UNIVERSITY OF WATERLOO
FACULTY OF ENGINEERING
Department of Electrical & Computer Engineering

ECE 150 *Fundamentals of Programming*

Main memory

Prof. Hiren Patel, Ph.D.
Douglas Wilhelm Harder, M.Math. LEL
hdpatel@uwaterloo.ca dwharder@uwaterloo.ca

Outline

- In this lesson, we will:
  - Describe main memory
  - Define bytes and byte-addressable memory
  - Describe how addresses are stored
  - Describe how bytes are given addresses on various processors
  - Look at some images of memory

Main memory

- Main memory stores data to be used by the processor
  - It also stores instructions for programs
  - It is volatile
    - When you shut down your computer, memory is lost
- Your executing programs must load instructions and data from memory to the processor and store data back in main memory
  - This is all done for you behind the scenes
  - You must never-the-less understand this

Main memory

- How would you organize memory?
  - A book has pages, and each page has a unique page number
  - You could reference a specific word by saying:
    "The 4th word on the 7th line of the 372nd page…"

Alex Proimos from Sydney, Australia

- We need something similar, but simpler for a computer
  - Recall we have years, months, days, hours, minutes and seconds
  - Unix counts time by seconds relative to midnight January 1, 1970
  - Every piece of data will have a unique integer address

Main memory

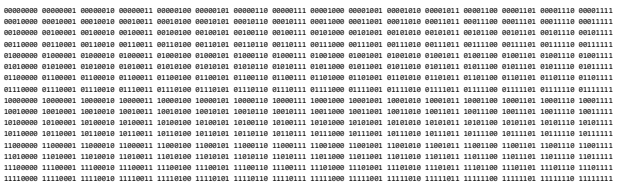
- Previously we described the computer only thinks in binary
  - Main memory is nothing more than billions or trillions of 0s and 1s
  - You could give each bit a separate address
    - You could then ask for the 32280514913rd bit
  - Problem: that's a lot of addresses
- Instead, hardware developers decided to give every 8 bits their own address
  - 8 bits is one byte
  - This is called *byte-addressable* memory

Byte-addressable memory

- In byte-addressable memory, each byte has its own unique address
  - The address is a binary number
    - On a 32-bit computer, the addresses are 32 bits long
    - On a 64-bit computer, the addresses are 64 bits long
  - A 32-bit computer can address 2^{32} bytes: 4 GiB
  - A 64-bit computer can address 2^{64} bytes: 16 EiB or ~17 million TiB
  - Not every address may correspond to an actual physical byte
    - Most computers today do not have 16 EiB of main memory
- To contrast, a hard drive is block-addressable
  - Every 4 kilobyte block of bytes has its own address

The byte

- A byte is 8 bits:
  - All possible bytes are shown here:



- Do you care? No

Byte-addressable memory

- What byte addressable means is:
  - You cannot read or write a single bit from main memory
  - If you need just a single bit, you must never-the-less refer to an entry byte in memory
  - If you want to write a single bit to main memory, you must write the entire byte in which it is contained
  - It is very convenient therefore to use multiples of bytes

The byte

- However, every byte can be represented by two hexadecimal digits:

0b00000000	0x00
0b00101100	0x2c
0b01000100	0x44
0b10001010	0x8a
0b10110101	0xb5
0b11101000	0xe8
0b11111111	0xff

0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	a
1011	b
1100	c
1101	d
1110	e
1111	f

- This is a common means of examining memory

Addresses

- Today:
  - Most processors have 64-bit addresses
  - Many microcontrollers have 32-bit addresses
  - Smaller microcontrollers may have 24-bit or 16-bit addresses
- Each of these are multiples of 8 bits
  - A 64-bit address occupies 8 bytes
  - A 32-bit address occupies 4 bytes
  - A 24-bit address occupies 3 bytes
- Thus, we may represent a
  - 64-bit address with 16 hexadecimal numbers
  - 32-bit address with 8 hexadecimal numbers
  - 24-bit address with 6 hexadecimal numbers

Addresses

- Thus, an address on your 64-bit desk- or lap-top computer may look like:

  `0x0000839d542a8b38`

- The address of the next byte would be

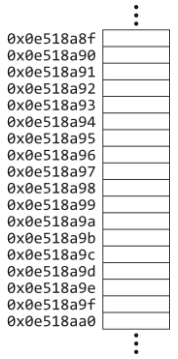
  `0x0000839d542a8b39`

- An address on a 32-bit computer might look like:

  `0x0e518a93`

Addresses

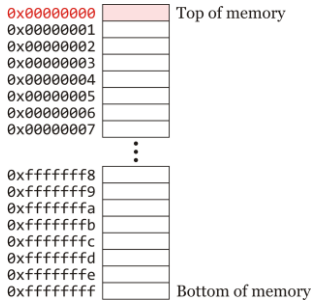
- If we are showing a block of bytes in memory, each byte will be listed with its address
  - In general, the addresses are not significant
  - What is important is what we store at those addresses



If you care: these are the addresses of bytes on a 32-bit processor

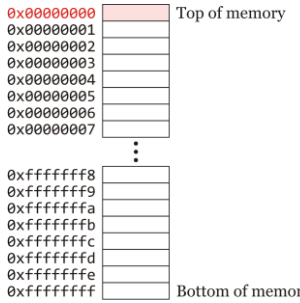
Addresses

- You can always buy as much memory as the processor can address
  - If you buy 4 GiB of memory for a 32-bit computer, every address is associated with a byte



Addresses

- The first address is always reserved
  - We will never be able to use or access the byte at `0x00000000`
  - It is at the *top* of memory



Addresses

- On occasion, addresses will be printed without leading zeros, but you should assume they are there:
  - On a 64-bit processor, the address `0x59243d48c` is actually

`0x000000059243d48c`

Naming conventions

- Main memory is immediately available to the processor
  - Other storage devices are referred to *auxiliary memory*
  - Data from auxiliary memory must be loaded into main memory prior to being used by the computer
- Main and auxiliary memory may also be referred to as
  - *Internal* and *external* memory, respectively
  - *Primary* and *secondary* memory, respectively

Properties

- Main memory is almost universally volatile
  - In general, when you shut down a computer, main memory is erased
- Auxiliary memory is non-volatile
  - Removing a power source to auxiliary memory does not erase the contents
  - E.g., flash drives, magnetic disks, optical discs and magnetic tapes
- Main memory is often referred to as *random access memory* (RAM)
  - Any byte can be accessed in the same amount of time
  - Flash drives are also RAM

Summary

- Following this lesson, you now
  - Understand that memory is byte addressable
    - Each byte has its own unique integer address
  - Know that the byte is the smallest amount of memory that can be accessed
  - Know that addresses tend to be multiples of bytes in length
    - They are best described as hexadecimal numbers

References

[1] No references?

Colophon

These slides were prepared using the Georgia typeface. Mathematical equations use Times New Roman, and source code is presented using `Consolas`.

The photographs of lilacs in bloom appearing on the title slide and accenting the top of each other slide were taken at the Royal Botanical Gardens on May 27, 2018 by Douglas Wilhelm Harder. Please see

https://www.rbg.ca/

for more information.



Disclaimer

These slides are provided for the ECE 150 *Fundamentals of Programming* course taught at the University of Waterloo. The material in it reflects the authors' best judgment in light of the information available to them at the time of preparation. Any reliance on these course slides by any party for any other purpose are the responsibility of such parties. The authors accept no responsibility for damages, if any, suffered by any party as a result of decisions made or actions based on these course slides for any other purpose than that for which it was intended.