APPENDIX A

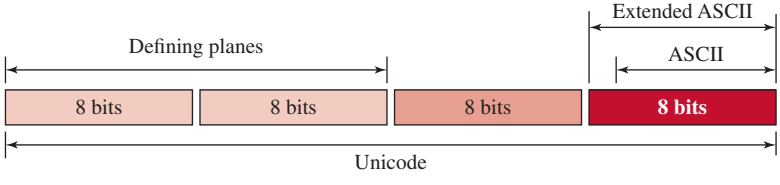
Unicode

Computers use numbers. They store characters by assigning a number for each one. The original coding system was called ASCII (American Standard Code for Information Interchange) and had 128 symbols (0 to 127) each stored as a 7-bit number. ASCII could satisfactorily handle lowercase and uppercase letters, digits, punctuation characters, and some control characters. An attempt was made to extend the ASCII character set to 8 bits. The new code, which was called Extended ASCII, was never internationally standardized.

To overcome the difficulties inherent in ASCII and Extended ASCII, the Unicode Consortium (a group of multilingual software manufacturers) created a universal encoding system to provide a comprehensive character set called ***Unicode*.**

Unicode was originally a 2-byte character set. Unicode version 3, however, is a 4-byte code and is fully compatible with ASCII and Extended ASCII. The ASCII set, which is now called *Basic Latin,* is Unicode with the most significant 25 bits set to zero. Extended ASCII, which is now called Latin-1, is Unicode with the most significant 24 bits set to zero. Figure A.1 shows how the different systems are compatible.

Figure A.1 *Unicode bytes*



Each character or symbol in this code is defined by a 32-bit number. The code can define up to 2^{32} (4,294,967,296) characters or symbols. The notation uses hexadecimal digits in the following format.

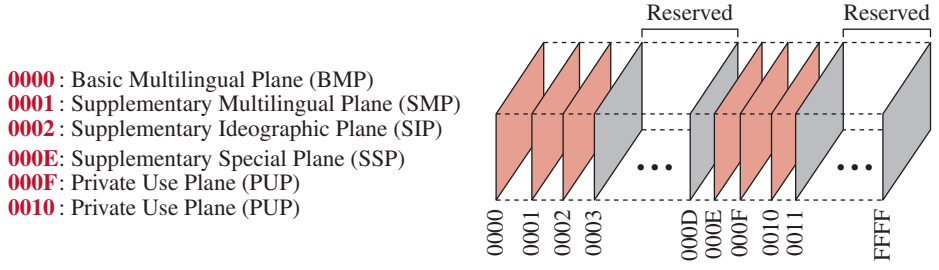
U-XXXXXXXX

Each X is a hexadecimal digit. Therefore, the numbering goes from U-00000000 to U-FFFFFFFF.

A.1 PLANES

Unicode divides the available space codes into planes. The most significant 16 bits define the plane, which means we can have 65,536 planes. Each plane can define up to 65,536 characters or symbols. Figure A.2 shows the structure of Unicode spaces and planes.

Figure A.2 *Unicode planes*



A.1.1 Basic Multilingual Plane (BMP)

Plane $(0000)_{16}$, the basic multilingual plane (BMP), is designed to be compatible with the previous 16-bit Unicode. The most significant 16 bits in this plane are all zeros. The codes are normally shown as U+XXXX with the understanding that XXXX defines only the least significant 16 bits. This plane mostly defines character sets in different languages with the exception of some codes used for control or other special characters.

A.1.2 Other Planes

There are some other (nonreserved) planes, which we briefly describe below:

Supplementary Multilingual Plane (SMP)

Plane $(0001)_{16}$, the supplementary multilingual plane (SMP), is designed to provide more codes for those multilingual characters that are not included in the BMP.

Supplementary Ideographic Plane (SIP)

Plane $(0002)_{16}$, the supplementary ideographic plane (SIP), is designed to provide codes for ideographic symbols, symbols that primarily denote an idea (or meaning) in contrast to a sound (or pronunciation).

Supplementary Special Plane (SSP)

Plane $(000E)_{16}$, the supplementary special plane (SSP), is used for special characters.

Private Use Planes (PUPs)

Planes (000F) and $(0010)_{16}$, private use planes (PUPs), are for private use.

A.2 ASCII

The American Standard Code for Information Interchange (ASCII) is a 7-bit code that was designed to provide code for 128 symbols, mostly in American English. Today, ASCII, or Basic Latin, is part of Unicode. It occupies the first 128 codes in Unicode (00000000 to 0000007F). Table A.1 contains the hexadecimal and graphic codes (symbols). The codes in hexadecimal just define the two least significant digits in Unicode. To find the actual code, we prepend 000000 in hexadecimal to the code.

Table A.1 *ASCII Codes*

Symbol	*Hex*	*Symbol*	*Hex*	*Symbol*	*Hex*	*Symbol*	*Hex*
NULL	00	SP	20	@	40	`	60
SOH	01	!	21	A	41	a	61
STX	02	"	22	B	42	b	62
ETX	03	#	23	C	43	c	63
EOT	04	$	24	D	44	d	64
ENQ	05	%	25	E	45	e	65
ACK	06	&	26	F	46	f	66
BEL	07	'	27	G	47	g	67
BS	08	(	28	H	48	h	68
HT	09	)	29	I	49	i	69
LF	0A	*	2A	J	4A	j	6A
VT	0B	+	2B	K	4B	k	6B
FF	0C	,	2C	L	4C	l	6C
CR	0D	–	2D	M	4D	m	6D
SO	0E	.	2E	N	4E	n	6E
SI	0F	/	2F	O	4F	o	6F
DLE	10	0	30	P	50	p	70
DC1	11	1	31	Q	51	q	71
DC2	12	2	32	R	52	r	72
DC3	13	3	33	S	53	s	73
DC4	14	4	34	T	54	t	74
NAK	15	5	35	U	55	u	75
SYN	16	6	36	V	56	v	76
ETB	17	7	37	W	57	w	77
CAN	18	8	38	X	58	x	78
EM	19	9	39	Y	59	y	79
SUB	1A	:	3A	Z	5A	z	7A
ESC	1B	;	3B	[	5B	{	7B
FS	1C	<	3C	\	5C	|	7C
GS	1D	=	3D	]	5D	}	7D
RS	1E	>	3E	^	5E	~	7E
US	1F	?	3F	_	5F	DEL	7F

A.2.1 Some Properties of ASCII

ASCII has some interesting properties that we briefly mention here.

1. The space character $(20)_{16}$, is a printable character. It prints a blank space.
2. The uppercase letters start from $(41)_{16}$. The lowercase letters start from $(61)_{16}$. When compared, uppercase letters are numerically smaller than lowercase letters. This means that in a sorted list based on ASCII values, the uppercase letters appear before the lowercase letters.
3. The uppercase and lowercase letters differ by only one bit in the 7-bit code. For example, character *A* is $(1000001)_2$ and character *a* is $(1100001)_2$. The difference is in bit 6, which is 0 in uppercase letters and 1 in lowercase letters. If we know the code for one case, we can easily find the code for the other by adding or subtracting $(20)_{16}$, or we can just flip the sixth bit.
4. The uppercase letters are not immediately followed by lowercase letters. There are some punctuation characters in between.
5. Digits (0 to 9) start from $(30)_{16}$. This means that if you want to change a numeric character to its face value as an integer, you need to subtract $(30)_{16} = 48$ from it.
6. The first 32 characters, $(00)_{16}$ to $(1F)_{16}$, and the last character, $(7F)_{16}$, are nonprintable characters. Character $(00)_{16}$ simply is used as a delimiter to define the end of a character string. Character $(7F)_{16}$ is the delete character used by some programming languages to delete the previous character. The rest of the nonprintable characters are referred to as *control characters* and are used in data communication. Table A.2 gives the description of these characters.

Table A.2 *ASCII Codes*

Symbol	*Interpretation*	*Symbol*	*Interpretation*
SOH	Start of heading	DC1	Device control 1
STX	Start of text	DC2	Device control 2
ETX	End of text	DC3	Device control 3
EOT	End of transmission	DC4	Device control 4
ENQ	Enquiry	NAK	Negative acknowledgment
ACK	Acknowledgment	SYN	Synchronous idle
BEL	Ring bell	ETB	End of transmission block
BS	Backspace	CAN	Cancel
HT	Horizontal tab	EM	End of medium
LF	Line feed	SUB	Substitute
VT	Vertical tab	ESC	Escape
FF	Form feed	FS	File separator
CR	Carriage return	GS	Group separator
SO	Shift out	RS	Record separator
SI	Shift in	US	Unit separator
DLE	Data link escape		