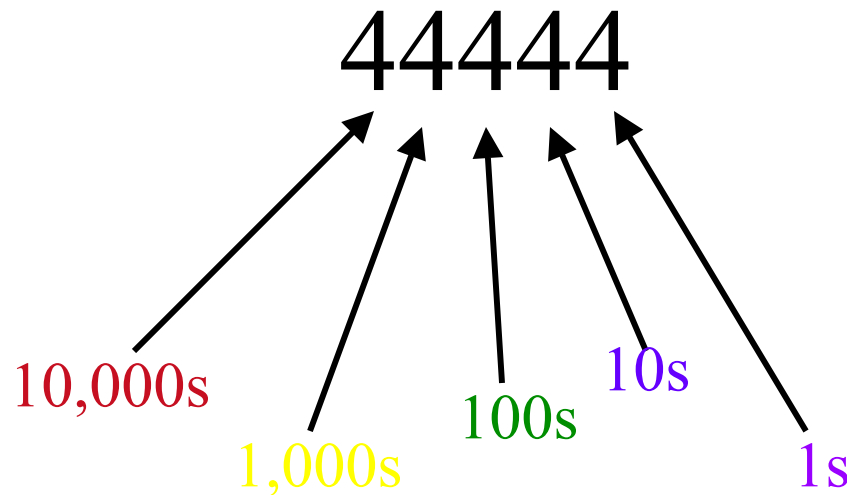


Data Representation

Numbers Are Represented in Bases



- Decimal System uses 10 digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- Base 10
- **Place-value** number system: position of a digit interpreted to give the value

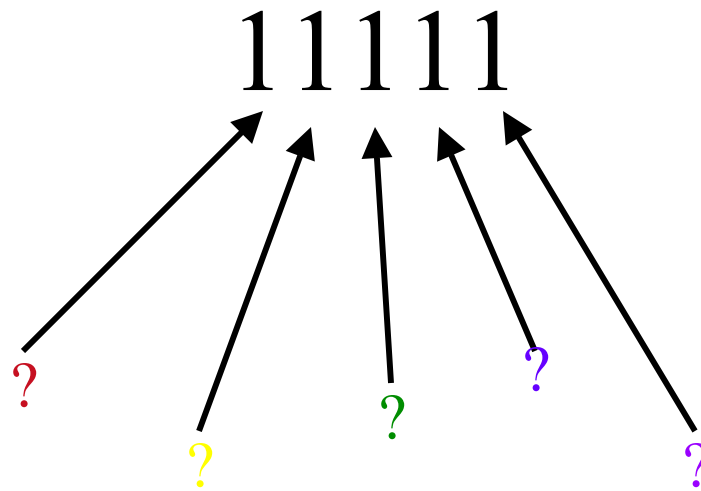
Decimal System

- $44,444 = 4 \times 10,000 + 4 \times 1,000 + 4 \times 100 + 4 \times 10 + 4 \times 1$
 $= 4 \times 10^4 + 4 \times 10^3 + 4 \times 10^2 + 4 \times 10^1 + 4 \times 10^0$
- 1 decimal digit produces ? distinct values
- 2 decimal digits produce ? distinct values
- 3 decimal digits produce ? distinct values
- n decimal digits produce ? distinct values

Decimal System

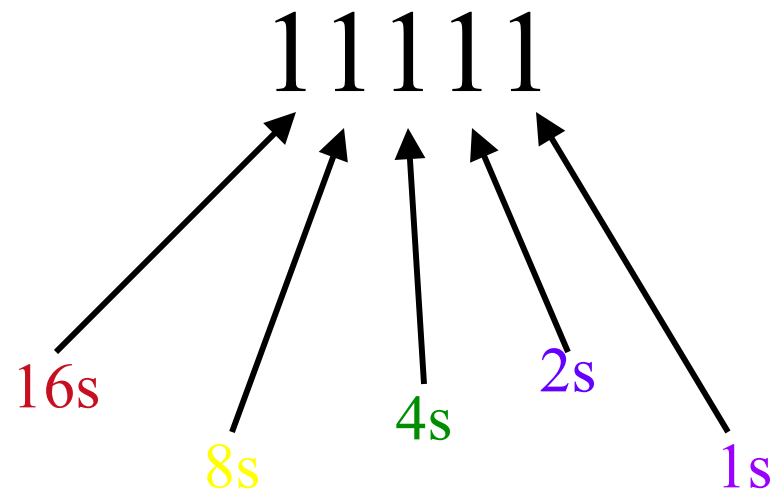
- $44,444 = 4 \times 10,000 + 4 \times 1,000 + 4 \times 100 + 4 \times 10 + 4 \times 1$
 $= 4 \times 10^4 + 4 \times 10^3 + 4 \times 10^2 + 4 \times 10^1 + 4 \times 10^0$
- 1 decimal digit produces 10 distinct values
- 2 decimal digits produce 100 distinct values
- 3 decimal digits produce 1000 distinct values
- n decimal digits produce 10^n distinct values

Binary System



- Decimal System uses 2 digits: 0, 1
- Base 2

Binary System



- Decimal System uses 2 digits: 0, 1
- Base 2

Binary System

- $11,111_2 = 1 \times 16 + 1 \times 8 + 1 \times 4 + 1 \times 2 + 1 \times 1$
 $= 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$
- 1 decimal digit produces ? distinct values
- 2 decimal digits produce ? distinct values
- 3 decimal digits produce ? distinct values
- n decimal digits produce ? distinct values

Binary System

- $11,111_2 = 1 \times 16 + 1 \times 8 + 1 \times 4 + 1 \times 2 + 1 \times 1$
 $= 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$
- 1 decimal digit produces 2 distinct values
- 2 decimal digits produce 4 distinct values
- 3 decimal digits produce 8 distinct values
- n decimal digits produce 2^n distinct values

Binary Numbers

- When counting in decimal, use the 9 “symbols” (0 – 9) for the first 10 numbers. After 9, use 10 (two symbols, the smallest non-zero symbol in the newly created second spot from the left)
- When counting in binary, use the 2 “symbols” (0, 1) for the first 2 numbers. After 1, use 10. So, 1 in the binary world is like 9 in the decimal world.

Binary Numbers

- Counting in Decimal:
- Counting in Binary

Binary Numbers

- Counting in Decimal:

0, 1, 2, ..., 9

10, 11, 12, ..., 19,

20, ..., 29, ..., 99,

100, ..., 999,

1000, ..., 9999,

10000, ..., 99999,

...

- Counting in Binary

0, 1,

10, 11,

100, 101, 110, 111,

1000, 1001, 1010, 1011,

1100, 1101, 1110, 1111,

10000, ..., 11111,

...

Decimal vs. Binary

Decimal	0	1	2	3	4	5	6	7
Binary	0	1	10	11	100	101	110	111

3 binary digits produce 8 distinct values: 0, ..., 7 (8 including 0). So the largest number $7_{10} = 111_2 = 2^3 - 1 = 8 - 1$

Binary to Decimal Conversion

- $1011_2 = ?_{10}$

Binary to Decimal Conversion

- $1011_2 = ?_{10}$
- $1011_2 = 1 \times 2_{10}^3 + 0 \times 2_{10}^2 + 1 \times 2_{10}^1 + 1 \times 2_{10}^0$
 $= 8_{10} + 0_{10} + 2_{10} + 1_{10} = 11_{10}$

Decimal to Binary Conversion

- How to find the decimal digits that make up 53624_{10} ?

Decimal to Binary Conversion

- How to find the decimal digits that make up 53624_{10} ?

$$53624 / 10 = 5362 + 4/10$$

$$5362 / 10 = 526 + 2/10$$

$$526 / 10 = 52 + 6/10$$

$$52 / 10 = 5 + 2/10$$

$$5 / 10 = 0 + 5/10$$

Decimal to Binary Conversion

- How to find the binary digits that make up 24_{10} ?

Decimal to Binary Conversion

- How to find the decimal digits that make up 24_{10} ?

$$24 / 2 = 12 + 0/2$$

$$12 / 2 = 6 + 0/2$$

$$6 / 2 = 3 + 0/2$$

$$3 / 2 = 1 + 1/2$$

$$1 / 2 = 0 + 1/2$$

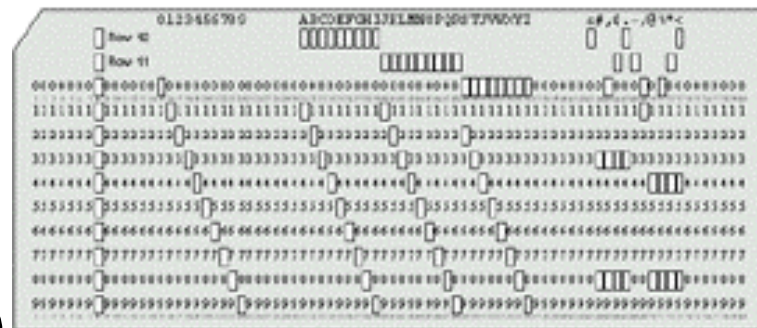
- $24_{10} = 11000_2$

Binary Addition

$$\begin{array}{r} \bullet \bullet \quad \text{carry (in 2s)} \\ 1110 \\ + 1011 \\ \hline 11001 \end{array}$$

Binary Numbers in Computing

- Easy to make fast, reliable, small devices that have only **2 states**
- **1/0** represented by
 - **hole/no hole** in punched card
 - **hi/low voltage** (memory chips)
 - **light bounces off/light doesn't bounce off** (CDs/DVDs)
 - **magnetic charge present/no magnetic charge** (disks)



Measuring Data

We can group number of binary digits and refer to the group sizes by special names:

- 1 **bit**(b) = 2^1 = represents 2 different values
- 1 **byte**(B) = 8 bits = 2^8 = 256 values
- 1 **kilobyte**(KB) = 1024 bytes = 2^{10} bytes
- 1 **megabyte**(MB) = 1024 KB = 2^{20} bytes
- 1 **gigabyte**(GB) = 1024 MB = 2^{30} bytes
- 1 **terabyte**(TB) = 1024 GB = 2^{40} bytes

Other Number Systems

- Hexadecimal (base 16)
 - digits: 0, 1, 2, ..., 9, A(10), B(11), ..., F(15)
 - example: $5B7 = 5 \times 16^2 + 11 \times 16 + 7 = 1463_{10}$
- Octal
 - digits: 0, 1, ..., 7
 - example: 239 an invalid octal number

Hexadecimal Addition

$$\begin{array}{r} \cdot \quad \cdot \quad \text{carry (in 16s)} \\ \text{D F 6 D} \\ + 2 4 6 \text{C} \\ \hline \end{array}$$

$D + 2 + 1 = 13 + 3 = 16 = 16 + 0$

1 0 3 D 9

$F + 4 = 15 + 4 = 19 = 16 + 3$

$D + C = 13 + 12 = 25 = 16 + 9$

$6 + 6 + 1 = 13 = D$

Representing Text

- Each **letter** is encoded using 1 **byte**
- ASCII (*American Standard Code for Information Interchange*) table

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	00 0000 NUL ☐	01 0001 SOH ▮	02 0010 STX ⌞	03 0011 ETX ⌟	04 0100 EOT ↵	05 0101 ENQ ⊠	06 0110 ACK ✓	07 0111 BEL ⚙	08 1000 BS ↩	09 1001 HT ➤	10 1010 LF ≡	11 1011 VT ∇	12 1100 FF ⇩	13 1101 CR ⏪	14 1110 SO ⊗	15 1111 SI ⊙
1	16 0001 DLE ☐	17 0011 DC1 ⌚	18 0010 DC2 ⌚	19 0011 DC3 ⌚	20 0101 DC4 ⌚	21 0110 NAK ↵	22 0111 SYN ⌞	23 1001 ETB ⌟	24 1010 CAN ⌘	25 1011 EM ↓	26 1101 SUB ?	27 1110 ESC ⊖	28 1100 FS ☐	29 1101 GS ☐	30 1110 RS ☐	31 1111 US ☐
2	32 0010 SP 	33 0011 !	34 0010 "	35 0011 #	36 0010 \$	37 0011 %	38 0010 &	39 0011 '	40 0100 (41 0101)	42 0110 *	43 0111 +	44 1000 ,	45 1001 -	46 1010 . /	47 1011
3	48 0011 0	49 0010 1	50 0011 2	51 0010 3	52 0011 4	53 0010 5	54 0011 6	55 0010 7	56 0011 8	57 0010 9	58 0011 :	59 0010 ;	60 0011 <	61 0010 =	62 0011 >	63 0010 ?
4	64 0100 @	65 0101 A	66 0110 B	67 0111 C	68 0100 D	69 0101 E	70 0110 F	71 0111 G	72 0100 H	73 0101 I	74 0110 J	75 0111 K	76 0100 L	77 0101 M	78 0110 N	79 0111 O
5	80 0101 P	81 0110 Q	82 0111 R	83 0100 S	84 0101 T	85 0110 U	86 0111 V	87 0100 W	88 0101 X	89 0110 Y	90 0111 Z	91 0100 [92 0101 \	93 0110]	94 0111 ^	95 0100 _
6	96 0110 ,	97 0111 a	98 0100 b	99 0101 c	100 0110 d	101 0111 e	102 0100 f	103 0101 g	104 0110 h	105 0111 i	106 0100 j	107 0101 k	108 0110 l	109 0111 m	110 0100 n	111 0101 o
7	112 0111 p	113 0110 q	114 0111 r	115 0110 s	116 0111 t	117 0100 u	118 0101 v	119 0110 w	120 0111 x	121 0110 y	122 0111 z	123 0100 {	124 0101 	125 0110 }	126 0111 ~	127 0110 DEL

Space: " "

Representing Text

“M

A

R

”C

1st byte	2nd byte	3rd byte	4th byte
77	97	114	99
01001101	01100001	01110010	01100011