IT1007
Introduction to Programming

Python and C

Part 2: C Programming

A/Prof. Bharadwaj Veeravalli

Dept of ECE, NUS

Email: elebv@nus.edu.sg

That's me! ☺

- VEERAVALLI, Bharadwaj
- Research Areas: Computing (Cloud/HPC, Data Analytics, Embedded systems, Parallel & Distributed computing, computational biology)
- Email:
  - elebv@nus.edu.sg
  - DO NOT send codes to me
  - Office: E4 07-04
- Ext. 65258
- Hobbies – Astronomy, drawing, e-gadgets, Snooker, TT, chess

Course Components & Marks Distribution

Entire Module

- Lab/homework 20%
  - Weekly in lab sessions
- Practical exam 40%
  - A Saturday after recess week
- Final exam 40%
  - 29 Nov Afternoon

C Part

3 Labs(10%) + 1 PE(10%)

Lab 1: 23/10/2017
Lab 2: 30/10/2017
Lab 3: 06/11/2017
PE: 13/11/2017

Final Exam(20%)

Lesson Plan – Next 5/6 weeks

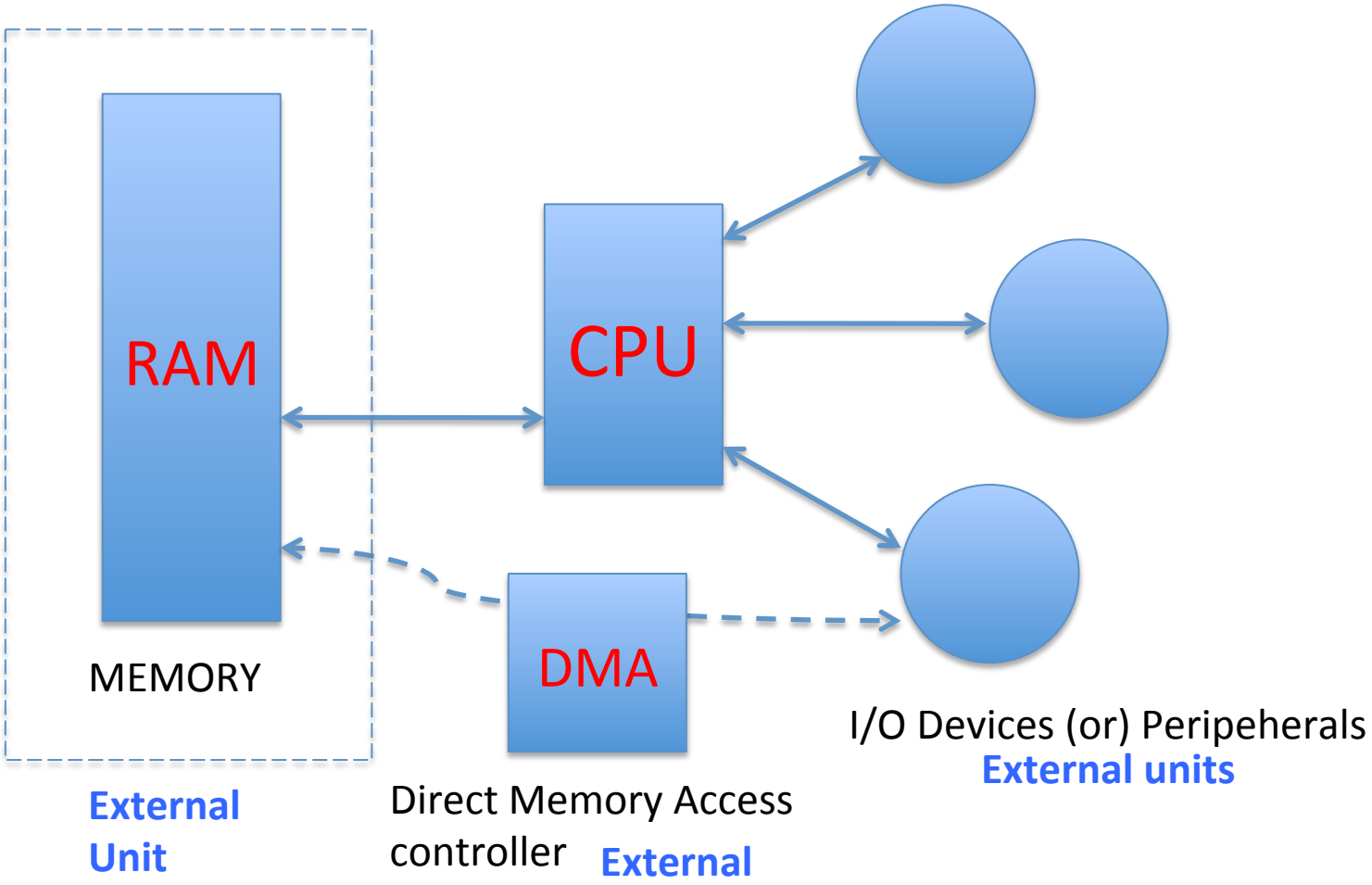
- Introduction to Computer Systems, concepts from computer systems towards Programming C, Basic Data types, expression writing, Input/Output functions;
- Decision-making, Control loops
- Array Processing, application examples
- C Functions
- C Structures
- Wrap up lecture towards EE2028! (Pointers)

Each chapter has:

- Tutorial Problems (Exam/Semi-exam type questions)
- DIY Problems!
- Discussion oriented problems
- Challenge-level problems

Let's begin some fundamentals on Computer Systems.....

Basic Computer System



Binary number system…Quick review

Binary Number system - O and 1 representation

Example:

(Decimal) $42 = 4 \times 10^1 + 2 \times 10^0$ *We know this!* ☺

(Binary) $42 = 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$

$= 32 + 8 + 2$

So the pattern will be: 101010 (*Verify the reverse!*)

What is effect of the choice of number of binary digits?

Size of the set of numbers that can be represented can increase or decrease depending on the number of digits used.

Hence, with {0,1} as the basic alphabet set for binary representation:

With just 2 digits – YY – {0,1,2,3} = {00,01,10,11} : 4 distinct #s

With just 3 digits – YYY – {0,1,…,7} = {000, 001, 010, 110,…,110, 111 } : 8 distinct #s

With just 4 digits – YYYY – {0,1,2,…,15} = (0000, 0001, …,1001, 1100, …, 1111}
16 distinct #s

. . .

With k digits: (YYYY….YY) – {0,1,…., 2^{k-1}} = (2^k) distinct numbers

Note that when we use k bits to represent a number, each number has k bits;
000. . . 000
000. . . 001
. . .
111 … 111

k-bits

Binary number system…*Quick review*

Hexa-decimal – Valid Numbers: {0,1,…,15} and they are represented as

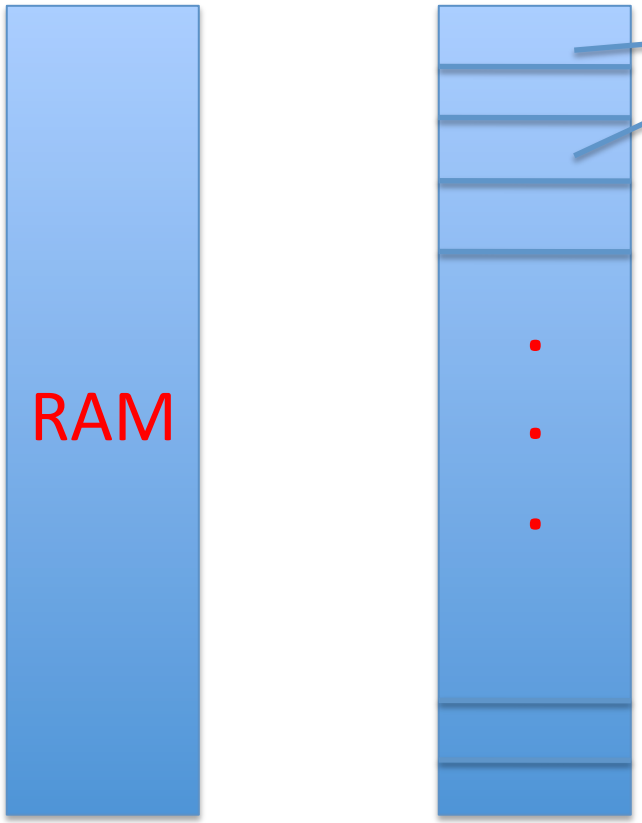
{0,1,2…,9, A, B,C,D,E,F}

Example:
(Hexa-Decimal) 42 = 2 x 16^1 + 10 x 16^0 (= 32 + 10)
So the pattern will be: 2A (*Verify the reverse!*)
We usually represent a hexadecimal number as '2A H';

Hence:

4E8 H = 1024+224 + 8 = 1256(decimal)

Computer System… Cont'd



1 Byte = 8 Bits (each slot)

0100 101**0**

Most Significant Bit(MSB)

Least Significant Bit(LSB)

Addressing:

Each slot has a unique address

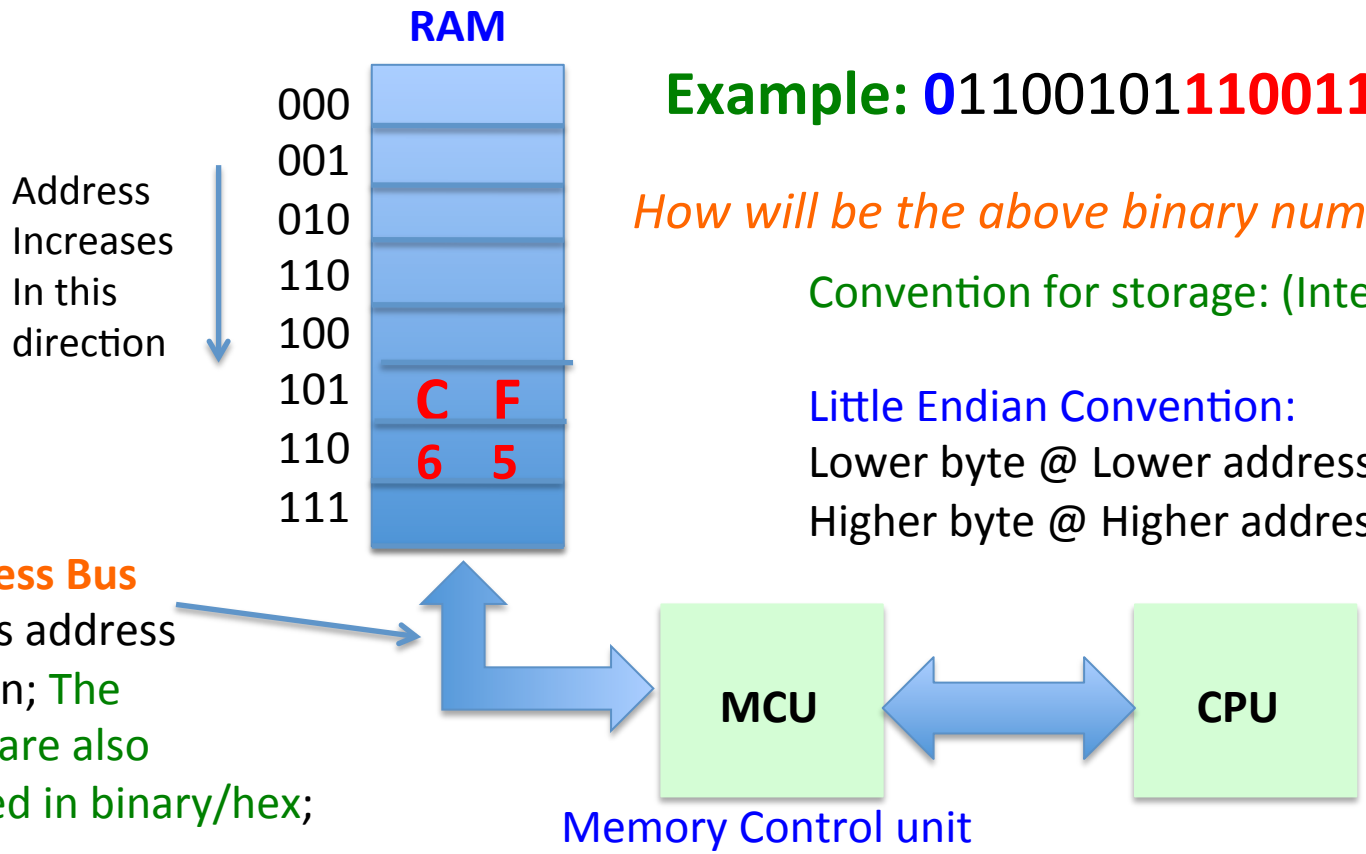
Address is represented as a binary word

RAM 8 Slots implies we need 3 bit number representing (addressing) each slot ;

RAM 16 Slots implies we need 4 bit number representing (addressing) each slot ;

0100 1010 - This pattern is represented as a hexa-decimal number 4A in the Memory for the ease of reading

Computer System… Cont'd



Example: 01100101**11001111**

How will be the above binary number stored?

Convention for storage: (Intel Chips)

Little Endian Convention:
Lower byte @ Lower address
Higher byte @ Higher address

Thus, if address bus has more wires, say 4, 8, 16,… more byte locations can be accessed by CPU. So memory space access is governed and restricted by the address bus size.

Keep this at the back of your mind always!

Therefore, we can understand that depending on the size of the number to be stored the storage requirement varies. This is a fundamental concept used during memory space allocation whenever required.

Examples of integers:

3, 25, -12, 1025, -20040, 8000000, 7565000000, 10^{15}

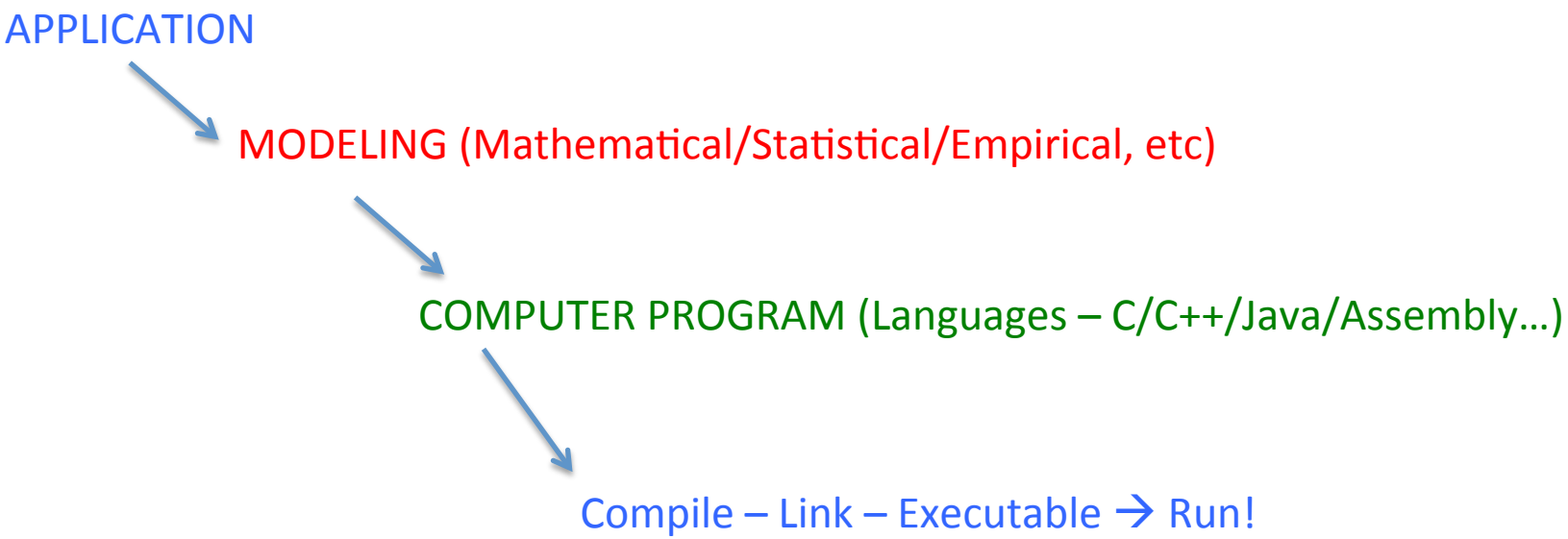
Examples of floating point numbers:

25.032, -12.101256783, 0.001025, -200.40, 0.00000002123, radius_of_an_atom

Examples of Characters / Strings:

'A', 'a', "my name is BV", "This is IT1007"

Computer System… Cont'd



How does a program "run"?

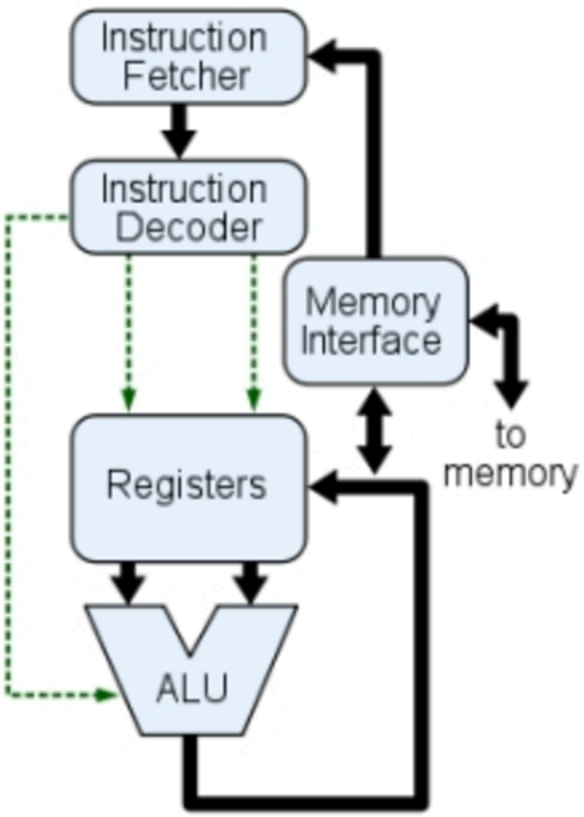
Program Execution/Running

Program must be brought to the memory (RAM) for Execution;

Each line of the program will be translated into a set of ***machine instructions*** (0's &1's) and all instructions are stored in RAM in a contiguous order;

Instructions are fetched from RAM, one at a time, and get executed in the CPU;

All such instructions will live in the memory until the entire program gets executed;



CPU Diagram

Program Execution/Running … (Cont'd)

Example

printf ("My name is BV\n"); /* typical C syntax */

10101011110001010101001010011110110 0 More than 1 byte!!

Hence,

1010 10111100 01010101 00101001 11101100

In this case, this instruction occupies 5 bytes; The last 4 most significant bits will be padded with 0s.

[DIY] Question:

- What is the hexadecimal equivalent of

  1010 10111100 01010101 00101001 11101100 ?

- How it will be stored in the RAM?