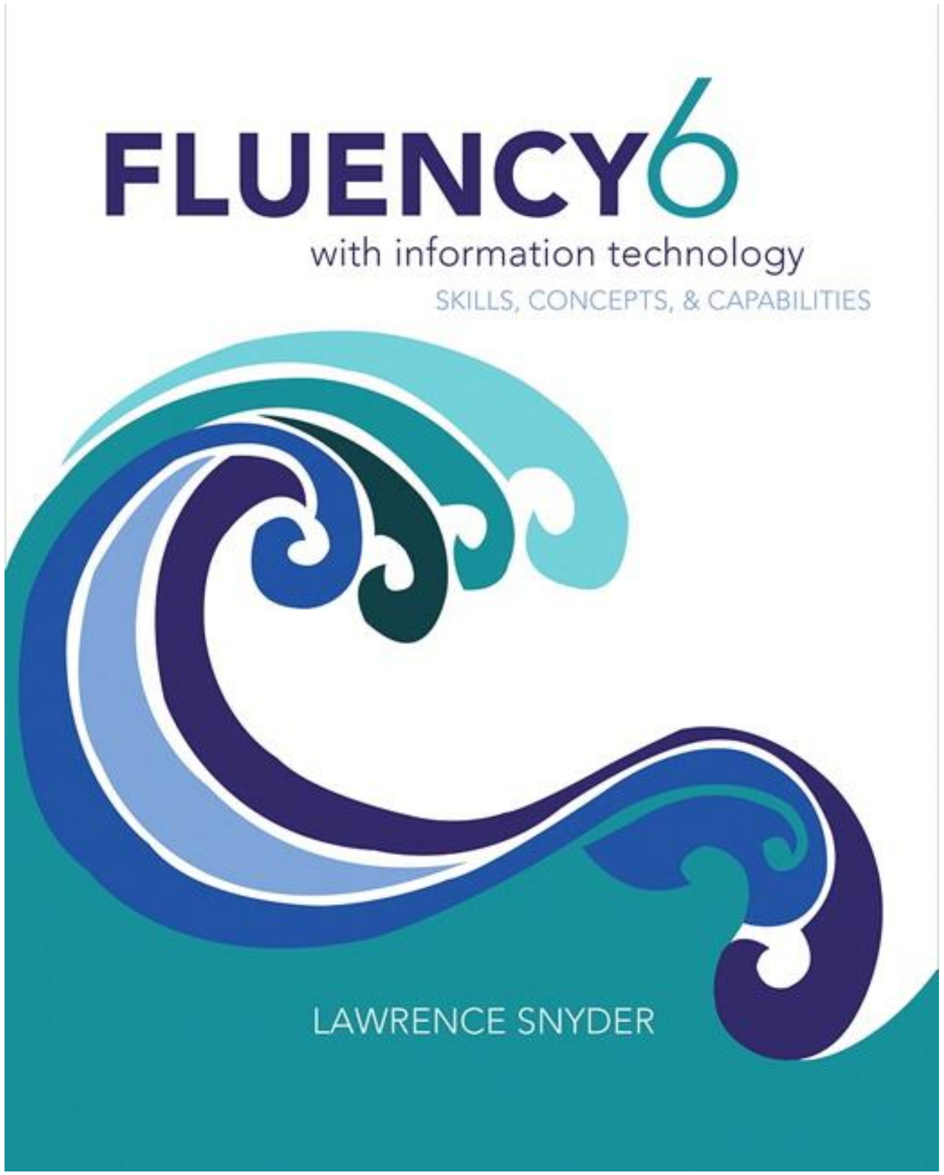


Chapter 7

Representing Information Digitally

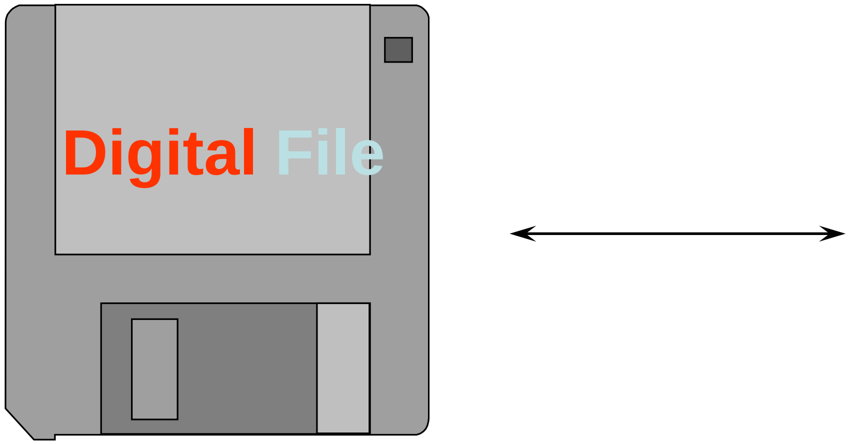
Table of Contents

Learning Objectives

- Explain the link between patterns, symbols, and information
- Determine possible PandA encodings using a physical phenomenon
- Encode and decode ASCII
- Represent numbers in binary form
- Compare the minimal encoding vs the long encoding
- Explain how structure tags (metadata) encode the Oxford English Dictionary

Digital : 통일된 정보표현방법

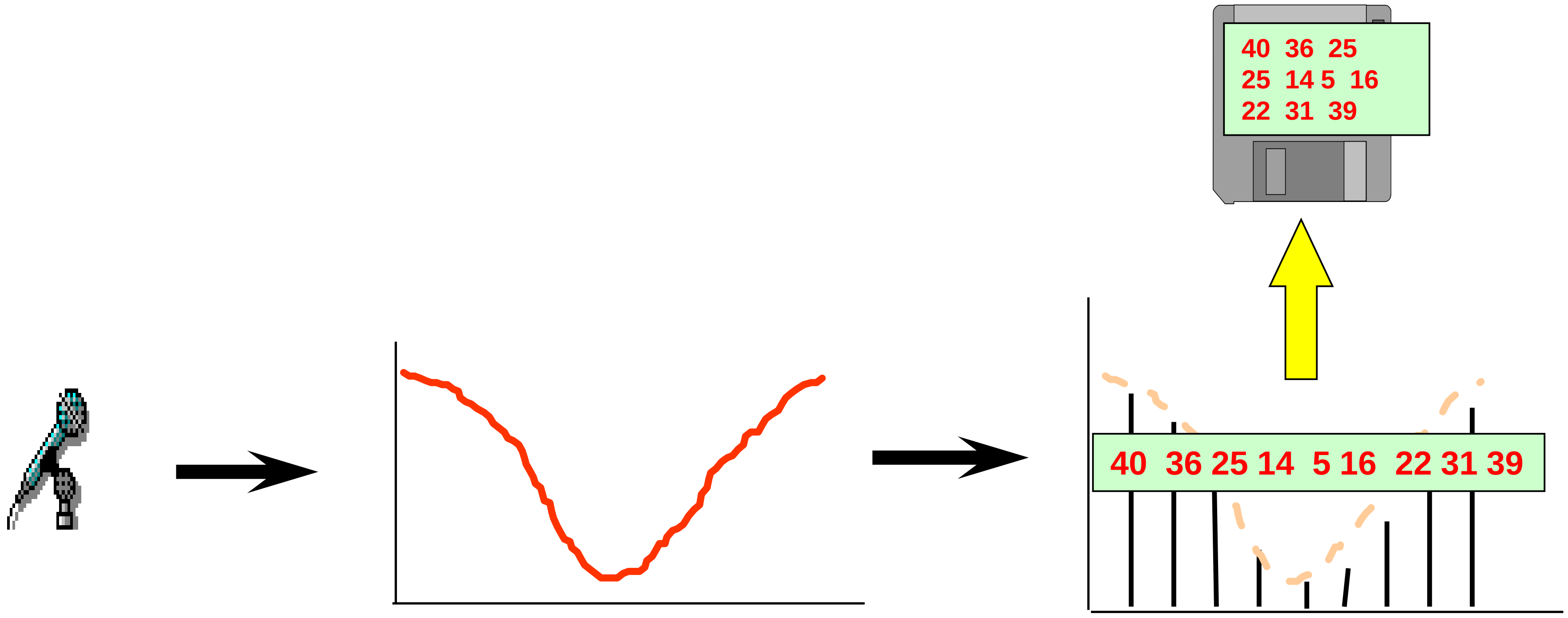
**모든 정보는 digital로 표현될 수 있다



- **수**
- **글씨(Text)**
- **소리(Audio)**
- **화면(Image)**
- **동화상(Video)**

모든 Multimedia 정보는 숫자 (digit)로 표현가능!!

Audio Data

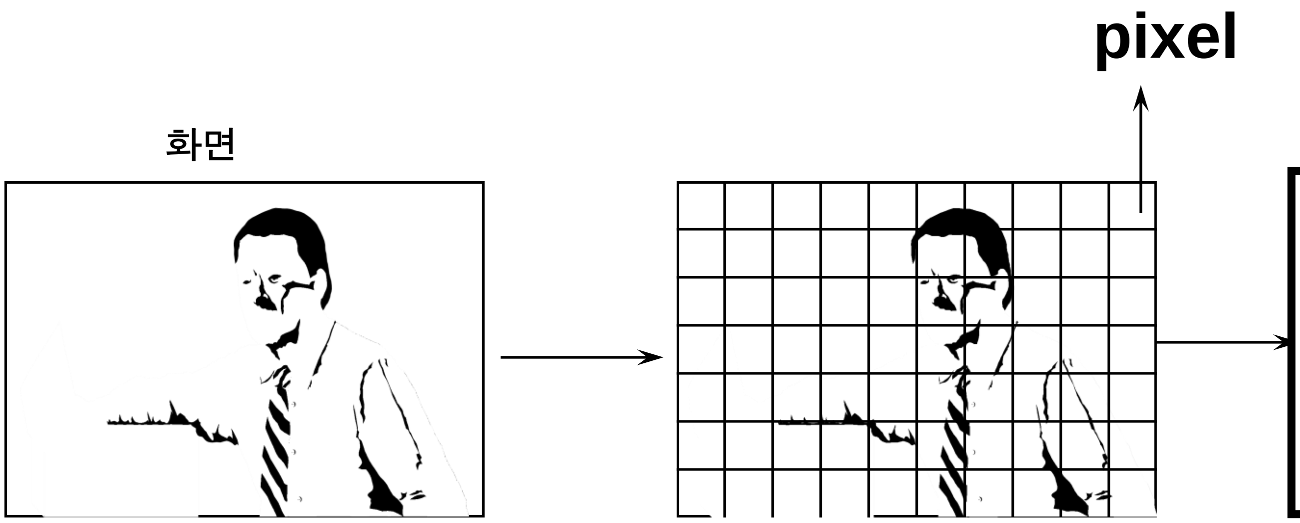


Analog 신호 **Digital 신호**

컴퓨터용 마이크 ==> A/D 변환
컴퓨터용 스피커 ==> D/A 변환

모든 Multimedia 정보는 숫자 (digit)로 표현가능!!

Image Data



```
00  00..  13 14.. 00 00
00  00..  15 19.. 00 00
00  00..  07 11.. 00 00
00  00..  06 03.. 00 00
00  07..  09 03.. 18 00
```

각 **pixel** 당 **Black**의 **density**를 나타내는 숫자

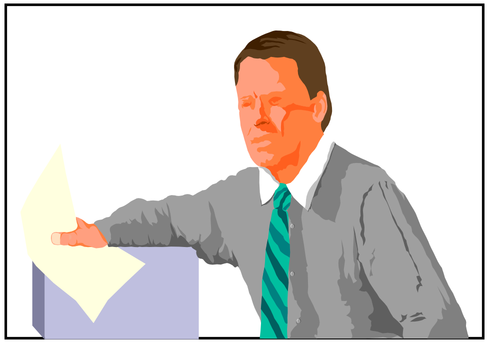
빛의 세기 $\longrightarrow$ **전기 신호 (analog)** $\longrightarrow$ **수치화 (digital)**

모든 Multimedia 정보는 숫자 (digit)로 표현가능!!

보충자료

Color Image Data

화면: RGB Monitor (빛)



→

12 4 34 57 99 12
……

56 7 13 44 66 23
12 4 34 57 99 12
……

12 8 12 33 99 12
56 7 13 44 66 23
12 4 34 57 99 12
……

** 동영상(Video) --- 초당 25 - 30 개의 정지화상(image)을 교체

보충자료

모든 Multimedia 정보는 숫자 (digit)로 표현가능!!

무게 온도 수질 당뇨

디지털

60 Kg **38** 도 **6** PH **110** mg/dl

Digitizing Discrete Information

- The dictionary definition of digitize is to represent information with digits
- Digit means the 10 Arabic numerals : 0 through 9

Limitation of Digits

- A limitation of the dictionary definition of “digitize” is that it calls for the use of the ten digits, which produces a whole number
- Phone No., SS No., ISBN No. are not quantities
  - Arithmetic are not meaningful on Phone No., SS No., ISBN No.
  - Having a bigger telephone number does not make you a better person
    - Any meaning?   880-1830 > 880-1827

Symbols, Briefly

- One practical advantage of digits is that digits have short names (one, two, etc)
- Imagine speaking your phone number the multiple syllable names:
  - “asterisk, exclamation, closing parenthesis…..”
- IT uses these symbols, and names are getting shorter:

exclamation point is bang (!), asterisk is star (*), percent is per (%), closing parenthesis is cp (( )

–The advantage of brevity is not limited to digits

rather than using {1, 2, 3, 4, 5, 6, 7, 8, 9, 0}, uses the symbol set {!, @, #, $, %, ^, &, *, (,)}

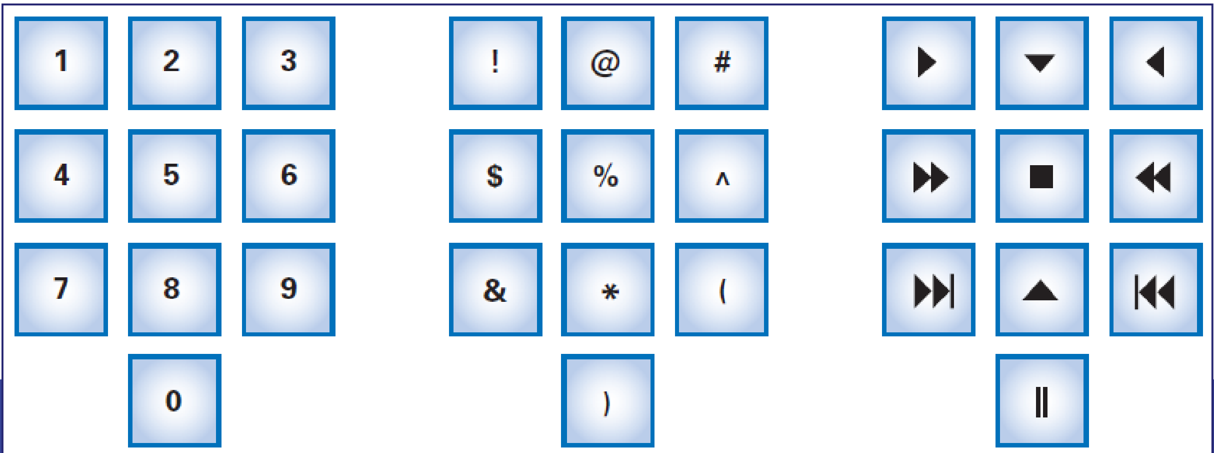
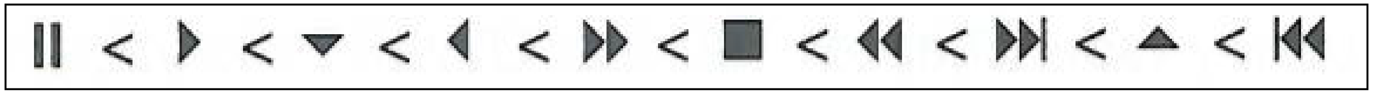


Figure 7.1 Three symbol assignments for a telephone keypad.

Ordering Symbols

- Another advantage of digits is that the items can be listed in numerical order
- Sometimes ordering items is useful
- Collating sequence: placing information in order by using non-digit symbols
  - need to agree on an ordering for the basic symbols



- Today, digitizing means representing information by symbols
- Then… which symbols would be best?

- Collate: 대조확인하다, 페이지순서를 맞추다

Fundamental Information Representation

- The fundamental patterns used in computing come into play when the physical world meets the logical world
- In the physical world, the most fundamental form of information is the presence or absence of a physical phenomenon
- From a digital information point of view, the amount of a phenomenon is not important as long as it is reliably detected
  - Whether there is some information or none;
  - Whether it is present or absent
- In the logical world, concepts of true and false are important
  - Logic is the foundation of reasoning and also the foundation of computing
- The physical world can implement the logical world by associating "true" with the presence of a phenomenon and "false" with its absence

The PandA Representation

- PandA is the name used for 2 fundamental patterns of digital information:
  - The mnemonic for “Presence and Absence”
- A key property of PandA is that the phenomenon is either present or not
- The presence or absence can be viewed as “true” or “false”
- Such a formulation is said to be discrete
- Discrete means “distinct” or “separable”
  - It is not possible to transform one value into another by tiny gradations
  - There are no “shades of gray”

* gradation: 점진적 변화

A Binary System

- The PandA encoding has two patterns: present and absent
- Two patterns make it a binary system
- There is no law that says on means "present" or off means "absent"

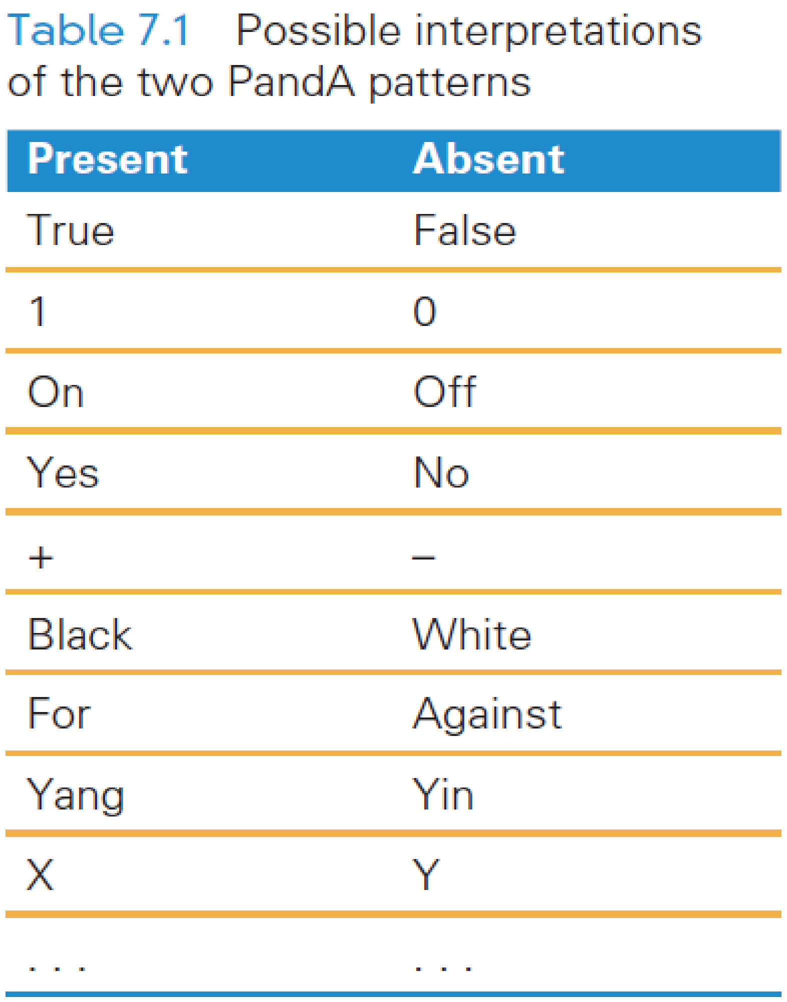


Table 7.1 Possible interpretations of the two PandA patterns

Present	Absent
True	False
1	0
On	Off
Yes	No
+	–
Black	White
For	Against
Yang	Yin
X	Y
. . .	. . .

Bits Form Symbols

- In the PandA representation, the unit is a specific place (in space and time), where the presence or absence of the phenomenon can be set and detected.
- The PandA unit is known as a bit
- Bit is a contraction for “binary digit”
- Bit sequences can be interpreted as binary numbers
- Groups of bits form symbols

Bits in Computer Memory

- Memory is arranged inside a computer in a very long sequence of bits
  - The physical phenomenon can be encoded ➔ The information can be set and detected to present or absent

Alternative PandA Encodings

- There is no limit to the ways to encode two physical states
  - Stones on all squares, but with white (absent) and black (present) stones for the two states
  - Multiple stones of two colors per square, more white stones than black means 1 and more black stones than white means 0
  - And so forth…

Combining Bit Patterns

- The two-bit patterns gives limited resources for digitizing information
- Only 2 values can be represented
- The 2 patterns must be combined into sequences to create enough symbols to encode the intended information

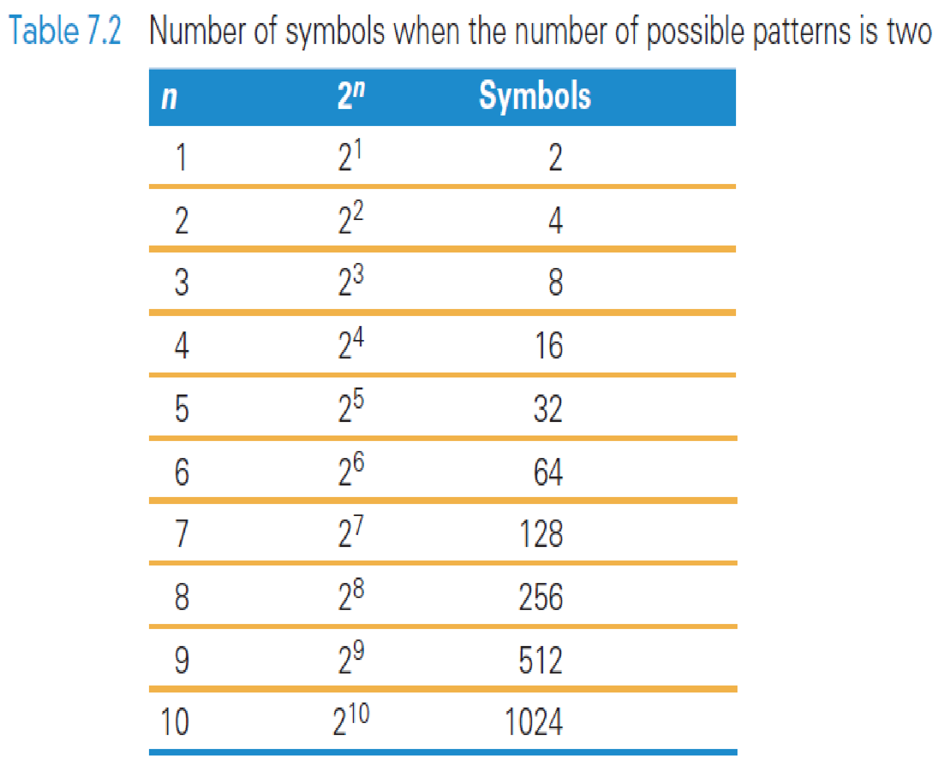


Table 7.2 Number of symbols when the number of possible patterns is two

n	2^n	Symbols
1	2^1	2
2	2^2	4
3	2^3	8
4	2^4	16
5	2^5	32
6	2^6	64
7	2^7	128
8	2^8	256
9	2^9	512
10	2^{10}	1024

Binary Explained

- Computers use base-2 numbers to represent numbers, also known as the binary number system
- When counting in binary you are limited to only use 0 and 1
  - 0, 1, 10, 11, 100, 101, 110, 111, 1000, …

Hex Explained

- Hex digits, short for hexadecimal digits, are base-16 numbers
- Uses decimal numbers, and then the first six Latin letters
  - **0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F**
- There needed to be a better way to write bit sequences… hexadecimal digits

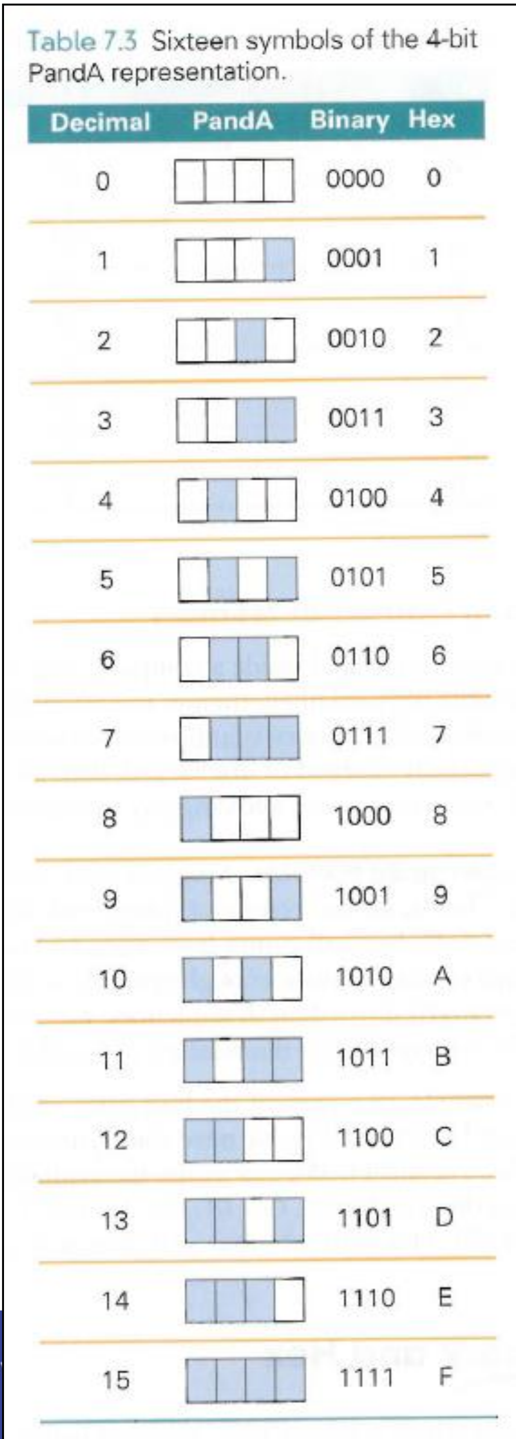


Table 7.3 Sixteen symbols of the 4-bit PandA representation.

Decimal	PandA	Binary	Hex
0	⬜⬜⬜⬜	0000	0
1	⬜⬜⬜🟦	0001	1
2	⬜⬜🟦⬜	0010	2
3	⬜⬜🟦🟦	0011	3
4	⬜🟦⬜⬜	0100	4
5	⬜🟦⬜🟦	0101	5
6	⬜🟦🟦⬜	0110	6
7	⬜🟦🟦🟦	0111	7
8	🟦⬜⬜⬜	1000	8
9	🟦⬜⬜🟦	1001	9
10	🟦⬜🟦⬜	1010	A
11	🟦⬜🟦🟦	1011	B
12	🟦🟦⬜⬜	1100	C
13	🟦🟦⬜🟦	1101	D
14	🟦🟦🟦⬜	1110	E
15	🟦🟦🟦🟦	1111	F

Changing Hex to Binary

- The 32 bits below represent a computer instruction
    - <u>1000 1110</u> <u>1101 1000</u> <u>1010 0011</u> <u>1010 0000</u>
- Writing so many 0's and 1's is tedious and error prone
- We can convert each group of bits (8) above to hex <u>8E</u> <u>D8</u> <u>A3</u> <u>A0</u>

Digitizing Numbers in Binary

- The two earliest uses of PandA were to:
    - Encode numbers
    - Encode keyboard characters
- Representations for sound, images, video, and other types of information are also important

Place Value in a Decimal Number

- Recall that to find the quantity expressed by a decimal number:
  - The digit in a place is multiplied by the place value and the results are added
- Example, 1010 (base 10) is:
  - Digit in the 1's place is multiplied by its place value
  - Digit in the 10's place is multiplied by its place value
  - and so on:

➔ $(0 \times 1) + (1 \times 10) + (0 \times 100) + (1 \times 1000)$

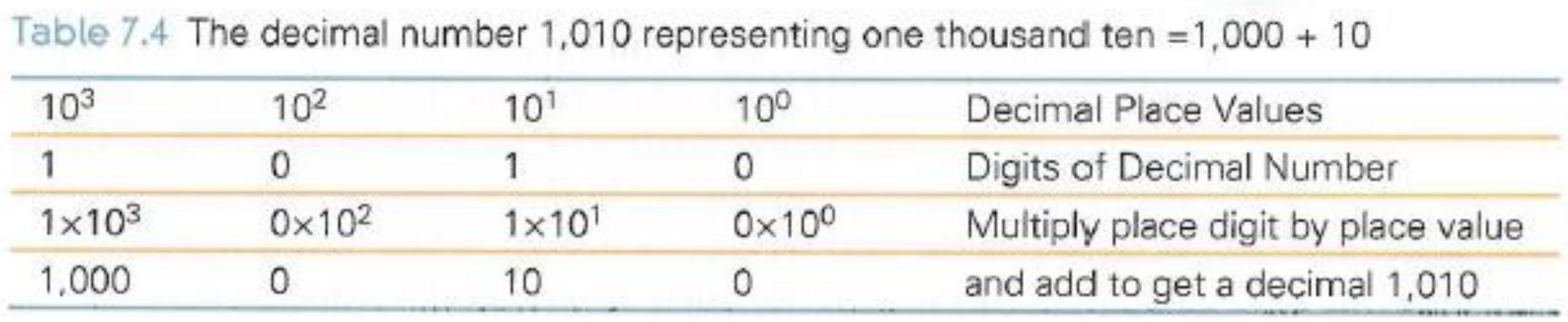


Table 7.4 The decimal number 1,010 representing one thousand ten =1,000 + 10

10^3	10^2	10^1	10^0	Decimal Place Values
1	0	1	0	Digits of Decimal Number
1×10^3	0×10^2	1×10^1	0×10^0	Multiply place digit by place value
1,000	0	10	0	and add to get a decimal 1,010

Place Value in a Binary Number

- Binary works the same way
- The base is 2 instead of 10
- Instead of the decimal place values:

  1, 10, 100, 1000, . . . ,

  the binary place values are:

  1, 2, 4, 8, 16, . . . ,

Power	Decimal	Binary
0	$1 = 10^0$	$1 = 2^0$
1	$10 = 10^1$	$2 = 2^1$
2	$100 = 10^2$	$4 = 2^2$
3	$1000 = 10^3$	$8 = 2^3$
4	$10{,}000 = 10^4$	$16 = 2^4$
. . .	. . .	. . .

- 1010 in binary:

  $(1 \times 8) + (0 \times 4) + (1 \times 2) + (0 \times 1)$

Table 7.5 The binary number 1010, representing the decimal number ten = 8 + 2

2^3	2^2	2^1	2^0	Binary Place Values
1	0	1	0	Bits of Binary Number
1×2^3	0×2^2	1×2^1	0×2^0	Multiply place bit by place value
8	0	2	0	and add to get a decimal 10

Digitizing Text

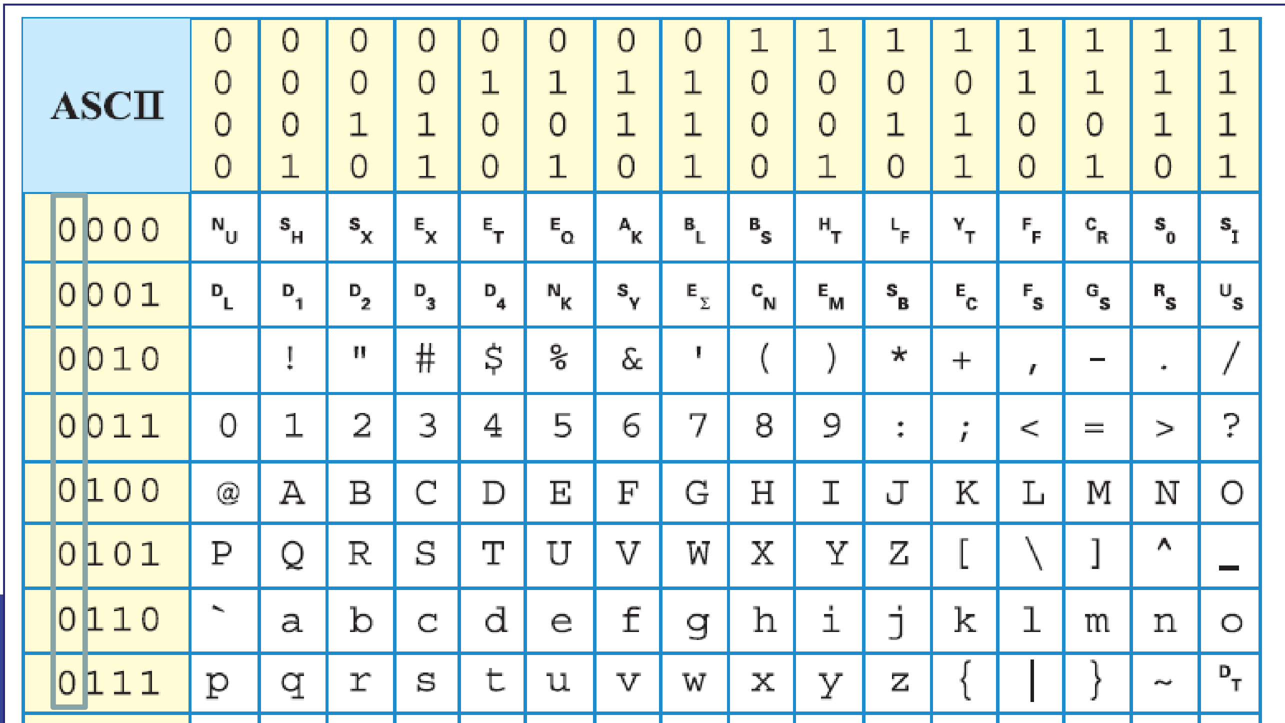
- The number of bits determines the number of symbols available for representing values:
  - n bits in sequence yield 2^n symbols
- The more characters you want encoded, the more symbols you need
- Roman letters, Arabic numerals, and about a dozen punctuation characters are the minimum needed to digitize English text
- What about:
  - Basic arithmetic symbols like +, −, *, /, = ?
  - Characters not required for English ö, é, ñ, ø ?
  - Punctuation mark (구두점) !, ?, :, /, ; ?
  - What about business symbols: ¢, £, ¥, ©, and ® ?

Assigning Symbols

- We need to represent:
  - 26 uppercase, 26 lowercase letters, 10 numerals,
  - 20 punctuation characters, 10 useful arithmetic characters,
  - 3 other characters (new line, tab, and backspace)
  - 95 symbols…enough for English
- To represent 95 distinct symbols, we need 7 bits
  - 6 bits gives only 2^6 = 64 symbols
  - 7 bits give 2^7 = 128 symbols
- 128 symbols is ample for the 95 different characters needed for English characters
- Some additional characters must also be represented

ASCII: a 7-bit code

- ASCII stands for American Standard Code for Information Interchange
- ASCII is a widely used 7-bit (2^7) code
- Advantages of a “standard”:
  - Computer parts built by different manufacturers can be connected
  - Programs can create data and store it so that other programs can process it later, and so forth

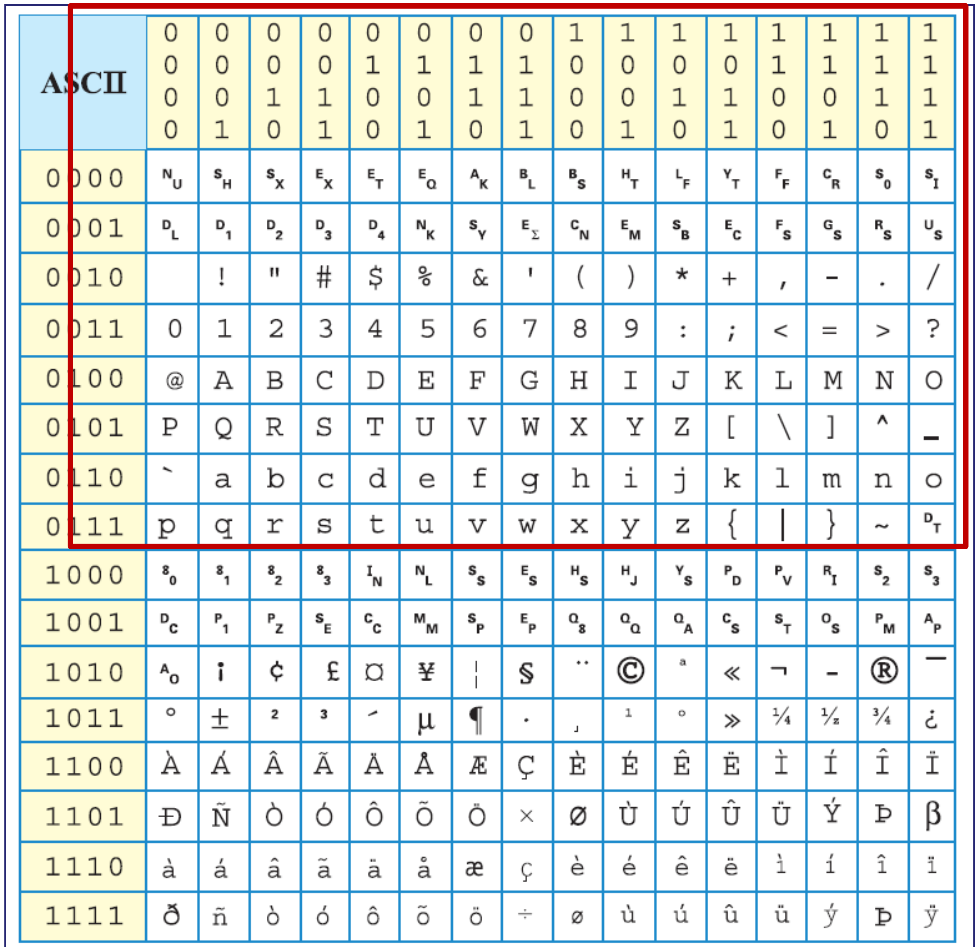


ASCII	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0000	N_U	S_H	S_X	E_X	E_T	E_Q	A_K	B_L	B_S	H_T	L_F	Y_T	F_F	C_R	S_0	S_I
0001	D_L	D_1	D_2	D_3	D_4	N_K	S_Y	E_Σ	C_N	E_M	S_B	E_C	F_S	G_S	R_S	U_S
0010		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
0011	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
0100	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
0101	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
0110	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
0111	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	D_T

PEARSON

Extended ASCII: An 8-Bit Code

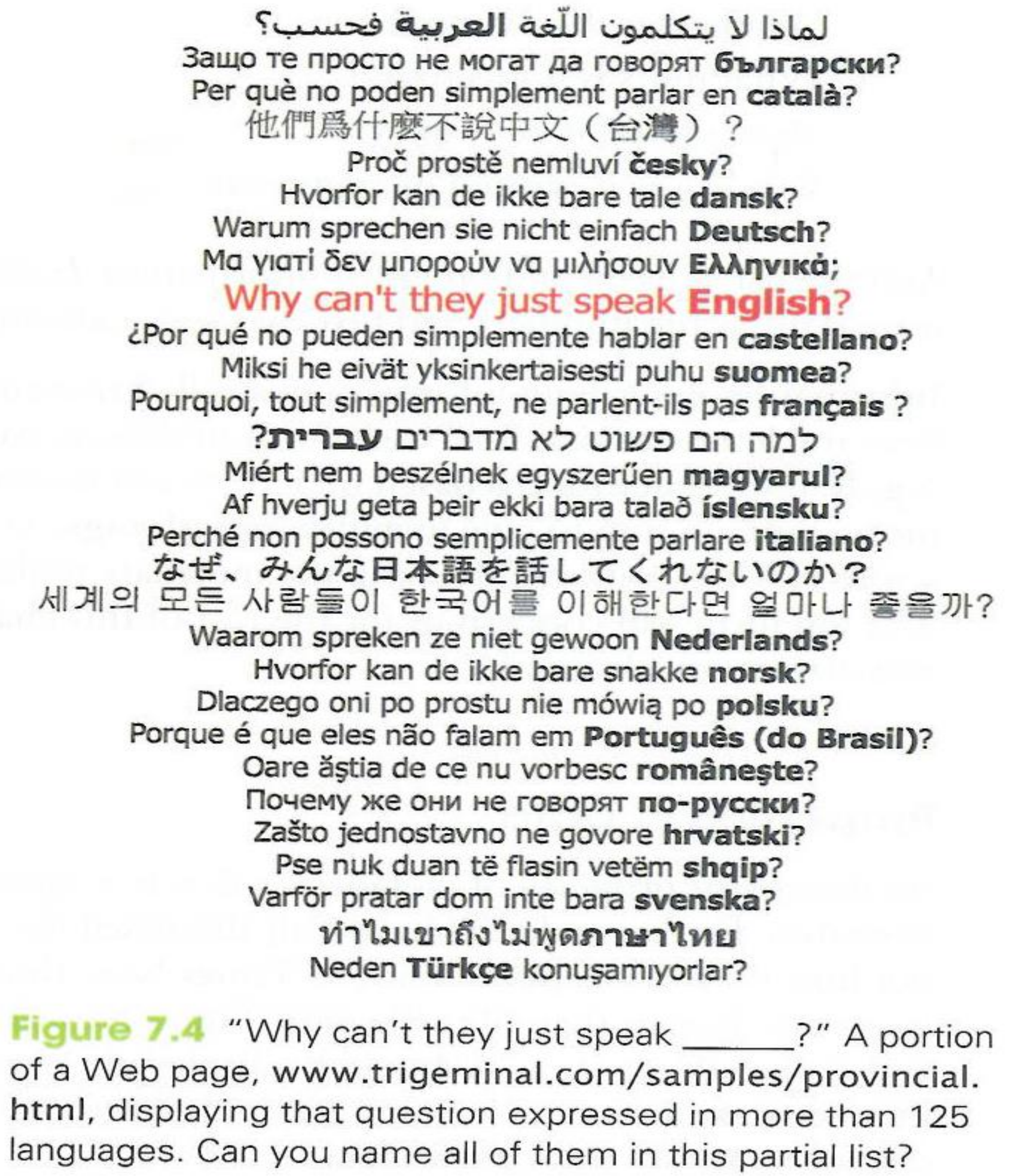
- 7-bit ASCII is not enough, it cannot represent text from other languages
- IBM decided to use the next larger set of symbols, the 8-bit symbols (2^8)
- Eight bits produce 2^8 = 256 symbols
  - The 7-bit ASCII is the 8-bit ASCII representation with the leftmost bit set to 0
  - Handles many languages that derived from the Latin alphabet
- Handling other languages is solved in two ways:
  - recoding the second half of Extended ASCII for the language
  - using the multibyte Unicode representation
- IBM gave 8-bit sequences a special name, byte
- It is a standard unit for computer memory



ASCII	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0000	$^{N}_{U}$	$^{S}_{H}$	$^{S}_{X}$	$^{E}_{X}$	$^{E}_{T}$	$^{E}_{Q}$	$^{A}_{K}$	$^{B}_{L}$	$^{B}_{S}$	$^{H}_{T}$	$^{L}_{F}$	$^{Y}_{T}$	$^{F}_{F}$	$^{C}_{R}$	$^{S}_{0}$	$^{S}_{I}$
0001	$^{D}_{L}$	$^{D}_{1}$	$^{D}_{2}$	$^{D}_{3}$	$^{D}_{4}$	$^{N}_{K}$	$^{S}_{Y}$	$^{E}_{\Sigma}$	$^{C}_{N}$	$^{E}_{M}$	$^{S}_{B}$	$^{E}_{C}$	$^{F}_{S}$	$^{G}_{S}$	$^{R}_{S}$	$^{U}_{S}$
0010		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
0011	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
0100	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
0101	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
0110	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
0111	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	$^{D}_{T}$
1000	$^{8}_{0}$	$^{8}_{1}$	$^{8}_{2}$	$^{8}_{3}$	$^{I}_{N}$	$^{N}_{L}$	$^{S}_{S}$	$^{E}_{S}$	$^{H}_{S}$	$^{H}_{J}$	$^{Y}_{S}$	$^{P}_{D}$	$^{P}_{V}$	$^{R}_{I}$	$^{S}_{2}$	$^{S}_{3}$
1001	$^{D}_{C}$	$^{P}_{1}$	$^{P}_{Z}$	$^{S}_{E}$	$^{C}_{C}$	$^{M}_{M}$	$^{S}_{P}$	$^{E}_{P}$	$^{Q}_{8}$	$^{Q}_{Q}$	$^{Q}_{A}$	$^{C}_{S}$	$^{S}_{T}$	$^{O}_{S}$	$^{P}_{M}$	$^{A}_{P}$
1010	$^{A}_{O}$	¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	¯
1011	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
1100	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
1101	Đ	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	β
1110	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
1111	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Figure 7.3 ASCII, the American Standard Code for Information Interchange.

Note: The original 7-bit ASCII is the top half of the table; the whole table is known as Extended ASCII (ISO-8859-1). The 8-bit symbol for a letter is the four row bits followed by the four column bits (e.g., A = 0100 0001, while z = 0111 1010). Characters shown as two small letters are control symbols used to encode nonprintable information (e.g., $^{B}_{S}$ = 0000 1000 is backspace). The bottom half of the table represents characters needed by Western European languages, such as Icelandic's eth (ð) and thorn (þ).



لماذا لا يتكلمون اللّغة **العربية** فحسب؟
Защо те просто не могат да говорят **български**?
Per què no poden simplement parlar en **català**?
他們爲什麼不說中文（台灣）？
Proč prostě nemluví **česky**?
Hvorfor kan de ikke bare tale **dansk**?
Warum sprechen sie nicht einfach **Deutsch**?
Μα γιατί δεν μπορούν να μιλήσουν **Ελληνικά**;
Why can't they just speak **English**?
¿Por qué no pueden simplemente hablar en **castellano**?
Miksi he eivät yksinkertaisesti puhu **suomea**?
Pourquoi, tout simplement, ne parlent-ils pas **français** ?
למה הם פשוט לא מדברים **עברית**?
Miért nem beszélnek egyszerűen **magyarul**?
Af hverju geta þeir ekki bara talað **íslensku**?
Perché non possono semplicemente parlare **italiano**?
なぜ、みんな日本語を話してくれないのか？
세계의 모든 사람들이 한국어를 이해한다면 얼마나 좋을까?
Waarom spreken ze niet gewoon **Nederlands**?
Hvorfor kan de ikke bare snakke **norsk**?
Dlaczego oni po prostu nie mówią po **polsku**?
Porque é que eles não falam em **Português (do Brasil)**?
Oare ăştia de ce nu vorbesc **româneşte**?
Почему же они не говорят **по-русски**?
Zašto jednostavno ne govore **hrvatski**?
Pse nuk duan të flasin vetëm **shqip**?
Varför pratar dom inte bara **svenska**?
ทำไมเขาถึงไม่พูด**ภาษาไทย**
Neden **Türkçe** konuşamıyorlar?

Figure 7.4 "Why can't they just speak ______?" A portion of a Web page, **www.trigeminal.com/samples/provincial.html**, displaying that question expressed in more than 125 languages. Can you name all of them in this partial list?

Remember UTF-8?

Special characters [1/2]

* Punctuation Mark (구두점)

apostrophe (’ ')	brackets ([ ], ( ), { }, ⟨ ⟩)	colon (:)
comma (,)	dash (–, –, —, —)	ellipsis (…, ...)
exclamation mark (!)	full stop/period (.)	guillemets (« »)
hyphen (-, -)	question mark (?)	quotation marks (‘ ’, “ ”)
semicolon (;)	slash/stroke (/)	solidus (⁄)

* Uncommon typography (활자체)

asterism (⁂) tee (⊤) up tack (⊥) index/fist (☞) therefore sign (∴)

because sign (∵) interrobang (‽) irony & sarcasm punctuation (⸮)

lozenge (◊) reference mark (※) tie (⁀)

Special Characters [2/2]

* Word dividers: space ( ) ( ) (   ) (␠) (ƀ) (␣) interpunct (·)

* General typography

ampersand (&) at sign (@) asterisk (*) backslash (\ ) bullet (•)

caret (^) copyright symbol (©) currency (generic) (¤)

currency (specific) ₳ ฿ ₵ ¢ ₡ ₢ ₠ $ ₫ ₯ ₯ € ƒ ₣ ₲ ₴ ₭ ℳ ₥ ₦ ₧ ₱ ₰ £ ₨ ₪ ₮ ₩ ¥

dagger (†, ‡) degree (°) ditto mark (〃)

inverted exclamation mark (¡) inverted question mark (¿)

number sign/pound/hash (#) numero sign (№)

ordinal indicator (º, ª) percent etc. (%, ‰, ‱)

pilcrow (¶) prime (′, ″, ‴) registered trademark (®)

section sign (§) service mark (℠) sound recording copyright (℗)

tilde (~) trademark (™) underscore/understrike (_) vertical/broken bar, pipe (|, ¦)

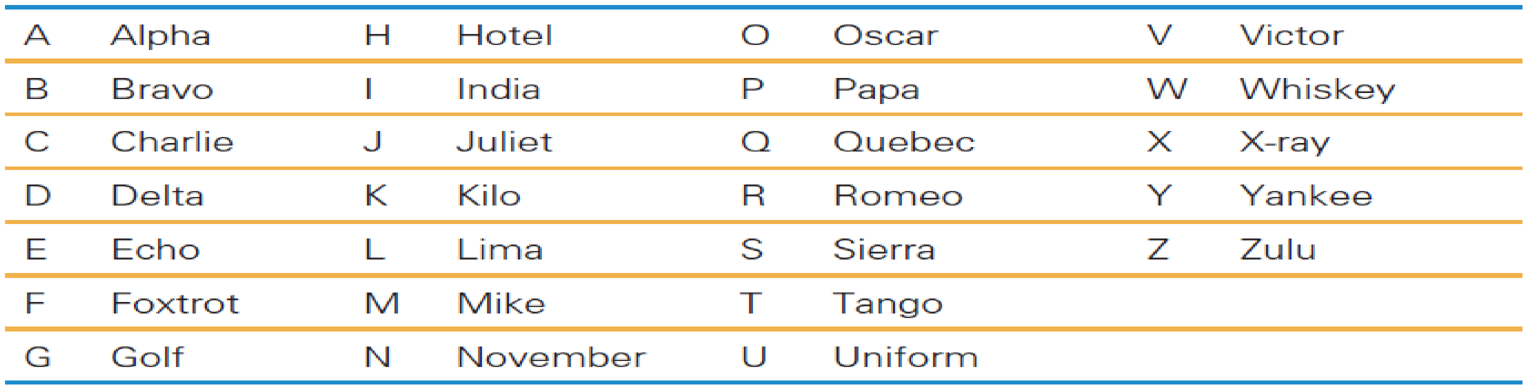
Advantages of Long Encodings

- With computing, we usually try to be efficient by using the shortest symbol sequence to minimize the amount of memory
- Minimal Encoding vs Long Encoding
- Examples of the opposite (Long Encodings):
  - NATO Broadcast Alphabet
  - Bar Codes

NATO Broadcast Alphabet

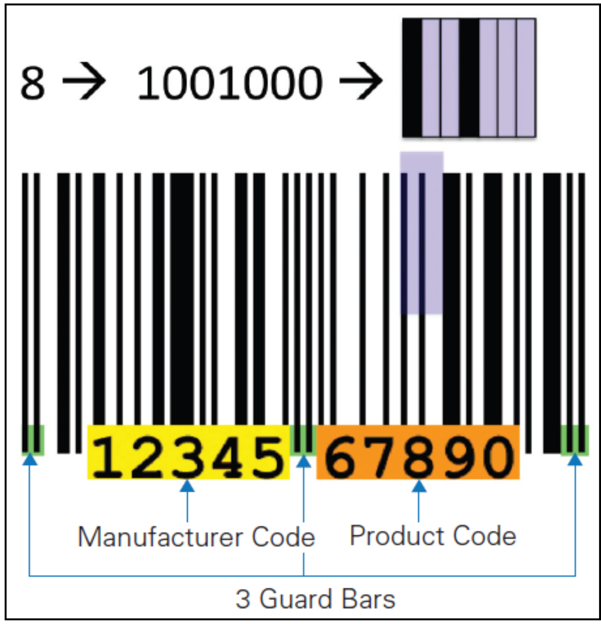
- The code for the letters used in radio communication is purposely inefficient
- The code is distinctive when spoken amid noise
- The alphabet encodes letters as words
  - Words are the encoding symbols
  - “Mike” and “November” replace “M” and “N”
- The longer encoding improves the chance that letters will be recognized
- Digits keep their usual names, except nine, which is known as niner

Table 7.7 NATO broadcast alphabet designed not to be minimal



A	Alpha	H	Hotel	O	Oscar	V	Victor
B	Bravo	I	India	P	Papa	W	Whiskey
C	Charlie	J	Juliet	Q	Quebec	X	X-ray
D	Delta	K	Kilo	R	Romeo	Y	Yankee
E	Echo	L	Lima	S	Sierra	Z	Zulu
F	Foxtrot	M	Mike	T	Tango		
G	Golf	N	November	U	Uniform		

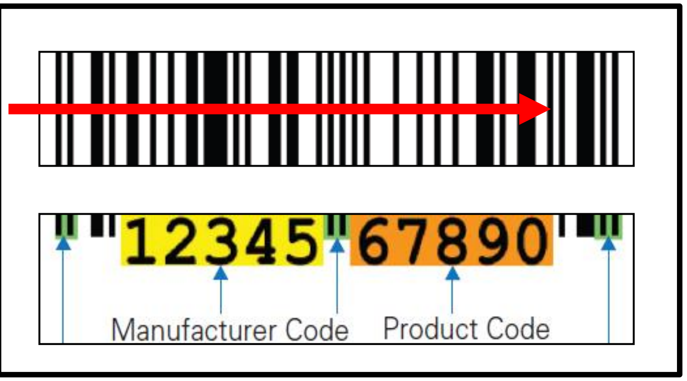
Bar Codes [1/2]

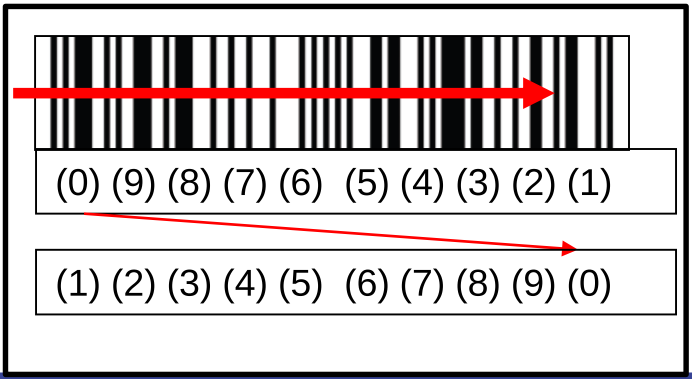


- Universal Product Codes (UPC) also use more than the minimum number of bits to encode information
- In the UPC-A encoding, 7 bits are used to encode the digits 0 – 9
- UPC encodes the manufacturer (left side) and the product (right side)
- Different bit combinations are used for each side
- One side is the complement of the other side
- The bit patterns were chosen to appear as different as possible from each other

Bar Codes [2/2]

- Different encodings for each side make it possible for the scanner to recognize whether the code is right side up or upside down





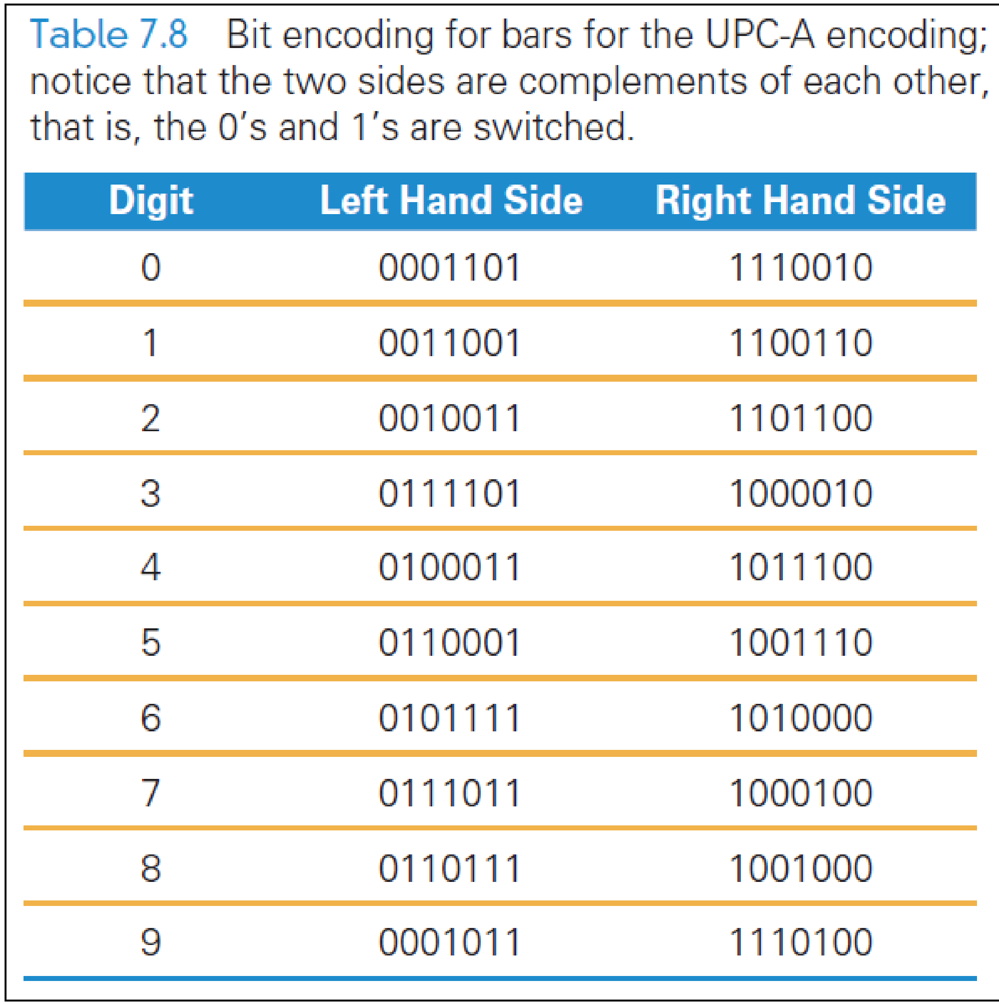


Table 7.8 Bit encoding for bars for the UPC-A encoding; notice that the two sides are complements of each other, that is, the 0's and 1's are switched.

Digit	Left Hand Side	Right Hand Side
0	0001101	1110010
1	0011001	1100110
2	0010011	1101100
3	0111101	1000010
4	0100011	1011100
5	0110001	1001110
6	0101111	1010000
7	0111011	1000100
8	0110111	1001000
9	0001011	1110100

You have only 20 numbers to represent, but you are using 7 bits!

Metadata

- Converting the content into binary is half of the problem of representing information
- The other half of the problem? Describing the information's properties
- Characteristics (properties) of the content also needs to be encoded:
  - – How is the content structured? / What other content is it related to?
  - – Where was it collected? / What units is it given in?
  - – How should it be displayed? / When was it created or captured? / And so on…
- Metadata: information describing information
  - Metadata is the third basic form of data
  - It does not require its own binary encoding
- The most common way to give metadata

  is with tags

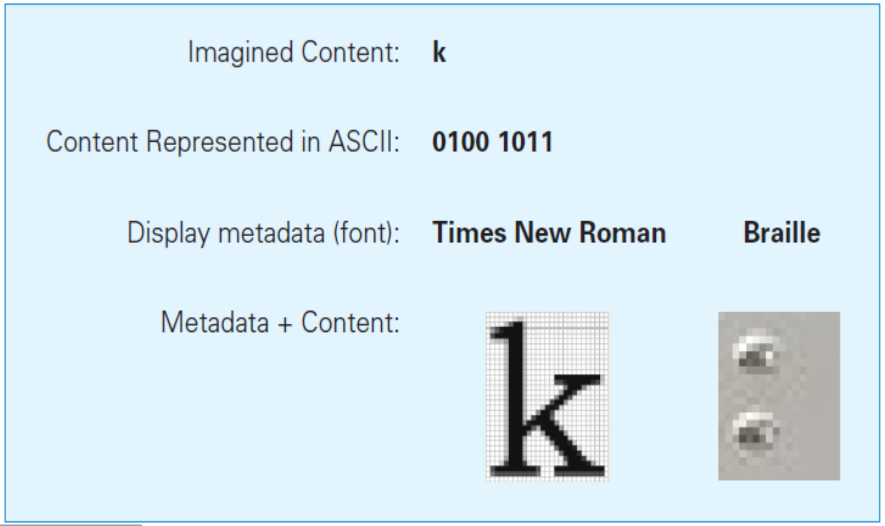


Illustration of the separation of content from the metadata that describes it.

The Oxford English Dictionary

- The OED is the definitive reference for every English word’s meaning, etymology (어원학, 어원설명), and usage
  - The printed version of the OED is truly monumental is 20 volumes, weighs 150 pounds, and fills 4 feet of shelf space
- In 1984, the conversion of the OED to digital form began
- Imagine, with what you know about searching on the computer, finding the definition for the verb “set”:
  - “set” is part of many words
  - You would find closet, horsetail, settle, and more
- Software can help sort out the words

OED’s Structure Tags

- <hw> for a headword (word being defined) / <pr> handles pronunciation
  <ph> does the phonetic notations / <ps> parts of speech /
  <hm> homonym numbers / <e> surrounds the entire entry /
  <hg> surrounds the head group or all of the information at the start of a definition

- With structure tags, software can use a simple algorithm to find what is needed
- Tags do not print and are included only to specify the structure, so the computer knows what part of the dictionary to use
- Structure tags are also useful for formatting…for example, boldface used for headwords
- Knowing the structure makes it possible to generate the formatting information

Why “Byte”?

- In late 1950s, IBM was building a supercomputer, called Stretch
- They needed a word for a quantity of memory between a bit and a word
  - A word of computer memory is typically the amount required to represent computer instructions (currently a word is 32 bits)
- Then, why not bite?
- The ‘i’ to a ‘y’ was done so that someone couldn’t accidentally change ‘bite’ to ‘bit’ by the dropping the ‘e’ ”
  - bite ➔ bit (the meaning changes)
  - byte ➔ byt (what’s a byt?)

Byte and Parity Bit

- Computer memory is subject to errors
- An 9th extra bit (called parity bit) is added to the memory to help detect errors
- Parity refers to whether a number is even or odd
  - Count the number of 1’s in the byte. If even number of 1’s, set the ninth bit to 0
- All 9-bit groups have even parity:
  - Any single bit error in a group causes its parity to become odd
  - This allows hardware to detect that an error has occurred
  - It cannot detect which bit is wrong, however (What if there are two errors?)

Summary

- We began by learning that digitizing doesn't require digits—any symbols will do
- PandA encoding, which is based on the presence and absence of a physical phenomenon
  - Their patterns are discrete; they form the basic unit of a bit. Their names (most often 1 and 0) can be any pair of opposite terms
- A bit's 0 and 1 states naturally encourage the binary representation of numbers
- The early 7-bit ASCII and the Extended or 8-bit ASCII is now the standard
- The need to use more than the minimum number of bits to encode information
- How documents like the Oxford English Dictionary are digitized
  - We learned that tags associate metadata with every part of the OED
  - Using that data, a computer can easily help us find words and other information.
- The mystery of the y in byte