

Introduction to computer systems

Lecture B01



Lecture notes section B01

2002-01-11

CSEI 303 Part B lecture notes

1

In this lecture

- Outline of course
- Bits
 - what a bit is and what it does
- Memory
 - how bits are used to store information

2002-01-11

CSEI 303 Part B lecture notes

2

Outline

- Bits and memory (1 lecture)
- Numbers (4 lectures)
 - binary
 - unsigned
 - signed
 - floating point
- Bit manipulation (1 lecture)
 - shifting
 - masking

2002-01-11

CSEI 303 Part B lecture notes

3

Outline

- Characters and strings (1 lecture)
 - ASCII
 - command-line arguments
- Compilers (1 lecture)
 - code generation
 - how compilers work
 - comparison with interpreters

2002-01-11

CSEI 303 Part B lecture notes

4

Outline

- Assembly language (2 lectures)
 - MIPS
 - SPIM simulator
- From C to MIPS (4 lectures)
 - control and data constructs
- Analysis (1 lecture)
- Computer hardware (1 lecture)

2002-01-11

CSEI 303 Part B lecture notes

5

Why?

- To write good C, need to know how C works
 - how are pointers and arrays related?

```
int foo(char *x)
{
  ... *(x+i) ...
}

=

int foo(char x[])
{
  ... x[i] ...
}
```

2002-01-11

CSEI 303 Part B lecture notes

6

Why?

- To write good C, need to know how C works
 - why do you use * and & to pass function parameters by reference?

```
void func(int *ref) {  
    ... *ref ...  
}  
  
int main() {  
    int x;  
    ... func(&x) ...  
}
```

2002-01-11

CSE1303 Part B lecture notes

7

Why?

- To write good C, need to know how C works
 - how much time does it take to call a function?

```
int main() {  
    ...  
    ... something ...  
    ...  
}
```



```
void func() {  
    something  
}  
  
int main() {  
    ...  
    ... func() ...  
    ...  
}
```

2002-01-11

CSE1303 Part B lecture notes

8

Why?

- To write good C, need to know how C works
 - how does a recursive function keep track of its local variables and parameters?

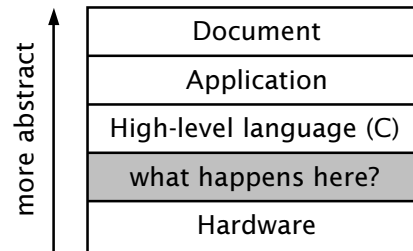
```
int factorial(int n)  
{  
    int result = 1;  
    if (n > 1)  
        result = n * factorial(n - 1);  
    return result;  
}
```

2002-01-11

CSE1303 Part B lecture notes

9

Organization hierarchy



2002-01-11

CSE1303 Part B lecture notes

10

What we'll cover

- Things you need to know as a programmer
 - efficiency
 - complexity
 - memory organization
 - representing data structures

2002-01-11

CSE1303 Part B lecture notes

11

What we won't cover

- Hardware implementation
 - digital logic and low-level architecture
 - for this, do CSE1101 and CSE1102
- Advanced computer architecture
 - interrupts
 - memory mapping and caching
 - pipelining
 - for these, do CSE2302 and CSE2324/CSE3324

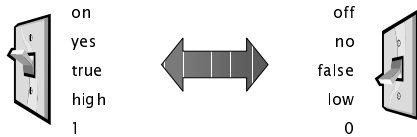
2002-01-11

CSE1303 Part B lecture notes

12

Bits

- The fundamental component of computer storage
- Can hold one single binary piece of information



2002-01-11

CSE1303 Part B lecture notes

13

Grouping bits

- One bit cannot hold much information
- Bits are grouped into larger units
 - byte = 8 bits
 - word = 16 or 32 or 64 bits
 - depending on computer
 - 32 in this course (MIPS R2000)
 - halfword, doubleword

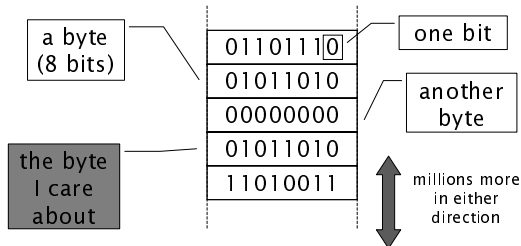
2002-01-11

CSE1303 Part B lecture notes

14

Addressing memory

- To refer to a byte, need to distinguish it from other bytes

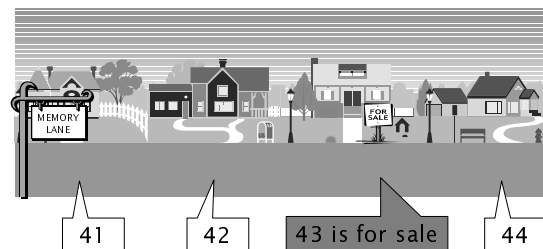


2002-01-11

CSE1303 Part B lecture notes

15

Addressing: an analogy



2002-01-11

CSE1303 Part B lecture notes

16

Addressing memory

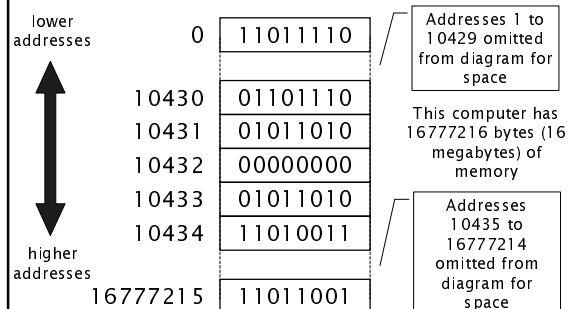
- Assign each byte a unique address
 - Address is an integer
 - no two bytes have the same address
- Addresses are sequential
 - lowest address 0
 - adjacent bytes have consecutive addresses
 - highest address depends on capacity of computer
- Address is independent of contents
 - changing the contents of a byte does not alter its address

2002-01-11

CSE1303 Part B lecture notes

17

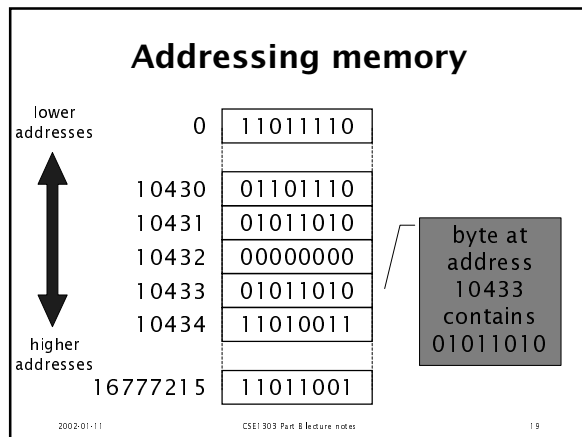
Addressing memory



2002-01-11

CSE1303 Part B lecture notes

18



- ### Looking for meaning
- What does "01011010" mean?
 - number 90?
 - ASCII character 'Z'?
 - instruction DECB for Motorola 6800 CPU?
 - just bits?
 - Depends on context
 - bits have no intrinsic meaning
 - value depends on how it is being used
 - if interpreting as number, has value 90
 - if interpreting as character, has value 'Z'
 - This is why C variables have types
- 2002-01-11 CSE1303 Part B lecture notes 20

- ### Encoding data structures
- Memory is made up of bits
 - data structures must be encoded for storage
 - Numbers (integers)
 - encoded using binary number system
 - Characters
 - encoded as numbers using ASCII code
 - Program code
 - instructions encoded as patterns of numbers and bits
 - Other structures (structs, floats, ...)
 - usually converted to numbers
- 2002-01-11 CSE1303 Part B lecture notes 21

- ### Covered in this lecture
- Outline
 - Bits
 - bytes
 - words
 - Memory
 - used to store
 - numbers
 - code
 - characters
- 2002-01-11 CSE1303 Part B lecture notes 22

- ### Going further
- Hardware implementation of memory
 - CSE1102 lecture notes
- 2002-01-11 CSE1303 Part B lecture notes 23

- ### Next time
- Numbers
 - binary representation
 - unsigned integers
- Reading:
Lecture notes section B02
- 2002-01-11 CSE1303 Part B lecture notes 24

Copyright

Copyright © 2001 Deborah Pickett.
No part of this presentation may be
duplicated without permission from
the author.