Announcement

- C Workshop
  - Time: 7:00 – 7:50 p.m.
  - Dates: Sunday, Sep. 14 thru Wed. Sep. 17
  - Place: MECC 127
  - Topic: String operations in C
  - Required (need to attend only one)

Lecture 4

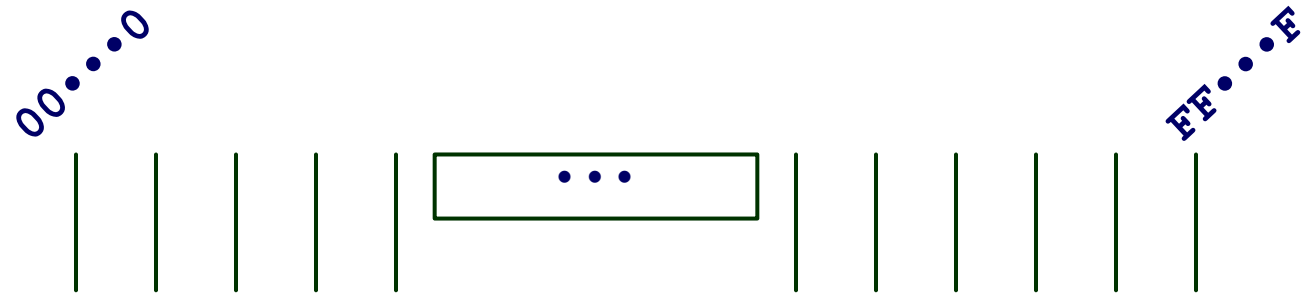
Data Representations I

CPSC 275
Introduction to Computer Systems

Data Representations in C

C Data Type	Typical 32-bit	Typical 64-bit	x86-64
`char`	1	1	1
`short`	2	2	2
`int`	4	4	4
`long`	4	8	8
`float`	4	4	4
`double`	8	8	8
pointer	4	8	8

Byte-Oriented Memory Organization



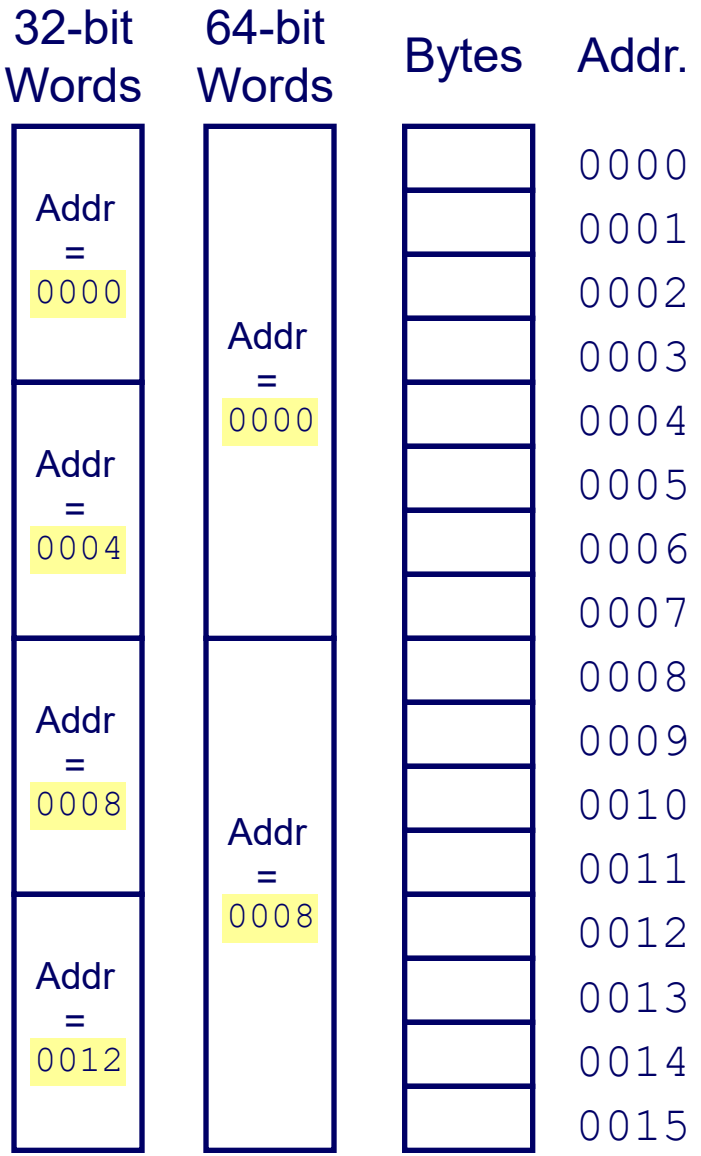
- Programs refer to *virtual addresses*
  - Conceptually very large array of bytes
  - Actually implemented with hierarchy of different memory types
  - System provides address space private to particular *process*
    - Program being executed
    - Process can access its own data, but not that of others

Machine Words

- Machine has *word* size
  - Nominal size of integer-valued data
    - Including addresses
  - Most older machines use 32 bits (4 bytes) words
    - Limits addresses to 4GB
    - Becoming too small for memory-intensive applications
  - Most current machines use 64 bits (8 bytes) words
    - Potential address space ≈ 1.8 X 10^{19} bytes
    - x86-64 machines support 48-bit addresses: 256 Terabytes

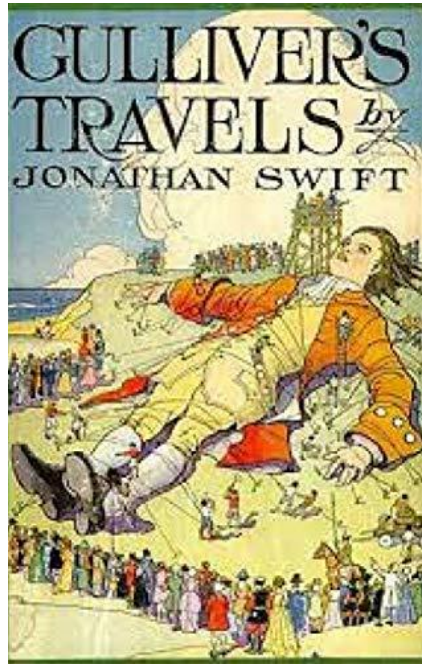
Word-Oriented Memory Organization

- Addresses specify byte locations
  - Address of first byte in word
  - Addresses of successive words differ by 4 (32-bit) or 8 (64-bit)



Byte Ordering

- How should bytes within a multi-byte word be ordered in memory?
- Conventions
  - Big Endian: RISC, Internet
    - Most significant byte has the smallest memory address
  - Little Endian: Intel x86, Apple M1
    - Least significant byte has the smallest memory address



A division in the land of Lilliput:
"Is it better to break the shell of a boiled egg from the big end or from the little end?"

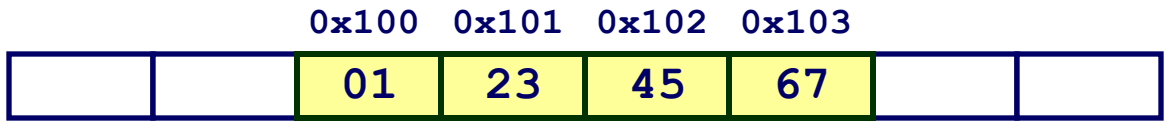
Byte Ordering Example

- Variable **x** has 4-byte representation

  0x01234567

- Address given to **x** is 0x100

Big Endian

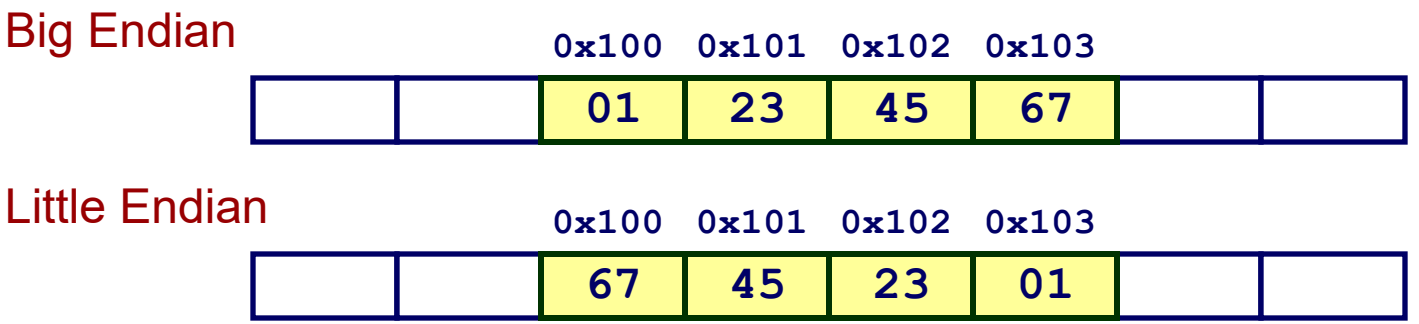


Byte Ordering Example

- Variable **x** has 4-byte representation

  0x01234567

- Address given to **x** is 0x100



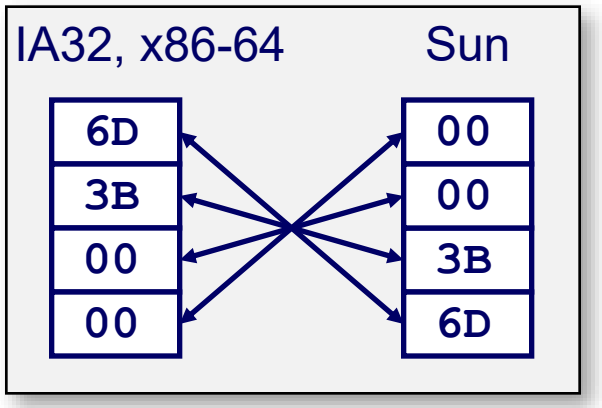
Representing Integers

Decimal: 15213

Binary: 0011 1011 0110 1101

Hex: 3 B 6 D

int A = 15213;



long int C = 15213;