byte ranges	133—135, 235—239	85—87, EB—EF
Second byte ranges	64—126, 128—252	40—7E, 80—FC

[a] Sometimes the code position 0xA0 is used for a half-width katakana space.
[b] The last defined character in this region is 0xEAA4—the same as Shift-JIS encoding.
[c] Note that these ranges correspond to code points within the JIS X 0208:1997 94×94 matrix.

Selected Kanji and Non-kanji are mapped to JIS X 0208:1997 and JIS X 0212-1990 code points. The results of this mapping process, at least for the 360 IBM Selected Kanji, concur with the findings of my own study that compared them with JIS X 0212-1990.

There are three user-defined character areas defined for IBM's Japanese DBCS-EUC implementation. They are listed in Table D-19.

IBM Japanese DBCS-Host encoding

DBCS-Host encoding, usually found on host computer systems, has a much larger character encoding space than DBCS-PC encoding. This two-byte encoding space can hold up to 36,481 unique characters. It is also used in conjunction with EBCDIC. This encoding method also existed long before any similar national standards existed. It is listed in Table D-20.

Figure D-6 illustrates the structure of IBM DBCS-Host encoding, which clearly shows how the full-width space is treated separately from other two-byte characters.

Converting Japanese text between IBM Japanese DBCS-Host and IBM Japanese DBCS-PC/EUC requires the use of mapping tables—no code conversion algorithm exists, which means that every character must be treated as a special case. These mapping tables exist in machine-readable form as part of IBM's *Character Data Representation Architecture Reference and Registry* (1995, IBM part number SC09-2190-00).

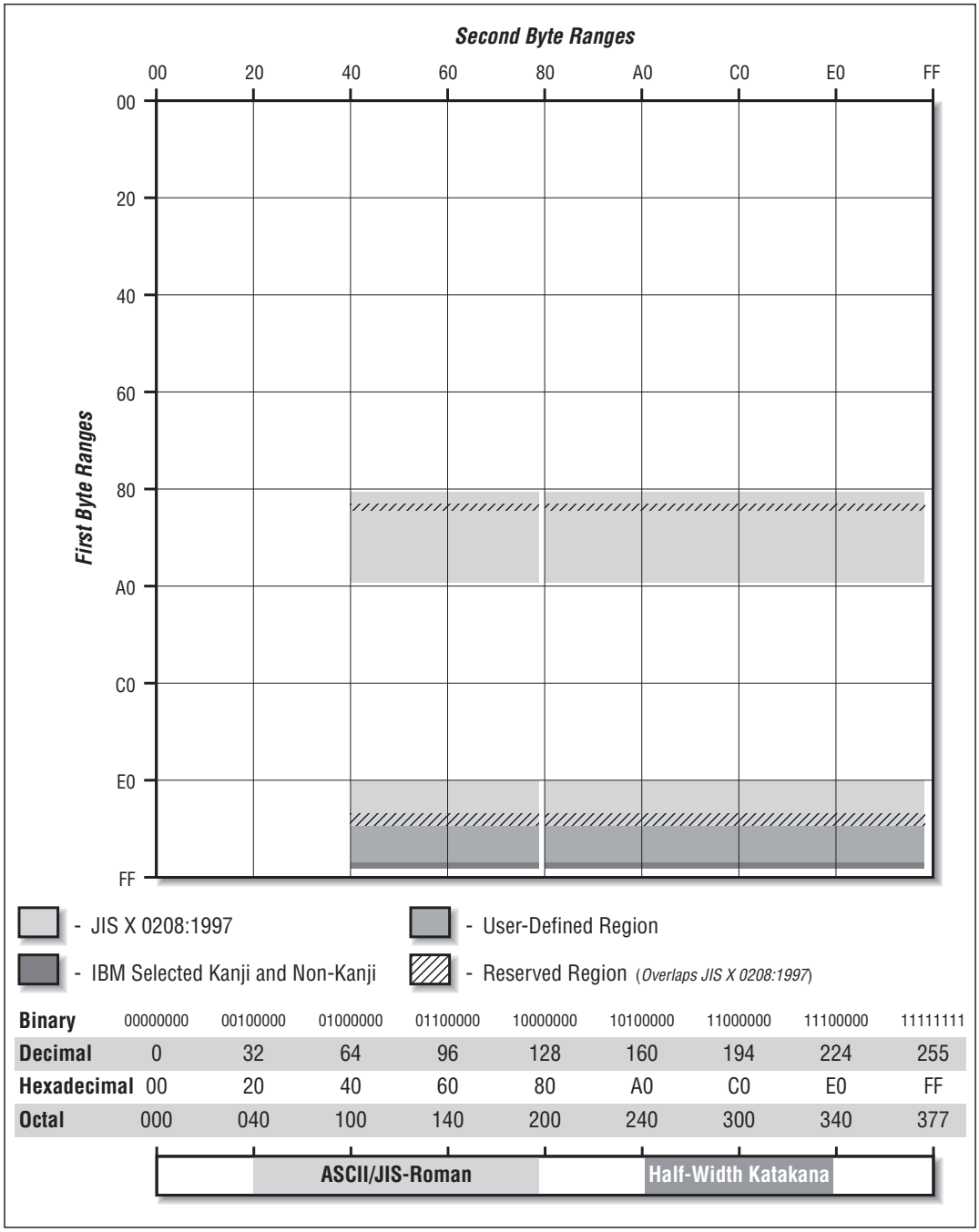


Figure D-5: IBM Japanese DBCS-PC encoding tables

IBM Japanese encodings versus Shift-JIS and EUC-JP encodings

IBM Japanese DBCS-Host encoding does not correspond to ISO-2022-JP, Shift-JIS, or EUC-JP encoding. IBM Japanese DBCS-PC corresponds to Shift-JIS encoding with the addition of a user-defined region (this is where the 386 IBM Selected

Table D-18: IBM Selected Kanji and Non-Kanji Mappings

	Total	JIS X 0208:1997	JIS X 0212-1990	User-Defined
IBM Selected Kanji	360	1	279	80
IBM Selected Non-kanji	28	2	1	25

Table D-19: IBM Japanese DBCS-EUC User-Defined Character Regions

Area	Location	Number of Code Points
Primary	Rows 85–94 of JIS X 0208:1997	940
Secondary	Rows 85–94 of JIS X 0212-1990	940
Tertiary	Rows 78–84 of JIS X 0212-1990	658

Table D-20: IBM Japanese DBCS-Host Encoding Specifications

	Decimal	Hexadecimal
One-byte characters		
Byte range	65—249	41—F9
Full-width space		
First byte	64	40
Second byte	64	40
Two-byte characters[a]		
First byte range	65—104	41—68
Second byte range	65—254	41—FE
User-defined characters[b]		
First byte range	105—114	69—72
Second byte range	65—254	41—FE
Shifting characters		
One-byte character	15	0F
Two-byte character	14	0E
Reserved		
First byte ranges	115—254	73—FE
Second byte range	65—254	41—FE

[a] The last defined character in this region is 0x6885.
[b] The last user-defined character in this region is 0x72EA.

Kanji and Non-kanji are encoded). IBM Japanese DBCS-EUC corresponds to EUC-JP encoding. You learned how the 386 IBM Selected Kanji and Non-kanji are mapped to appropriate positions in IBM Japanese DBCS-EUC encoding.

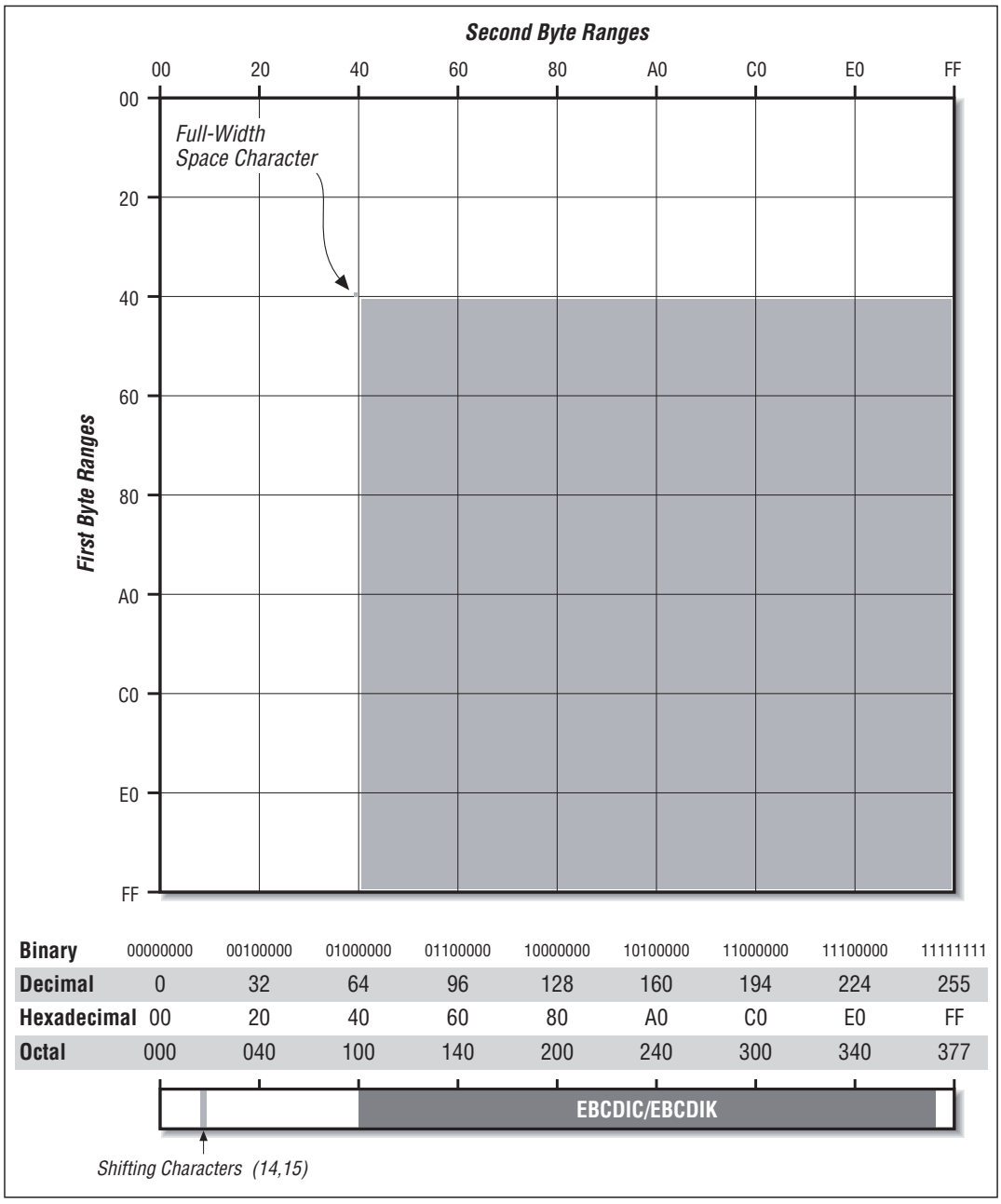


Figure D-6: IBM Japanese DBCS-Host encoding tables

IKIS Encoding

The encoding method for IKIS closely resembles EUC-JP, but lacks two of the code sets (code set 2 for half-width katakana, and code set 3 for JIS X 0212-1990). Remember from our discussion in Appendix C that the half-width katakana character set is included within the JIS X 0208-1983 character space, specifically in row

8 (where the line-drawing characters are normally found). Table D-21 illustrates IKIS' encoding specifications.

Table D-21: IKIS Encoding Specifications

	Decimal	Hexadecimal
ASCII/JIS-Roman		
Byte range	33—126	21—7E
JIS X 0208-1983		
First byte range	161—254	A1—FE
Second byte range	161—254	A1—FE

IKIS and EUC-JP encodings

IKIS encoding is identical to the encoding specified in EUC-JP complete two-byte format. Only the equivalent of EUC-JP code sets 0 and 1 is supported by IKIS.

KEIS Encoding

The encoding method used for KEIS includes encoding ranges found in parts of EUC-JP encoding, specifically EUC-JP code set 1. It also uses shifting sequences to shift between one- and two-byte-per-character modes. KEIS encoding is used in conjunction with EBCDIK. The valid two-byte encoding region for KEIS is actually quite large, but a large chunk of it is reserved and apparently unused. Table D-22 illustrates this large two-byte encoding region.

Table D-22: KEIS Two-Byte Encoding Region

	Decimal	Hexadecimal
Two-byte characters		
First byte range	65—254	41—FE
Second byte range	65—254	41—FE

Note that the KEIS two-byte encoding region is identical to generic instance of IBM DBCS-Host encoding.

You may recall from Appendix C that KEIS Basic Character Set and KEIS Extended Character Set 1 together constitute the JIS X 0208 character set.

Table D-23 provides the full specifications for KEIS encoding.

Table D-23: KEIS Encoding Specifications

	Decimal	Hexadecimal
One-byte characters		
Byte range	65—249	41—F9

Table D-23: KEIS Encoding Specifications (continued)

	Decimal	Hexadecimal
Full-width space[a]		
First byte	64 or 161	40 or A1
Second byte	64 or 161	40 or A1
KEIS Basic set		
First byte range	161–207	A1–CF
Second byte range	161–254	A1–FE
KEIS Extended Set 1		
First byte range	208–254	D0–FE
Second byte range	161–254	A1–FE
KEIS Extended Set 3		
First byte range	89–128	59–80
Second byte range	161–254	A1–FE
User-defined characters		
First byte range	129–160	81–A0
Second byte range	161–254	A1–FE
Shifting sequences		
One-byte character	10 65	A0 42
Two-byte character	10 66	A0 42

[a] A full-width space can be represented by either 0x4040 or 0xA1A1.

KEIS and EUC-JP encodings

The encoding for KEIS Basic Character Set and KEIS Extended Character Set 1 is identical to the encoding used for EUC-JP code set 1. The encoding for KEIS Extended Character Set 3 and KEIS user-defined characters departs from what we find in EUC-JP encoding, and does not correspond to ISO-2022-JP nor Shift-JIS encodings.

The big difference is that KEIS encoding is modal; it uses shifting sequences to switch between one- and two-byte characters.

MacOS-J Encoding

Apple's MacOS-J encoding, used for the KanjiTalk6 and KanjiTalk7 character sets, is Shift-JIS plus an additional encoding range for up to 2,444 user-defined characters equal to 26 extra rows of 94 characters each.* This additional encoding range,

* Or 13 rows of 188 characters each when thinking in terms of Shift-JIS encoding.

however, is not compatible with other encodings such as ISO-2022-JP and EUC-JP—they do not convert to valid code points in those encoding methods.

Table D-24 illustrates the encoding space for MacOS-J.

Table D-24: MacOS-J Encoding Specifications

	Decimal	Hexadecimal
ASCII/JIS-Roman		
Byte ranges	33–126, 128, 253–255	21–7E, 80, FD–FF
Half-width katakana		
Byte range	161–223	A1–DF
Two-byte characters		
First byte ranges	129–159, 224–239	81–9F, E0–EF
Second byte ranges	64–126, 128–252	40–7E, 80–FC
User-defined characters		
First byte range	240–252	F0–FC
Second byte ranges	64–126, 128–252	40–7E, 80–FC

MacOS-J and Shift-JIS encodings

The encoding method used by MacOS-J is more or less identical to Shift-JIS, except that there is an extra user-defined character encoding area and four additional single-byte code points (0x80 and 0xFD–0xFF). Both MacOS-J character sets, KanjiTalk6 and KanjiTalk7, use the same encoding as described above.

Microsoft Japanese Encoding

The encoding used by the Microsoft Japanese character set is simply Shift-JIS. Characters from IBM Japanese and NEC Kanji have been included. The Shift-JIS user-defined region is required because it is used to encoded the IBM Japanese Selected characters.

NEC Kanji Encoding

NEC Kanji can be encoded similarly to ISO-2022-JP, Shift-JIS, and EUC-JP encodings, depending on the environment. The ISO-2022–like implementation is sometimes referred to as NEC-JIS. ASCII/JIS-Roman (half-width) and half-width katakana are also in the same encoding space as kanji, and thus can be encoded using two bytes. This can simplify the handling of text streams (no need to have any state shifting between one- and two-byte modes), but significantly increases the storage requirements for documents not written in Japanese. Table D-25 lists its specifications.

Table D-25: NEC Kanji Encoding Specifications

	Decimal	Hexadecimal	Graphical (ASCID)
JIS-Roman			
Byte range	33—126	21—7E	
JIS C 6226-1978			
First byte range	33—126	21—7E	
Second byte range	33—126	21—7E	
NEC Extended Set			
First byte range	161—254	A1—FE	
Second byte range	161—254	A1—FE	
Escape sequences			
JIS-Roman	27 72	1B 48	<ESC> H
JIS C 6226-1978	27 75	1B 4B	<ESC> K
JIS8 half-width katakana			
Byte range	161—223	A1—DF	

Refer to Tables 4-42 and 4-32 for descriptions of the Shift-JIS and EUC-JP implementations of the NEC Kanji character set on pages 176 and 166, respectively. Note that Shift-JIS does not support the NEC Extended Character Set, but includes the 360 IBM Selected Kanji and 14 of the 28 IBM Selected Non-kanji encoded in rows 89 through 92.

NEC Kanji and ISO-2022-JP encodings

The escape sequences for NEC Kanji encoding are unique in that they are made up of the escape character followed by only a single printable character. While this makes the escape sequences shorter, it does not leave much context with which you may insert lost escape characters (the restoration of lost escape characters and otherwise mangled escape sequences was discussed in Chapter 4 and Chapter 7, *Typography*).

NTT Kanji Encoding

NTT Kanji encoding is much like ISO-2022-JP encoding, except that there is an additional two-byte character escape sequence defined for the extended character set. See Table D-26 for a listing of its encoding specifications.

Table D-26: NTT Kanji Encoding Specifications

	Decimal	Hexadecimal	Graphical (ASCID)
One-byte characters			
Byte range	33—126	21—7E	

Table D-26: NTT Kanji Encoding Specifications (continued)

	Decimal	Hexadecimal	Graphical (ASCII)
Two-byte characters			
First byte range	33—126	21—7E	
Second byte range	33—126	21—7E	
Escape sequences			
JIS-Roman	27 40 74	1B 28 4A	<ESC> (J
JIS C 6226-1978	27 36 64	1B 24 40	<ESC> $ @
NTT Extended	27 36 41 48	1B 24 29 30	<ESC> $) 0

NTT Kanji and ISO-2022-JP encodings

NTT Kanji encoding is like ISO-2022-JP plus an additional two-byte escape sequence for the extended character set. Note that only the JIS C 6226-1978 character set is supported through a single escape sequence.

TRON Encoding

The encoding used by operating systems based on TRON (such as BTRON) is a mixed one- and two-byte modal encoding. The two-byte encoding region includes four zones, designated "A" through "D." Zone A is used for encoding JIS X 0208:1997, Zone B for JIS X 0212-1990, Zone C for GB 2312-80, and Zone D for KS X 1001:1992. Table D-27 provides the complete TRON encoding definition.

Table D-27: TRON Encoding Specifications

	Decimal	Hexadecimal
Single-byte characters		
Byte ranges	33—126, 128—253	21—7E, 80—FD
Two-byte control characters		
First byte	0	00
Second byte range	0—254	00—FE
Zone "A" (8,836 code points)		
First byte range	33—126	21—7E
Second byte range	33—126	21—7E
Zone "B" (11,844 code points)		
First byte range	128—253	80—FD
Second byte range	33—126	21—7E
Zone "C" (11,844 code points)		
First byte range	33—126	21—7E
Second byte range	128—253	80—FD

Table D-27: TRON Encoding Specifications (continued)

	Decimal	Hexadecimal
Zone "D" (15,876 code points)		
First byte range	128–253	80–FD
Second byte range	128–253	80–FD
Two-byte language specifiers (94)		
First byte	254	FE
Second byte range	33–126	21–7E
One-byte language specifiers (127)		
First byte	254	FE
Second byte range	128–254	80–FE
Special codes (94)		
First byte	255	FF
Second byte range	33–126	21–7E
Escapes (127)		
First byte	255	FF
Second byte range	128–254	80–FE
EOF (End-Of-File) mark		
First byte	255	FF
Second byte	255	FF

JIS X 0212-1990 is encoded in Zone B of TRON encoding, which represents a 126×94 matrix. So, how does a 94×94 character set fit into this matrix? The last row of JIS X 0212-1990 is ignored, forming a 93×94 matrix that fits into the highest bounds of zone B (0xA121–0xFD7E).

Perhaps of greater curiosity is how GB 2312-80 and KS X 1001:1992, both 94×94 character sets, fit into a 94×126 and 126×126 matrix, respectively. Zero-based code conversion techniques are used to fit them so that they fill up encoding rows consisting of 126 code points each. Table D-28 lists ISO-2022 and EUC encoding ranges, along with the corresponding TRON encoding range.

Table D-28: GB 2312-80 and KS X 1001:1992 in TRON Encoding

Character Set	ISO-2022	EUC	TRON[a]
GB 2312-80	2121–7E7E	A1A1–FEFE	2180–678F
KS X 1001:1992	2121–7E7E	A1A1–FEFE	B780–FD8F

[a] Although the second-byte values appear to end at 0x8F, it does, in fact, use the range 0x80–0xFD. This is an artifact of zero-based code conversion.

The language specifiers are used for the modal aspect of TRON encoding. There are two language specifiers that are always defined in TRON. 0xFE21 is for "Japanese" (two-byte language), and 0xFE80 is for "English" (one-byte language). There

are 93 remaining two-byte language specifiers, and 126 remaining one-byte language specifiers.

There are three types of "space" characters according to TRON encoding, as illustrated in Table D-29.

Table D-29: Three Types of "Space" Characters in TRON Encoding

	Japanese Mode	English Mode
Full-width space	2121	*not applicable*
Half-width space	00A0	A0
Proportional space	0020	20

Figure D-7 illustrates the four two-byte zones and other encoding regions defined by TRON encoding.

The algorithms for converting between TRON encoding and encodings that support GB 2312-80 (ISO-2022-CN and EUC-CN) and KS X 1001:1992 (ISO-2022-KR and EUC-KR) can be found in the section entitled "TRON Code Conversion," starting on page 1015 in Appendix W, *Perl Code Examples*.

Korean Vendor Encodings

Nearly all Korean vendor encodings are based on EUC-KR encoding. Exceptions to this are some of the IBM encodings. The following sections provide encoding tables for a number of Korean vendor encodings.

DEC Korean Encoding

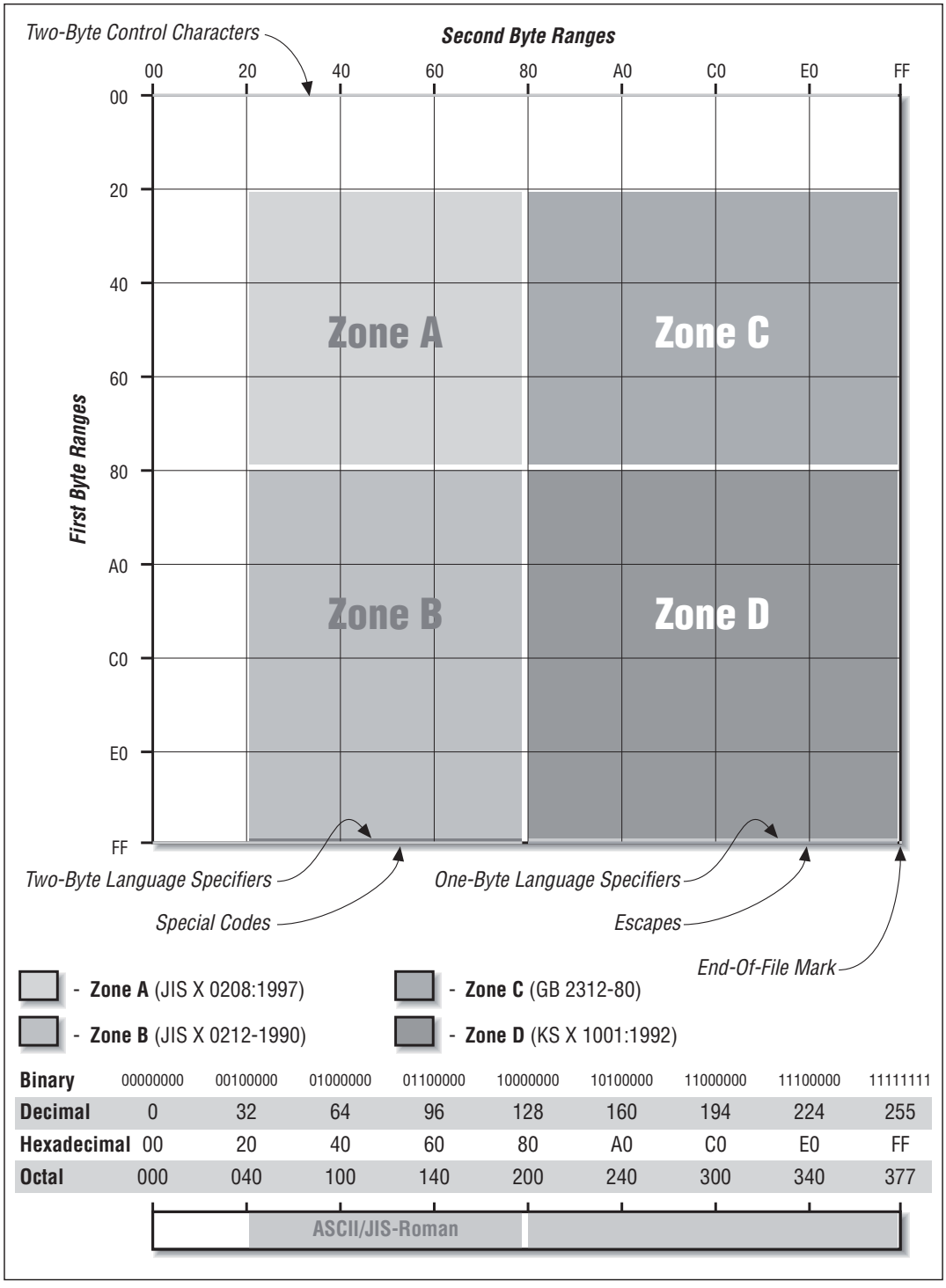
DEC Korean encoding is identical to EUC-KR encoding, meaning that it is a simple mixed one- and two-byte encoding that encodes the ASCII/KS-Roman plus KS X 1001:1992 character sets.

HangulTalk Encoding

Table D-30 provides the complete specification for HangulTalk encoding, which includes an enlarged two-byte region to accommodate an extension defined by Elex Computer, and adopted by Apple Computer. Also note that the second byte range is shortened to 0xA1 through 0xFD to accommodate one single-byte character at 0xFE.

IBM Korean Encodings

As you observed in previous sections regarding Chinese and Japanese encodings, IBM has developed a number of encodings for use on their various operating systems. The following sections describe some of IBM's Korean encodings.



Binary	00000000	00100000	01000000	01100000	10000000	10100000	11000000	11100000	11111111
Decimal	0	32	64	96	128	160	194	224	255
Hexadecimal	00	20	40	60	80	A0	C0	E0	FF
Octal	000	040	100	140	200	240	300	340	377

Figure D-7: TRON encoding tables

Table D-30: HangulTalk Encoding Specifications

	Decimal	Hexadecimal
ASCII/KS-Roman		
Byte ranges	33—126, 129—131, 254—255	21—7E, 81—83, FE—FF
Two-byte characters		
First byte range	161—253	A1—FD
Second byte ranges	65—125, 129—254	41—7D, 81—FE

IBM Korean DBCS-PC encoding

Table D-31 provides the encoding specifications for IBM Korean DBCS-PC encoding. It is important to note how the IBM Selected Characters, which are encoded in the range of rows 0x9A through 0xA0, are encoded in a separate block, not mixed in with the rest of the characters. In IBM DBCS-Host encoding, they are mixed in with the rest of the characters.

Table D-31: IBM Korean DBCS-PC Encoding Specifications

	Decimal	Hexadecimal
ASCII/KS-Roman		
Byte range	33—126	21—7E
User-defined characters[a]		
First byte ranges	143—154, 201, 254	8F—9A, C9, FE
Second byte range	161—254	A1—FE
IBM Selected Characters[b]		
First byte range	154—160	9A—A0
Second byte range	161—254	A1—FE
KS X 1001:1992[c]		
First byte ranges	161—172, 176—253	A1—AC, B0—FD
Second byte range	161—254	A1—FE
Reserved		
First byte ranges	129—143, 173—175	81—8E, AD—AF
Second byte range	161—254	A1—FE

[a] The range 0x9AA6 through 0x9AFE is for IBM Selected Characters.
[b] The first IBM Selected Character is at 0x9AA6; 0x9AA1 through 0x9AA5 are user-defined code points.
[c] The last defined character in this region is 0xFDFE—the same as EUC-KR encoding.

IBM Korean DBCS-Host encoding

Table D-32 provides the encoding specifications for IBM Korean DBCS-Host encoding, which clearly shows that characters are allocated slightly differently than

in DBCS-PC encoding. Like other instances of IBM DBCS-Host encoding, shifting characters are used for switching between one- and two-byte modes.

Table D-32: IBM Korean DBCS-Host Encoding Specifications

	Decimal	Hexadecimal
One-byte characters		
Byte range	65—249	41—F9
Full-width space		
First byte	64	40
Second byte	64	40
Two-byte characters[a]		
First byte ranges	65—75, 80—108, 132—211	41—4B, 50—6C, 84—D3
Second byte range	65—254	41—FE
User-defined characters		
First byte range	212—221	D4—DD
Second byte ranges	65—127, 129—253	41—7F, 81—FD
Shifting characters		
One-byte character	15	0F
Two-byte character	14	0E
Reserved		
First byte ranges	76—79, 109—131, 222—254	4C—4F, 6D—83, DE—FE
Second byte range	65—254	41—FE

[a] The last defined character in this region is 0xD3B7. Also, the range 0x8441–0xD3FE is not only similar to Johab encoding, but the hangul are encoded according to the Johab encoding principles.

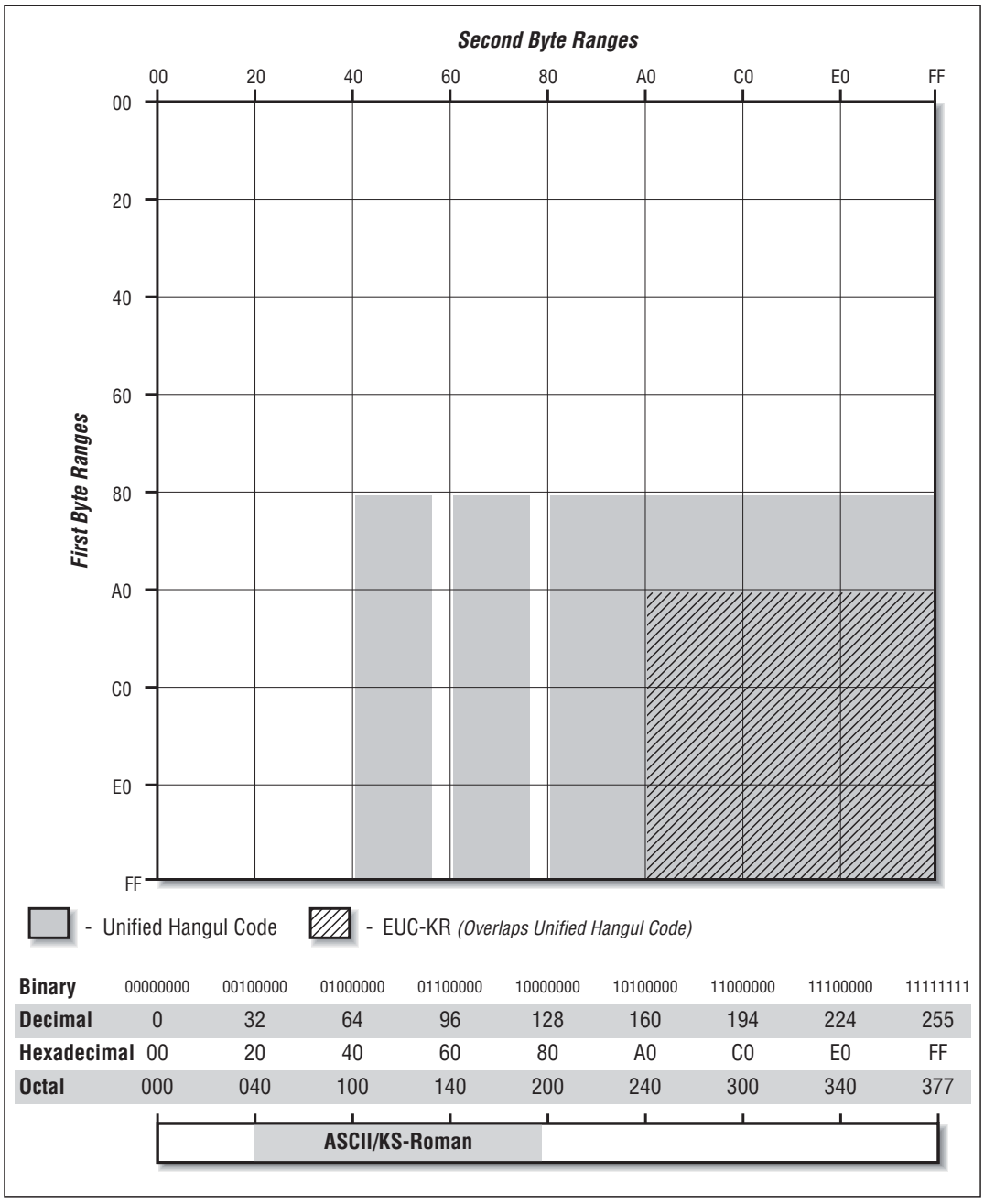
Unified Hangul Code Encoding

The encoding for Unified Hangul Code (UHC) is simply an extension to EUC-KR encoding that is used to encode 8,822 additional hangul. It is supported by the Korean version of Microsoft Windows 95. Table D-33 provides the full specifications for Unified Hangul Code encoding.

Table D-33: Unified Hangul Code Encoding Specifications

	Decimal	Hexadecimal
ASCII/KS-Roman		
Byte range	33—126	21—7E
Two-byte characters		
First byte range	129—254	81—FE
Second byte ranges	65—90, 97—122, 129—254	41—5A, 61—7A, 81—FE

Figure D-8 illustrates three distinct two-byte regions defined by Unified Hangul Code encoding. One region is a superset of EUC-KR encoding (0xA1A1–0xFEFE), and the others' sole purpose is to encode the additional 8,822 hangul.



Binary	00000000	00100000	01000000	01100000	10000000	10100000	11000000	11100000	11111111
Decimal	0	32	64	96	128	160	194	224	255
Hexadecimal	00	20	40	60	80	A0	C0	E0	FF
Octal	000	040	100	140	200	240	300	340	377

Figure D-8: Unified Hangul Code encoding tables