Announcements

- Labs come in two forms: exercises and free work; Thursday’s lab is some of each

How we represent bits, numbers, letters?

Communicating in the Blink of an Eye

Lawrence Snyder
University of Washington, Seattle

Today… Bits

- Key principle: *Information is the presence or absence of a phenomenon at given place/time*
- Turn signal is an example
  - Phenom: Flashing light
  - Present: Flashing
  - Absent: Off
  - Info: Present == intention to turn in specific direction
  - Place (side of car)
  - Time: now

A General Idea

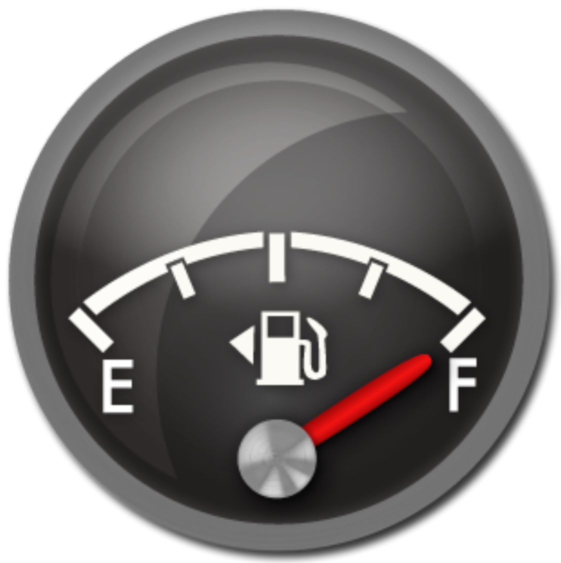
- The Presence and Absence of a phenomenon at a specific place and time abbreviated: PandA
- Phenomena: light, magnetism, charge, mass, color, current, …
- Detecting depends on phenomenon – but the result must be discrete: was it detected or not; there is no option for "sorta there"
- Place and time apply, but usually default to "obvious" values; not so important to us
- Many alternatives …

Subtleties In Interpreting Bits

- Pay attention to the meaning of the bits

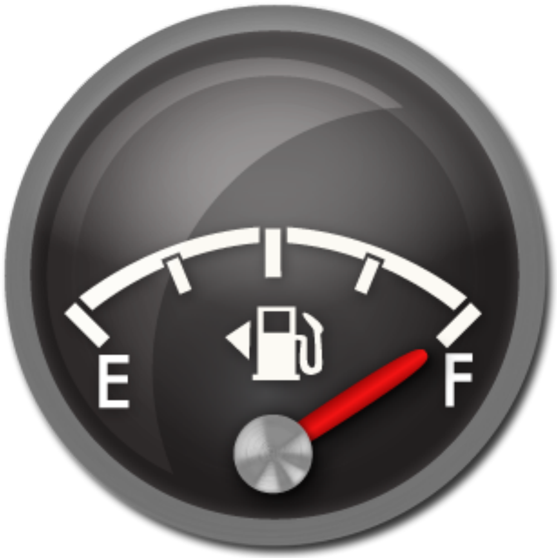
Subtleties In Interpreting Bits

- Pay attention to the meaning of the bits



Subtleties In Interpreting Bits

- Pay attention to the meaning of the bits



A Curious Story… That You Know

The Diving Bell and the Butterfly
Jean-Dominique Bauby

Asking Yes/No Questions

- A protocol for Yes/No questions
  - One blink == Yes
  - Two blinks == No
- PandA implies that this is not the fewest number of blinks … really?

Alternatives ...

- "Presence and absence" is too long, use 0, 1
- At the coffee shop ... record passersby:

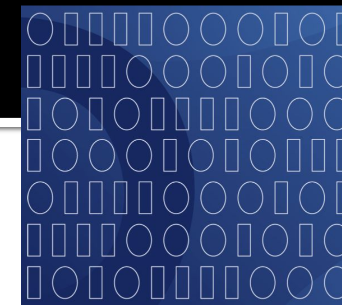
 == going right 

 == wearing purple 

  - In multi-state cases, pick one for present, all others are absent

Back 2 Bits





- PandA is a *binary representation* because it uses 2 patterns

Bit – it's a contraction for "binary digit"

Information exists even if the phenom is absent

Sherlock Holmes's *Mystery of Silver Blaze* -- a popular example where "absent" gives information … the dog didn't bark, that is the phenomenon wasn't detected

Memory -- a position in space/time capable of being set and detected in 2 patterns

Bytes

- A byte is eight bits treated as a unit
  - Adopted by IBM in 1960s
  - A standard measure ever since
  - Bytes encode the Latin alphabet using ASCII -- the American Standard Code for Information Interchange

0101 0101
0101 0111

ASCII

0100 0011
0101 0011
0101 0000

A byte has 256 configurations (symbols), good enough for usual American and Western European languages

ASCII	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0000	NU	SH	SX	EX	ET	EQ	AK	BL	BS	HT	LF	YT	FF	CR	S0	SI
0001	DL	D1	D2	D3	D4	NK	SY	EΣ	CN	EM	SB	EC	FS	GS	RS	US
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	{	|	}	~	DT
1000	80	81	82	83	IN	NL	SS	ES	HS	HJ	YS	PD	PV	RI	S2	S3
1001	DC	P1	PZ	SE	CC	MM	SP	EP	Q8	QQ	QA	CS	ST	OS	PM	AP
1010	AO	¡	¢	£	♀	¥	¦	§	¨	©	♂	«	¬	-	®	¯
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	ß
1111	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

0100 1000 | 0111 0101 | 0111 0011 | 0110 1011 | 0110 1001 | 0110 0101 | 0111 0011 | 0010 0001

UTF-8

Uniform Transformation Format for bytes (UTF-8) is universal … all characters have a place: 1,2,3,4 B

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 **한국어**를 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**?
ทำไมเขาถึงไม่พูดภาษาไทย
Neden **Türkçe** konuşamıyorlar?

UTF-8

Uniform Transformation Format for bytes (UTF-8) is universal … all characters have a place: 1,2,3,4 B

- 100,000 characters

¿sıɥʇ pɐǝɹ noʎ uɐɔ

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 **한국어**를 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?

Encoding Information

- 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
  - Tags are one form of meta-data: *meta-data* is information about information

OED Entry For Byte -- Metadata

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 &amp.
<a>Brooks <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 <w>Digital
Computer Programming</w> <lc>xix. 351</lc> <qt>The normal operations in fixed
point are done on four bytes at a time.</qt></q><q><qd>1968</qd> <w>Dataweek</w>
<lc>24 Jan. 1/1</lc> <qt>Tape reading and writing is at from 34,160 to 192,000
bytes per second.</qt></q></qp></e>

Encoding Nos. Positional Notation

- Binary numbers, like decimal numbers, use *place notation*

$$1101 = 1\times1000 + 1\times100 + 0\times10 + 1\times1$$

$$= 1\times10^3 + 1\times10^2 + 0\times10^1 + 1\times10^0$$

except that the base is 2 not 10

$$1101 = 1\times8 + 1\times4 + 0\times2 + 1\times1$$

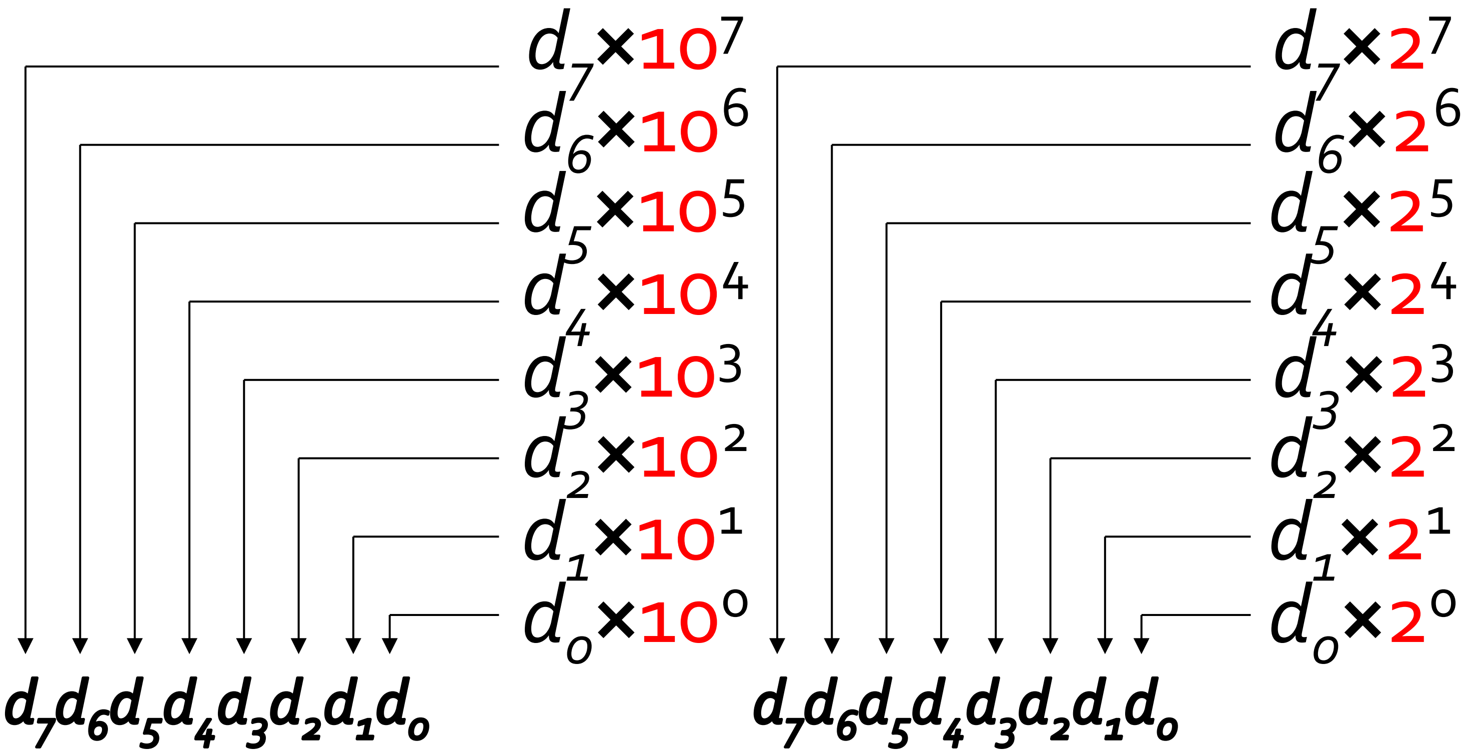
$$= 1\times2^3 + 1\times2^2 + 0\times2^1 + 1\times2^0$$

Base or radix

1101 in binary is 13 in decimal

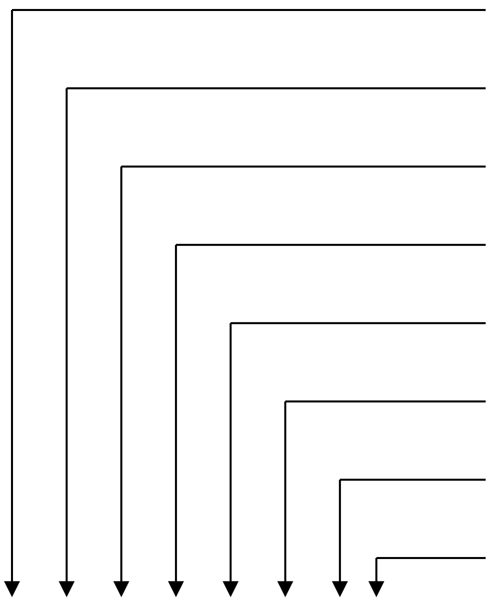
Positional Notation Logic

Binary is just like decimal except that it uses base 2 rather than base 10 …



Representing 160 in Binary

Given a binary number, add up the powers of 2 corresponding to its 1s



$$1\times2^7 = 1\times128 = 128$$
$$0\times2^6 = 0\times64 = 0$$
$$1\times2^5 = 1\times32 = 32$$
$$0\times2^4 = 0\times16 = 0$$
$$0\times2^3 = 0\times8 = 0$$
$$0\times2^2 = 0\times4 = 0$$
$$0\times2^1 = 0\times2 = 0$$
$$0\times2^0 = 0\times1 = 0$$

1 0 1 0 0 0 0 0

$$=160$$

Representing 76 In Binary

- Given a binary number, add up the powers of 2 corresponding to 1s

0×2^7	$= 0 \times 128$	$= 0$
1×2^6	$= 1 \times 64$	$= 64$
0×2^5	$= 0 \times 32$	$= 0$
0×2^4	$= 0 \times 16$	$= 0$
1×2^3	$= 1 \times 8$	$= 8$
1×2^2	$= 1 \times 4$	$= 4$
0×2^1	$= 0 \times 2$	$= 0$
0×2^0	$= 0 \times 1$	$= 0$
		$=76$

0 1 0 0 1 1 0 0

Representing Color Husky Purple

- So Husky purple is (160,76,230) which is

  1010 0000 0100 1100 1110 0110

  160 76 230

Representing Color Husky Purple

- So Husky purple is (160,76,230) which is

  1010 0000 0100 1100 1110 0110

  160 76 230

  Suppose you decide it's not "red" enough
  - **Increase the red by 16 = 1 0000**

```
  1010 0000
+    1 0000
-----------
  1011 0000
```

Adding in binary is pretty much like adding in decimal

Adding In Binary … like Decimal

- Increase by 16 more

```
   00110 000  <———  Carries

    1011 0010
+      1 0100
   ----------
    1100 0110
```

```
   1
 + 1
 ---
  10
```

The rule: When the "place sum" equals the radix or more, subtract radix & carry

Check it out online: searching binary addition *hits 19M times, and all of the p.1 hits are good explanations*

Find Binary From Decimal

What is 230? Fill in the Table:

Num Being Converted	230	230	102	38	6	6	6	2	0
Place Value	256	128	64	32	16	8	4	2	1
Subtract		102	38	6			2	0	
Binary Num	0	1	1	1	0	0	1	1	0

Find Binary From Decimal

Place number to be converted into the table; fill place value row with decimal powers of 2

Num Being Converted	230								
Place Value	256	128	64	32	16	8	4	2	1
Subtract									
Binary Num									

Find Binary From Decimal

Rule: Subtract PV from the number; a positive result gives new number and "1"; otherwise, "0"

Num Being Converted	230	230							
Place Value	256	128	64	32	16	8	4	2	1
Subtract									
Binary Num	0								

Find Binary From Decimal

Rule: Subtract PV from the number; a positive result gives new number and "1"; otherwise, "0"

Num Being Converted	230	230	102						
Place Value	256	128	64	32	16	8	4	2	1
Subtract		102							
Binary Num	0	1							

Find Binary From Decimal

Rule: Subtract PV from the number; a positive result gives new number and "1"; otherwise, "0"

Num Being Converted	230	230	102	38					
Place Value	256	128	64	32	16	8	4	2	1
Subtract		102	38						
Binary Num	0	1	1						

Find Binary From Decimal

Rule: Subtract PV from the number; a positive result gives new number and “1”; otherwise, “0”

Num Being Converted	230	230	102	38	6				
Place Value	256	128	64	32	16	8	4	2	1
Subtract		102	38	6					
Binary Num	0	1	1	1					

Find Binary From Decimal

Rule: Subtract PV from the number; a positive result gives new number and "1"; otherwise, "0"

Num Being Converted	230	230	102	38	6	6			
Place Value	256	128	64	32	16	8	4	2	1
Subtract		102	38	6					
Binary Num	0	1	1	1	0				

Find Binary From Decimal

Rule: Subtract PV from the number; a positive result gives new number and “1”; otherwise, “0”

Num Being Converted	230	230	102	38	6	6	6		
Place Value	256	128	64	32	16	8	4	2	1
Subtract		102	38	6					
Binary Num	0	1	1	1	0	0			

Find Binary From Decimal

Rule: Subtract PV from the number; a positive result gives new number and “1”; otherwise, “0”

Num Being Converted	230	230	102	38	6	6	6	2	
Place Value	256	128	64	32	16	8	4	2	1
Subtract		102	38	6			2		
Binary Num	0	1	1	1	0	0	1		

Find Binary From Decimal

Rule: Subtract PV from the number; a positive result gives new number and "1"; otherwise, "0"

Num Being Converted	230	230	102	38	6	6	6	2	0
Place Value	256	128	64	32	16	8	4	2	1
Subtract		102	38	6			2	0	
Binary Num	0	1	1	1	0	0	1	1	

Find Binary From Decimal

Rule: Subtract PV from the number; a positive result gives new number and "1"; otherwise, "0"

Num Being Converted	230	230	102	38	6	6	6	2	0
Place Value	256	128	64	32	16	8	4	2	1
Subtract		102	38	6			2	0	
Binary Num	0	1	1	1	0	0	1	1	0

Read off the result: 0 1110 0110

Final Fact: Bits Are IT

- We ALL KNOW computers represent data by binary numbers
- NOT QUITE TRUE
- Computers represent information by bits
  - ASCII, numbers (yes, in binary), metadata + computer instructions, color, sound, video, etc.
- Fundamental Fact –
  - Bits can represent ALL information
  - Bits have no inherent meaning … you don't know what 1100 0100 1010 1110 means … it could be anything