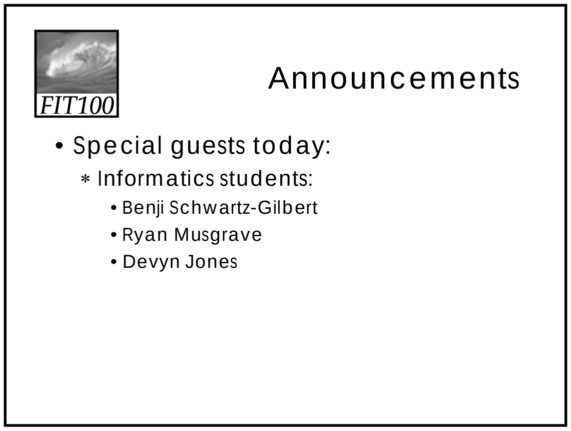
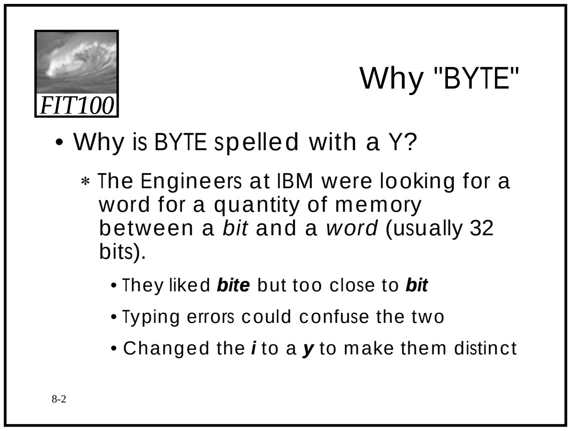


FIT100

Announcements

- Special guests today:
  - Informatics students:
    - Benji Schwartz-Gilbert
    - Ryan Musgrave
    - Devyn Jones

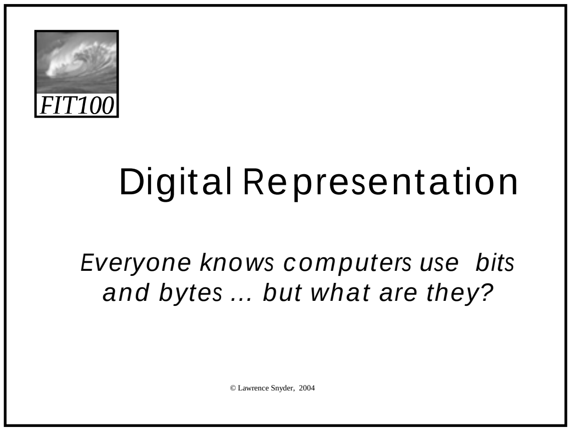


FIT100

Why "BYTE"

- Why is BYTE spelled with a Y?
  - The Engineers at IBM were looking for a word for a quantity of memory between a *bit* and a *word* (usually 32 bits).
    - They liked ***bite*** but too close to ***bit***
    - Typing errors could confuse the two
    - Changed the ***i*** to a ***y*** to make them distinct

8-2

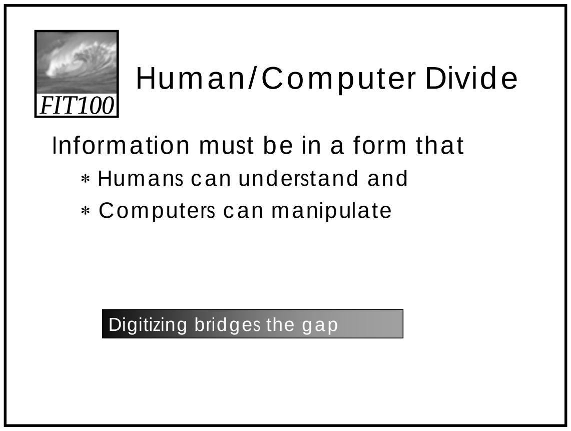


FIT100

Digital Representation

Everyone knows computers use bits and bytes ... but what are they?

© Lawrence Snyder, 2004



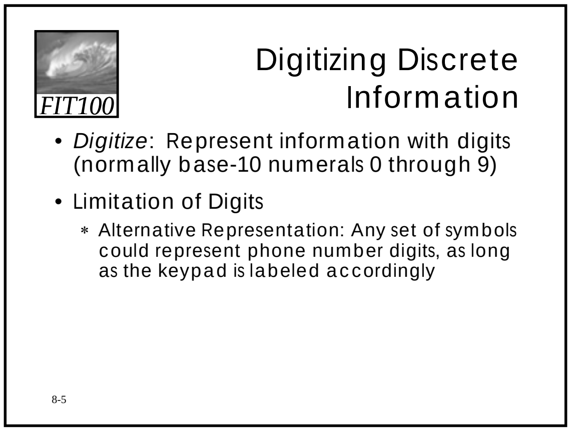
FIT100

Human/Computer Divide

Information must be in a form that

- Humans can understand and
- Computers can manipulate

Digitizing bridges the gap

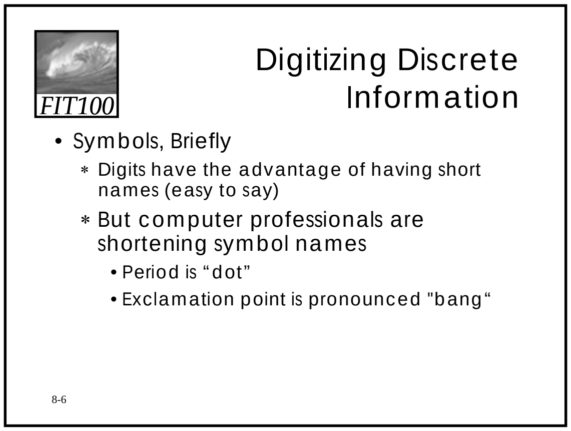


FIT100

Digitizing Discrete Information

- *Digitize*: Represent information with digits (normally base-10 numerals 0 through 9)
- Limitation of Digits
  - Alternative Representation: Any set of symbols could represent phone number digits, as long as the keypad is labeled accordingly

8-5



FIT100

Digitizing Discrete Information

- Symbols, Briefly
  - Digits have the advantage of having short names (easy to say)
  - But computer professionals are shortening symbol names
    - Period is “dot”
    - Exclamation point is pronounced "bang“

8-6

Symbols for Phone Keypad

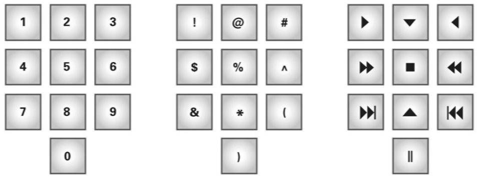


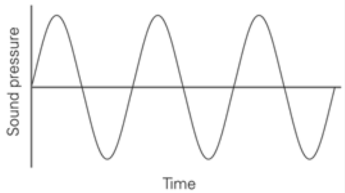
Figure 8.1. Three symbol assignments for a telephone keypad.

Ordering Symbols

- Digits for encoding info
  - Can list items in numerical order
- To use other symbols, we need an ordering system (*collating sequence*)
  - Agreed order from smallest to largest value
- In choosing symbols for encoding, consider how symbols interact with things being encoded

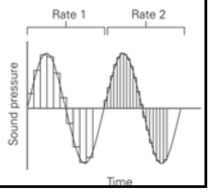
Analog vs. Digital

- Analog is continuous data/information
  - Sound waves



Analog vs. Digital

- Digital is discrete data/information
  - Many distinct samples of data
  - Stored in binary (0's and 1's)
    - All data in a computer is represented in binary



The Fundamental Representation of Information

- The fundamental patterns used in IT come when the physical world meets the logical world
- The most fundamental form of information is the presence or absence of a physical phenomenon
- In the logical world, the concepts of true and false are important
  - Associate true with presence of a phenomenon and false with its absence, we use the physical world to implement the logical world, and produce information technology

PandA Representation

- *PandA* is the mnemonic for "presence and absence"
- It is *discrete* (distinct or separable)—the phenomenon is present or it is not (true or false; 1 or 0). There in no continuous gradation in between.

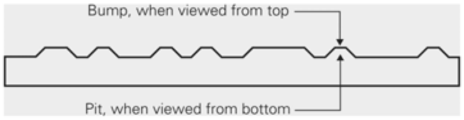
A Binary System

- Two patterns make a *binary system*
  - Base 2 (0's or 1's)
- The basic binary unit is known as a *"bit"* (short for **b**inary dig**it**)
- 8 bits are grouped together to form a *byte*
  - Memory accessed by byte addresses
- We can give any names to these two patterns as long as we are consistent
  - PandA (Presence and Absence can represent 1 and 0, respectively)

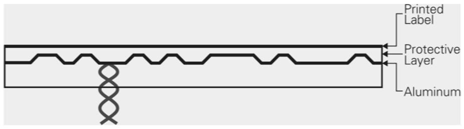
PandA Patterns

Present	Absent
1	0
On	Off
Yes	No
+	-
Black	White
For	Against
Yang	Yin
Lisa	Bart

Encoding Bits on a DVD



Encoding Bits on a DVD



Bits in Computer Memory

- Memory is arranged inside a computer in a very long sequence of bits
  - Bits = places where a phenomenon can be set and detected
- Analogy: Sidewalk "Memory"
  - Each sidewalk square represents a memory slot, or bit, and stones represent the presence or absence
  - If a stone is on the square, the value is 1, if not the value is 0

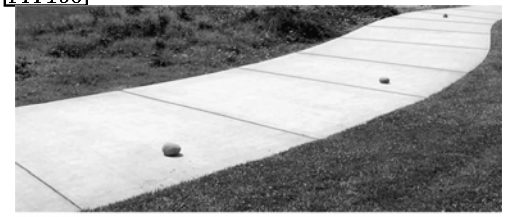


Figure 8.2 Sidewalk sections as a sequence of bits (1010 0010).

Alternative PandA Encodings

- Alternate ways to encode two states using physical phenomena
  - Use stones on all squares, but black stones for one state and white for the other
  - Use multiple stones of two colors per square, saying more black than white means 0 and more white than black means 1
  - Stone in center for one state, off-center for the other
  - etc.

Combining Bit Patterns

- Since we only have two patterns, we must combine them into sequences to create enough symbols to encode necessary information
- Binary (PandA) has 2 patterns, arranging them into *n*-length sequences, we can create 2^n symbols

PandA Patterns

- Number of symbols when the number of possible patterns is two (0 and 1)

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	61
7	2^7	128
8	2^8	256
9	2^9	512
10	2^{10}	1,024

Hex Explained

- Recall in Chapter 4, we specified custom colors in HTML using hex digits
  - e.g., <p bgColor ="#FF8E2A">
  - *Hex* is short for hexadecimal (base 16)
- Why use hex?
  - Writing the sequence of bits is long, tedious, and error-prone

The 16 Hex Digits

- 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
  - 10 = A, 11 = B, ... , 15 = F
- Sixteen values can be represented perfectly by 4-bit sequences (2^4 = 16)
- Changing hex digits to bits and back again:

Binary	0101	1100
Hex	5	C
Hex	3	G
Binary		

Hex (0-9,A-F)

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

FIT100

Digitizing Text

- Early binary representation—1 and 0—encoded numbers and keyboard characters
- Now representation for sound, video, and other types of information
- For encoding text, what symbols should be included?
  - ∗ We want to keep the list small enough to use fewer bits, but we don't want to leave out critical characters

8-25

FIT100

Assigning Symbols

Characters	Quantity
Uppercase letters	26
Lowercase letters	26
Arithmetic characters (0-9)	10
Punctuation characters (including space)	20
Non-Printable characters	3
Total	95

8-26

FIT100

ASCII Character Set

Decimal – Character Set

```
  0 NUL    1 SOH    2 STX    3 ETX    4 EOT    5 ENQ    6 ACK    7 BEL
  8 BS     9 HT    10 NL    11 VT    12 NP    13 CR    14 SO    15 SI
 16 DLE   17 DC1   18 DC2   19 DC3   20 DC4   21 NAK   22 SYN   23 ETB
 24 CAN   25 EM    26 SUB   27 ESC   28 FS    29 GS    30 RS    31 US
 32 SP    33 !     34 "     35 #     36 $     37 %     38 &     39 '
 40 (     41 )     42 *     43 +     44 ,     45 -     46 .     47 /
 48 0     49 1     50 2     51 3     52 4     53 5     54 6     55 7
 56 8     57 9     58 :     59 ;     60 <     61 =     62 >     63 ?
 64 @     65 A     66 B     67 C     68 D     69 E     70 F     71 G
 72 H     73 I     74 J     75 K     76 L     77 M     78 N     79 O
 80 P     81 Q     82 R     83 S     84 T     85 U     86 V     87 W
 88 X     89 Y     90 Z     91 [     92 \     93 ]     94 ^     95 _
 96 `     97 a     98 b     99 c    100 d    101 e    102 f    103 g
104 h    105 i    106 j    107 k    108 l    109 m    110 n    111 o
112 p    113 q    114 r    115 s    116 t    117 u    118 v    119 w
120 x    121 y    122 z    123 {    124 |    125 }    126 ~    127 DEL
```

8-27

FIT100

Hexadecimal ASCII Character Set

Hexadecimal - Character

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

8-28

FIT100

Extended ASCII: An 8-bit Code

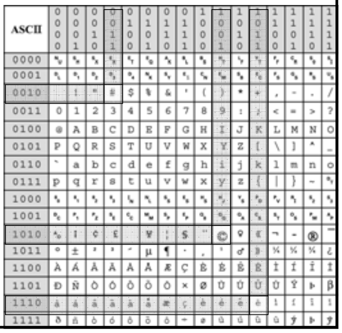
- By the mid-1960's, it became clear that 7-bit ASCII was not enough to represent text from languages other than English
- IBM extended ASCII to 8 bits (256 symbols)
- Called "*Extended ASCII*," the first half is original ASCII with a 0 added at the beginning of each group of bits
- Handles most Western languages and additional useful symbols

8-29

FIT100

ASCII

Character	Binary
#	0010 0011
©	1010 1001
è	1110 1011



ASCII	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0000	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
0001	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
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	{	|	}	~	[illegible]
1000	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
1001	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
1010	[illegible]	¡	¢	£		¥	¦	§	¨	©	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	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
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

ASCII

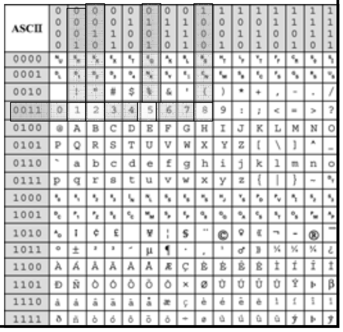
FIT100

Digits in Phone number	Binary for 888 555 1212

888	0011 1000
	0011 1000
	0011 1000

5	0011 0101
	0011 0101
	0011 0101

1	0011 0001
2	0011 0010
1	0011 0001
2	0011 0010



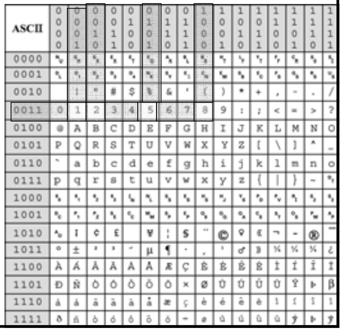
ASCII	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0000	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
0001	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
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	{	|	}	~	[illegible]
1000	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
1001	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
1010	[illegible]	¡	¢	£		¥	¦	§	¨	©	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	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
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

ASCII

FIT100

Digits in Phone number	Binary for 888 555 1212
8	0011 1000
5	0011 0101
1	0011 0001
2	0011 0010

0011 1000
0011 1000
0011 1000



ASCII	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0000	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
0001	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
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	{	|	}	~	[illegible]
1000	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
1001	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
1010	[illegible]	¡	¢	£		¥	¦	§	¨	©	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	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
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

Unicode

FIT100

- Several languages around the world have more than 256 individual characters
- Unicode uses 16 bits; 2^{16} = 65536 characters
  - 1st 7 bits (128 chars) are ASCII chars
  - Different locales – different characters beyond 1st 7 bits

NATO Broadcast Alphabet

FIT100

- The code for broadcast communication is purposefully inefficient, to be distinctive when spoken amid noise

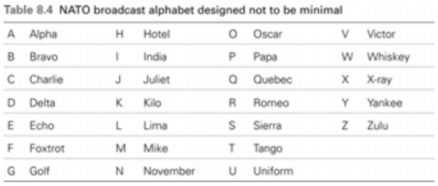


Table 8.4 NATO broadcast alphabet designed not to be minimal

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		

Metadata

FIT100

- Extended ASCII encodes letters and characters well, but most documents contain more than just text.
  - Format information like font, font size, justification
- Formatting characters could be added to ASCII, but that mixes the content with the description of its form (*metadata*)
  - Metadata is "data about data"
- Metadata is represented using tags, as in HTML

Encoding Information

FIT100

Bits and bytes encode the information, but that's not all

- Tags encode format and some structure in word processors
- Tags encode format and some structure in HTML
- In the *Oxford English Dictionary* tags encode structure and some formatting

Using Tags to Encode

- Oxford English Dictionary (OED) printed version is 20 volumes
- We could type the entire contents as ASCII characters (in about 120 years), but searching would be difficult
  - Suppose you search for the word "set." It is included in many other words like closet, horsetail, settle, etc.
  - How will the software know what characters comprise the definition of set?
    - Incorporate metadata

Structure Tags

- Special set of tags was developed to specify OED's structure
  - <hw> means headword, the word being defined
  - Other tags label pronunciation <pr>, phonetic notation <ph>, parts of speech <ps>
- The tags do not print. They are there only to specify structure so the computer knows what part of the dictionary it is looking at

OED Entry For Byte

byte (baIt). *Computers*. [Arbitrary, prob. influenced by bit *sb.*^{4} and bite *sb.*] A group of eight consecutive bits operated on as a unit in a computer. **1964** ***Blaauw*** & ***Brooks*** in *IBM Systems Jrnl.* III. 122 An 8-bit unit of information is fundamental to most of the formats [of the System/360]. A consecutive group of *n* such units constitutes a field of length *n*. Fixed-length fields of length one, two, four, and eight are termed bytes, halfwords, words, and double words respectively. **1964** *IBM Jrnl. Res. & Developm.* VIII. 97/1 When a byte of data appears from an I/O device, the CPU is seized, dumped, used and restored. **1967** ***P. A. Stark*** *Digital Computer Programming* xix. 351 The normal operations in fixed point are done on four bytes at a time. **1968** *Dataweek* 24 Jan. 1/1 Tape reading and writing is at from 34,160 to 192,000 bytes per second.

```
<e><hg><hw>byte</hw> <pr><ph>baIt</ph></pr></hg>. <la>Computers</la>.
<etym>Arbitrary, prob. influenced by <xr><x>bit</x></xr>
<ps>n.<hm>4</hm></ps>and <xr><x>bite</x> <ps>n.</ps> </xr></etym>
<s4>A group of eight consecutive bits operated on as a unit in a
computer.</s4> <qp><q><qd>1964</qd><a>Blaauw</a> &amp. <a>Brooks</a>
<bib>in</bib> <w>IBM Systems Jrnl.</w> <lc>III. 122</lc> <qt>An 8-bit
unit of information is fundamental to most of the formats <ed>of the
System/360</ed>.&es.A consecutive group of <i>n</i> such units
constitutes a field of length <i>n</i>.&es.Fixed-length fields of
length one, two, four, and eight are termed bytes, halfwords, words,
and double words respectively. </qt></q><q><qd>1964</qd> <w>IBM Jrnl.
Res. &amp. Developm.</w> <lc>VIII. 97/1</lc> <qt>When a byte of data
appears from an I/O device, the CPU is seized, dumped, used and
restored.</qt></q> <q><qd>1967</qd> <a>P. A. Stark</a> <w>Digital
Computer Programming</w> <lc>xix. 351</lc> <qt>The normal operations
```

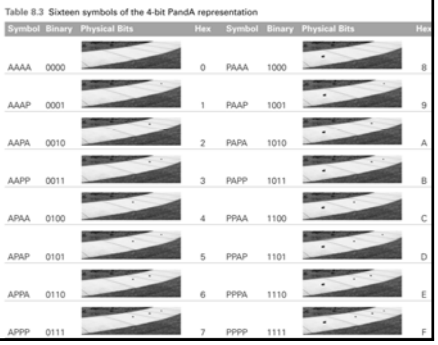


Table 8.3 Sixteen symbols of the 4-bit PandA representation

Symbol	Binary	Physical Bits	Hex	Symbol	Binary	Physical Bits	Hex
AAAA	0000		0	PAAA	1000		8
AAAP	0001		1	PAAP	1001		9
AAPA	0010		2	PAPA	1010		A
AAPP	0011		3	PAPP	1011		B
APAA	0100		4	PPAA	1100		C
APAP	0101		5	PPAP	1101		D
APPA	0110		6	PPPA	1110		E
APPP	0111		7	PPPP	1111		F

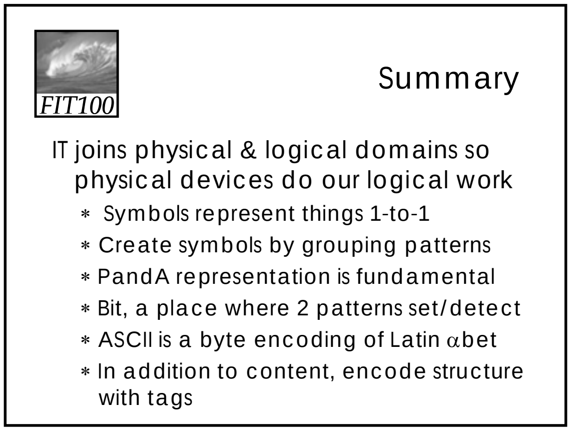
Demonstration

Course Web site:

- http://www.cs.washington.edu/education/courses/100/07au/

Address munging:

http://www.addressmunger.com



FIT100

Summary

IT joins physical & logical domains so physical devices do our logical work

- Symbols represent things 1-to-1
- Create symbols by grouping patterns
- PandA representation is fundamental
- Bit, a place where 2 patterns set/detect
- ASCII is a byte encoding of Latin αbet
- In addition to content, encode structure with tags