CS 146: Intro to Web Programming and Project Development

Instructor: Iraklis Tsekourakis

Email: itsekour@stevens.edu

Number Systems

Objectives

- Understand the different representations of numbers
- Learn how to convert a number from one system to another
- Learn how to do basic math operations on binary numbers

Common Number Systems

System	Base	Symbols	Used by humans?	Used in computers?
Decimal	10	0, 1, … 9	Yes	No
Binary	2	0, 1	No	Yes
Octal	8	0, 1, … 7	No	No
Hexa-decimal	16	0, 1, … 9, A, B, … F	No	No

Why use octal or hexadecimal?

- Octal and hex use the human advantage that they can work with lots of symbols
- It is still easily convertible back and forth to the between binary
    - $16=2^4$, $8=2^3$

Counting

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

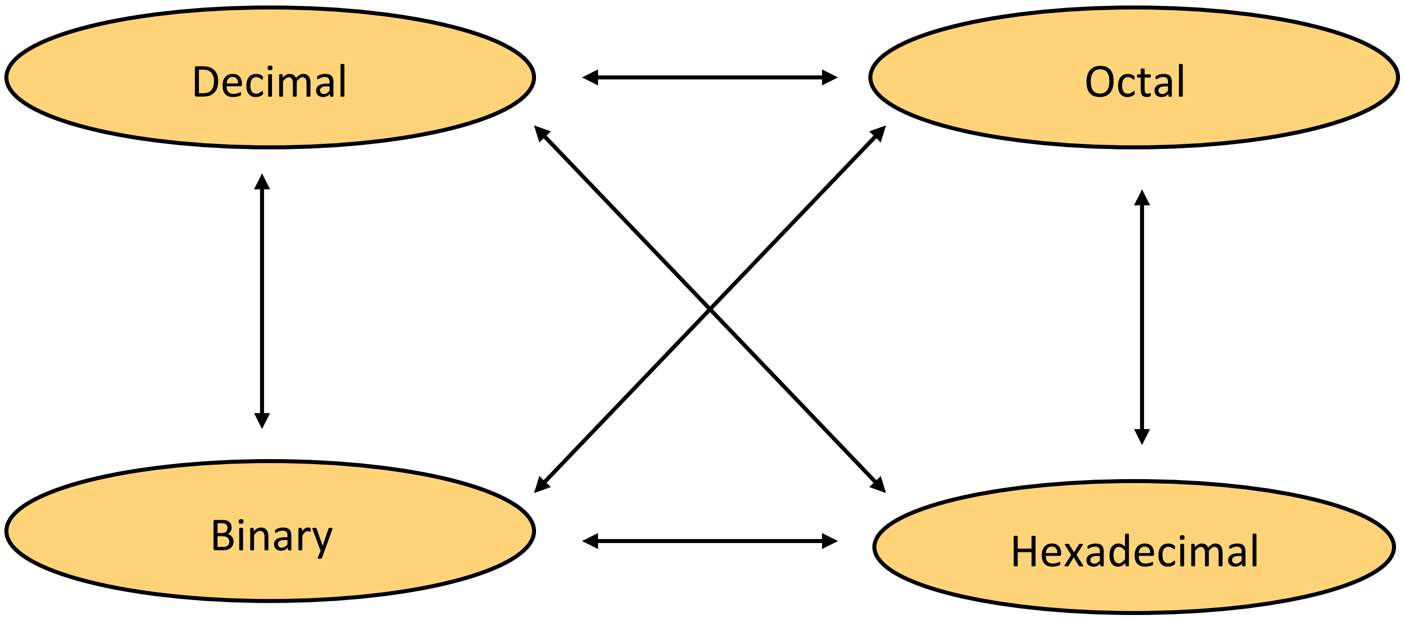
Counting II

Decimal	Binary	Octal	Hexa-decimal
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Counting III

Decimal	Binary	Octal	Hexa-decimal
16	10000	20	10
17	10001	21	11
18	10010	22	12
19	10011	23	13
20	10100	24	14
21	10101	25	15
22	10110	26	16
23	10111	27	17

Conversion Among Bases



Quick Example

$$25_{10} = 11001_2 = 31_8 = 19_{16}$$

Base

Binary to Decimal

- Technique
  – Multiply each bit by 2^n, where n is the "weight" of the bit
  – The weight is the position of the bit, starting from 0 on the right
  – Add the results
- Ex: 101011_2

Binary to Decimal

$$
\begin{array}{llrcr}
101011_2 \Rightarrow & 1 \times 2^0 & = & 1 \\
& 1 \times 2^1 & = & 2 \\
& 0 \times 2^2 & = & 0 \\
& 1 \times 2^3 & = & 8 \\
& 0 \times 2^4 & = & 0 \\
& 1 \times 2^5 & = & 32 \\
& & & 43_{10}
\end{array}
$$

Octal to Decimal

- Technique
  - Multiply each bit by 8^n, where *n* is the "weight" of the bit
  - The weight is the position of the bit, starting from 0 on the right
  - Add the results
- Ex: 724_8

Octal to Decimal

$$
\begin{array}{llrr}
724_8 \Rightarrow & 4 \times 8^0 = & 4 \\
& 2 \times 8^1 = & 16 \\
& 7 \times 8^2 = & 448 \\
& & 468_{10}
\end{array}
$$

Hexadecimal to Decimal

- Technique
  - Multiply each bit by 16^n, where n is the "weight" of the bit
  - The weight is the position of the bit, starting from 0 on the right
  - Add the results
- Ex: $\texttt{ABC}_{16}$

Hexadecimal to Decimal

ABC_{16} =>

$$
\begin{array}{lcl}
C \times 16^0 = 12 \times 1 = & 12 \\
B \times 16^1 = 11 \times 16 = & 176 \\
A \times 16^2 = 10 \times 256 = & 2560 \\
 & 2748_{10}
\end{array}
$$

Decimal to Binary

- Technique
  - Divide by two, keep track of the remainder
  - First remainder is bit 0 (LSB, least-significant bit)
  - Second remainder is bit 1
  - Etc.
- Ex: $125_{10} = ?_2$

Decimal to Binary

2	125	
2	62	1
2	31	0
2	15	1
2	7	1
2	3	1
2	1	1
	0	1

$125_{10} = 1111101_2$

Octal to Binary

- Technique
  - Convert each octal digit to a 3-bit equivalent binary representation
- Ex: $705_8 = ?_2$

Octal to Binary

7	0	5
↓	↓	↓
111	000	101

$705_8 = 111000101_2$

Hexadecimal to Binary

- Technique
  - Convert each hexadecimal digit to a 4-bit equivalent binary representation
- Ex: $\texttt{10AF}_{16} = \texttt{?}_{2}$

Hexadecimal to Binary

1	0	A	F
↓	↓	↓	↓
0001	0000	1010	1111

$10AF_{16} = 0001000010101111_2$

Decimal to Octal

- Technique
  - Divide by 8
  - Keep track of the remainder
- Ex: $1234_{10} = ?_8$

Decimal to Octal

8	1234	
8	154	2
8	19	2
8	2	3
	0	2

$1234_{10} = 2322_8$

Decimal to Hexadecimal

- Technique
  - Divide by 16
  - Keep track of the remainder
- Ex: $1234_{10} = ?_{16}$

Decimal to Hexadecimal

```
16 | 1234
16 |   77     2
16 |    4    13 = D
        0     4
```

$1234_{10} = 4D2_{16}$

Binary to Octal

- Technique
  - Group bits in threes, starting on right
  - Convert to octal digits
- Ex: $1011010111_2 = ?_8$

Binary to Octal

1 011 010 111

1 3 2 7

$1011010111_2 = 1327_8$

Binary to Hexadecimal

- Technique
  - Group bits in fours, starting on right
  - Convert to hexadecimal digits
- Ex: $1010111011_2 = ?_{16}$

```
10  1011  1011
↓     ↓     ↓
2     B     B
```

$1010111011_2 = 2BB_{16}$

Octal to Hexadecimal

- Technique
  - Use binary as an intermediary
- Ex: $1076_8 = ?_{16}$

1 0 7 6

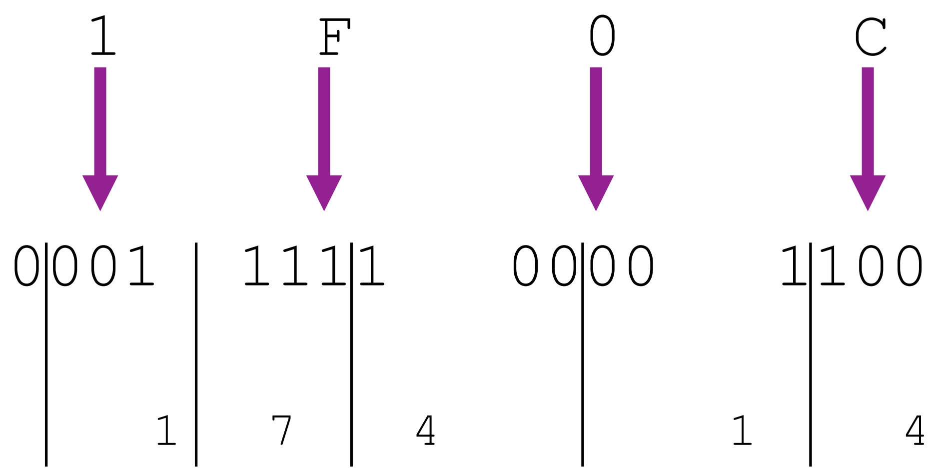
001 0|00 11|1 110

2 | 3 | E

$1076_8 = 23E_{16}$

Hexadecimal to Octal

- Technique
  - Use binary as an intermediary
- Ex: $1F0C_{16} = ?_8$



$1F0C_{16} = 17414_8$

Grouping of conversion methods

- Binary/Octal/Hex to Decimal: Bitwise multiplication with base ($2^n/8^n/16^n$)
- Decimal to Binary/Octal/Hex: Divide by base(2/8/16) and keep track of remainders
- Oct/Hex <-> Binary: Grouping into 3/4 bits (Works both ways)
- Octal <-> Hex: Use binary as intermediate step (both ways)

Exercise: Conversions

Decimal	Binary	Octal	Hexa-decimal
33			
	1110101		
		703	
			1AF

Answers

Decimal	Binary	Octal	Hexa-decimal
33	100001	41	21
117	1110101	165	75
451	111000011	703	1C3
431	110101111	657	1AF

Common Powers

Base 10

Power	Preface	Symbol	Value
10^{-12}	pico	p	.000000000001
10^{-9}	nano	n	.000000001
10^{-6}	micro	µ	.000001
10^{-3}	milli	m	.001
10^{3}	kilo	k	1000
10^{6}	mega	M	1000000
10^{9}	giga	G	1000000000
10^{12}	tera	T	1000000000000

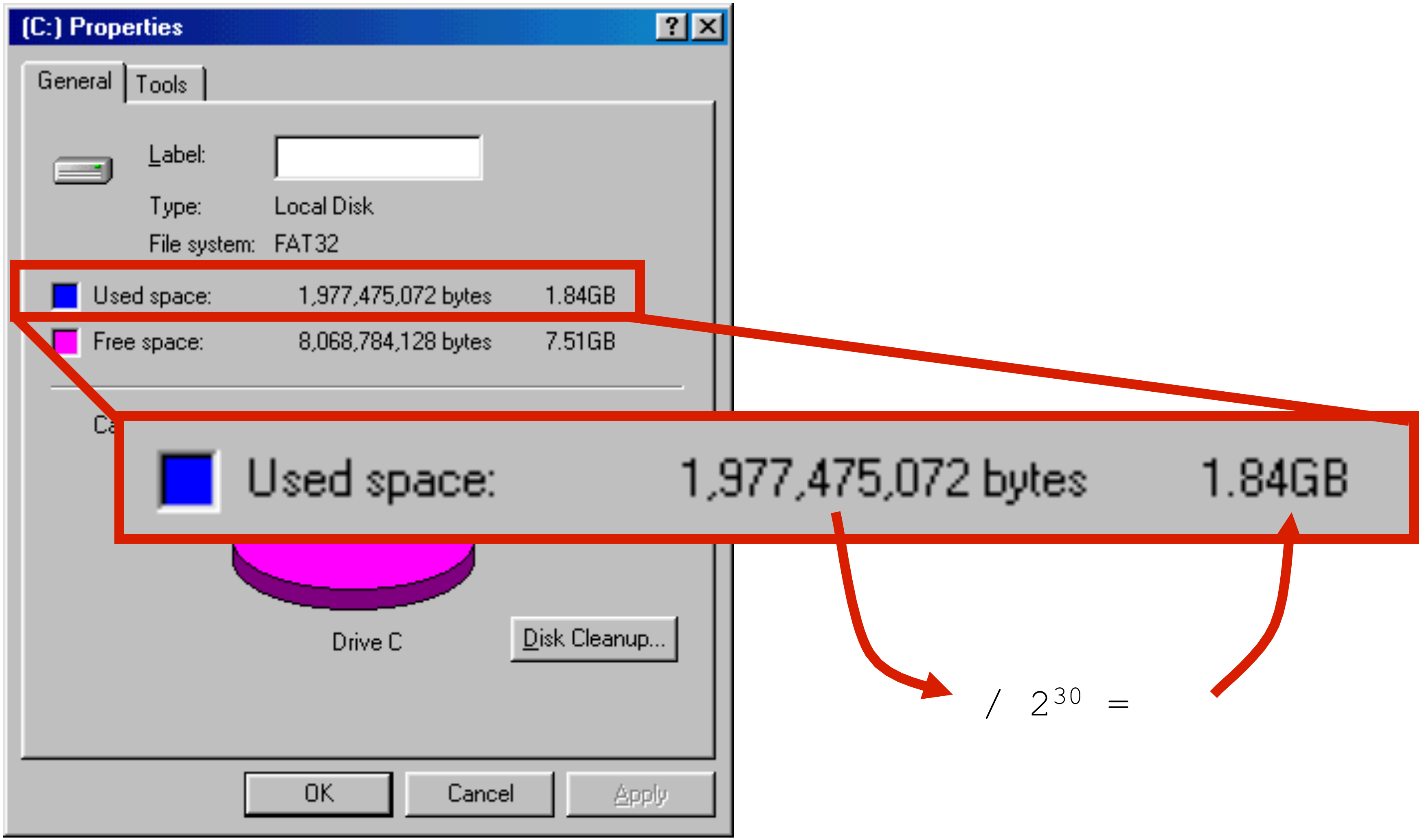
Common Powers

Base 2

Power	Preface	Symbol	Value
2^{10}	kilo	k	1024
2^{20}	mega	M	1048576
2^{30}	Giga	G	1073741824

- In Computing, particularly w.r.t. memory and speed, the base-2 interpretation generally applies

For example..



(C:) Properties
General
Tools
Label:
Type: Local Disk
File system: FAT32
Used space: 1,977,475,072 bytes 1.84GB
Free space: 8,068,784,128 bytes 7.51GB
Drive C
Disk Cleanup...
OK
Cancel
Apply
Used space: 1,977,475,072 bytes 1.84GB
/ 2^30 =

Binary Addition

- Two *n*-bit values
  - Add individual bits
  - Propagate carries
  - E.g.,

$$
\begin{array}{r}
10101 \\
+\ \underline{11001} \\
101110
\end{array}
\qquad\qquad
\begin{array}{r}
21 \\
+\ \underline{25} \\
46
\end{array}
$$

Binary Subtraction

- Two *n*-bit values
  - Subtract individual bits
  - Borrow from the left if we have a 0 on the top row
  - E.g.,

```
   11001       26
 - 10101     - 21
 -------     ----
   00100        5
```

Binary Multiplication

- Binary, two *n*-bit values
  - As with decimal values
  - E.g.,

```
       1110
     x 1011
   --------
       1110
      1110
     0000
    1110
   --------
   10011010
```

Binary Division

- Binary, two *n*-bit values
  - As with decimal values
  - E.g.,

101010 / 000110 = 000111

```
              1 1 1        = 7₁₀
        ___________
000110 ) 1 0 1 0 1 0       = 42₁₀
        - 1 1 0            = 6₁₀
         -------
          1 0 0 1
         - 1 1 0
         --------
              1 1 0
             - 1 1 0
             -------
                  0
```

Fractions

- Binary to decimal

```
10.1011 =>    1 x 2^-4 = 0.0625
                    1 x 2^-3 = 0.125
                    0 x 2^-2 = 0.0
                    1 x 2^-1 = 0.5
                    0 x 2^0  = 0.0
                    1 x 2^1  = 2.0
                               2.6875
```

$10.1011 \Rightarrow$
$1 \times 2^{-4} = 0.0625$
$1 \times 2^{-3} = 0.125$
$0 \times 2^{-2} = 0.0$
$1 \times 2^{-1} = 0.5$
$0 \times 2^{0} = 0.0$
$1 \times 2^{1} = 2.0$
2.6875

Fractions

- Decimal to binary

3.14579

$$
\begin{array}{r}
.14579 \\
\times \quad\quad\quad 2 \\
\boxed{0}.29158 \\
\times \quad\quad\quad 2 \\
\boxed{0}.58316 \\
\times \quad\quad\quad 2 \\
\boxed{1}.16632 \\
\times \quad\quad\quad 2 \\
\boxed{0}.33264 \\
\times \quad\quad\quad 2 \\
\boxed{0}.66528 \\
\times \quad\quad\quad 2 \\
\boxed{1}.33056
\end{array}
$$

etc.

11.001001...