Computer Systems

Lecture 1

Analogue and Digital Representation

Dr José Cano, Dr Lito Michala
School of Computing Science
University of Glasgow

Spring 2020

Outline

- Course information
- Introduction to Computer Systems
- Analogue and Digital Representation

Course information

- Instructors
  - José Cano: Jose.CanoReyes@Glasgow.ac.uk, SAWB 206 (Wed 3-4 pm)
  - Lito Michala: AnnaLito.Michala@glasgow.ac.uk, SAWB 405 (Wed 1-2 pm)

- Lectures
  - Tuesday 12:00 – 13:00
    - Weeks 1-6: LC01 Adam Smith 1115, LC02 Main Bldg 255 Humanity
    - Weeks 7-11 LC01/LC02: Main Bldg 413 Kelvin Gallery
  - Thursday 12:00 – 13:00 (Main Bldg 420 Bute Hall)

- Labs
  - Weekly 2-hour lab at various times Wednesday to Friday (Boyd Orr 715)

Course information

- Check Moodle page frequently
  - Announcements and all course materials will be there
  - Online discussion forum (primary means to Q&A outside of class)

- Schedule
  - Two lectures every week
  - One 2-hour lab every week (no lab in first week)
  - One quiz on Moodle every week (available on Thursdays)

- Assessment
  - 80% degree examination in May
  - 10% quiz average (8 best scores)
  - 10% assessed exercise (assembly language programming)
  - You need to attend at least 7/10 labs to pass the course!

Time management

- You need regular study, so budget your time
  - You can't learn computer systems overnight!
  - Revise the lecture slides and handouts every week
  - Prepare the lab exercises before your lab

- Study tips
  - Focus on main concepts
  - Don't spend time memorising numbers!
  - Develop a working understanding
  - Try some simple examples

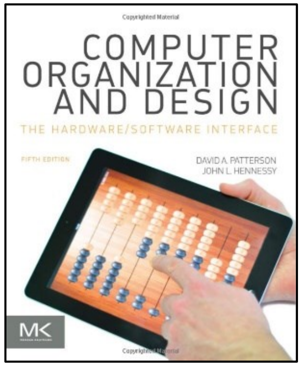
Voting on questions (YACRS)

- We will use the online voting system **YACRS** (Yet Another Classroom Response System)
  - https://classresponse.gla.ac.uk/

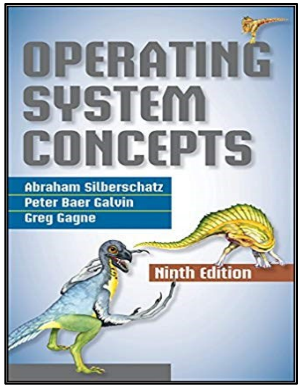
- Please bring your device (phone, tablet, laptop) to next lecture

Books

- **Computer Organization & Design**, 5th edition, 2013
  - Instruction sets, processor organisation, circuit design, interaction with operating system, etc



- **Operating System Concepts**, 9th edition, 2013
  - Some sections useful for this course



Outline

- Course information
- Introduction to Computer Systems
- Analogue and Digital Representation

Computer systems

- This course is about…

- We will see how computers work
  – Broad overview all the way from electronics up to systems software

Why learn about computer systems?

- **Intellectual**: a fascinating technology and a central aspect of our culture
- **Practical**: knowledge of computer systems is helpful in programming and improving application performance
- **Future-proofing**: you often need to design software solutions for the systems we'll have several years in the future
- **Understanding the basic concepts** of systems will help you to use computers more effectively
- Computing is a deep subject, with interesting history, ideas, theory, philosophy
  - It's more than just how to use software!

The goal of Computer Systems

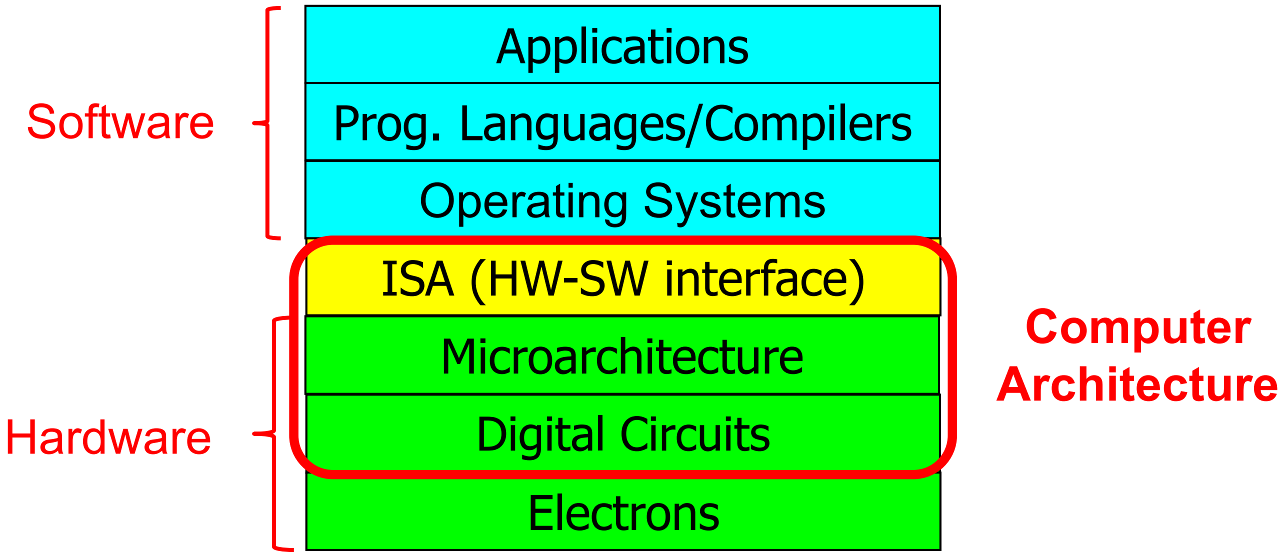
- Implement what we want using what we have
  - What we want is a usable computer system supporting programming and applications
  - What we have is digital electronics: transistors and wires

The challenge: Complexity

- There is a huge distance between the simple electronic components and the usable systems!
- Computer hardware
  - A processor contains around 10^7 electronic components (ten million)
- System software
  - A modern operating system contains around 10^7 lines of code (largely C and C++)

Levels of abstraction

- Applications: spreadsheets, databases, your own programs, …
- Programming languages and compilers
- Operating Systems
- Instruction Set Architecture (ISA): machine language
- Microarchitecture
- Digital Circuits
- Electrons



Digital circuits

- This subject consists of
  - The basic “building block” components
  - The way they behave when connected together
  - How to combine components in order to make a circuit that does something useful

- Surprisingly
  - There are only a few types of components
  - They are very simple
  - Yet when connected the right way, they give extraordinary behaviour

Instruction Set Architecture (ISA)

- The machine language (executed directly by the computer hardware)
  - Examples: MIPS, arm, x86, etc

- The interface between low-level circuits and high-level software

- Simple enough that it's possible to design a digital circuit that executes machine language

- Powerful enough that high level languages can be translated into it

Operating System

- System software that provides lots of facilities needed by applications, which are too complicated to provide directly in the hardware
  - Files
  - Protection
  - Processes
  - Threads
  - Virtual memory
  - Communication
  - etc

Types of computer systems

- Servers
  - Used for either few large tasks (e.g. engineering apps), or many small tasks (e.g. web server, Google)
  - Fast processors, lots of memory
  - Multi-user, multi-program

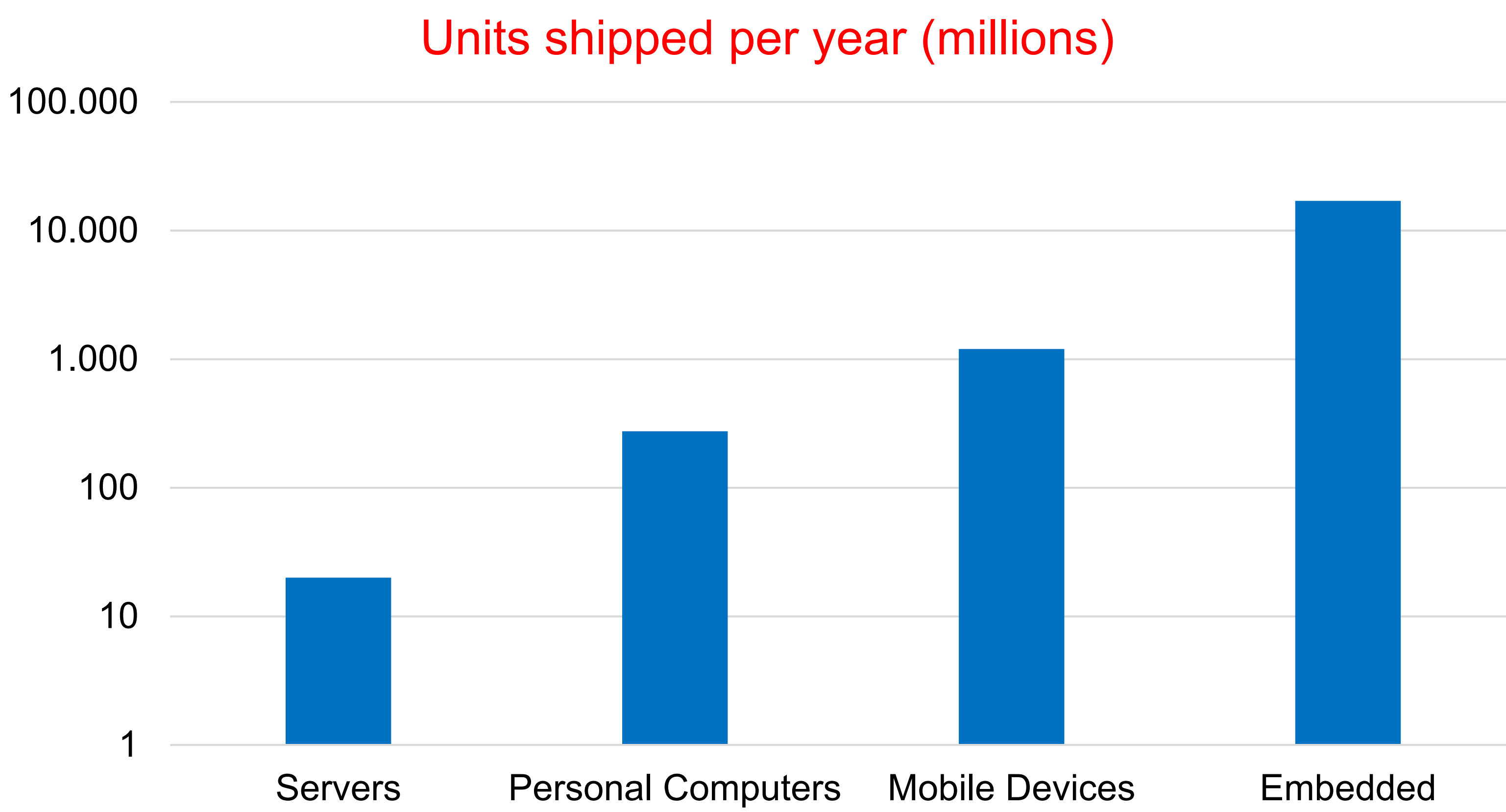
- Personal computers
  - Laptops, desktops
  - Balance cost, processing power
  - Few users, multi-program

Types of computer systems

- Mobile devices
  - Smart phones, tablets
  - Highly integrated (multiple processors, GPU, GPS, media accelerators, etc), low-power
  - Single-user, multi-program

- Embedded
  - Task specific: sensing, control, media playback, etc
  - Low-cost, low-power
  - Single program

Which computer system category is the largest?



Outline

- Course information
- Introduction to Computer Systems
- Analogue and Digital Representation

Data representation

- Data types used in programming
  - Numbers, character strings, booleans, data structures, etc
- They must all be represented in the computer hardware, which is based on basic electronic components
  - For example, we could use the voltage on a wire to represent information
- Computation then requires manipulation of these voltages
- There are two completely different approaches
  - Analogue
  - Digital

Analogue

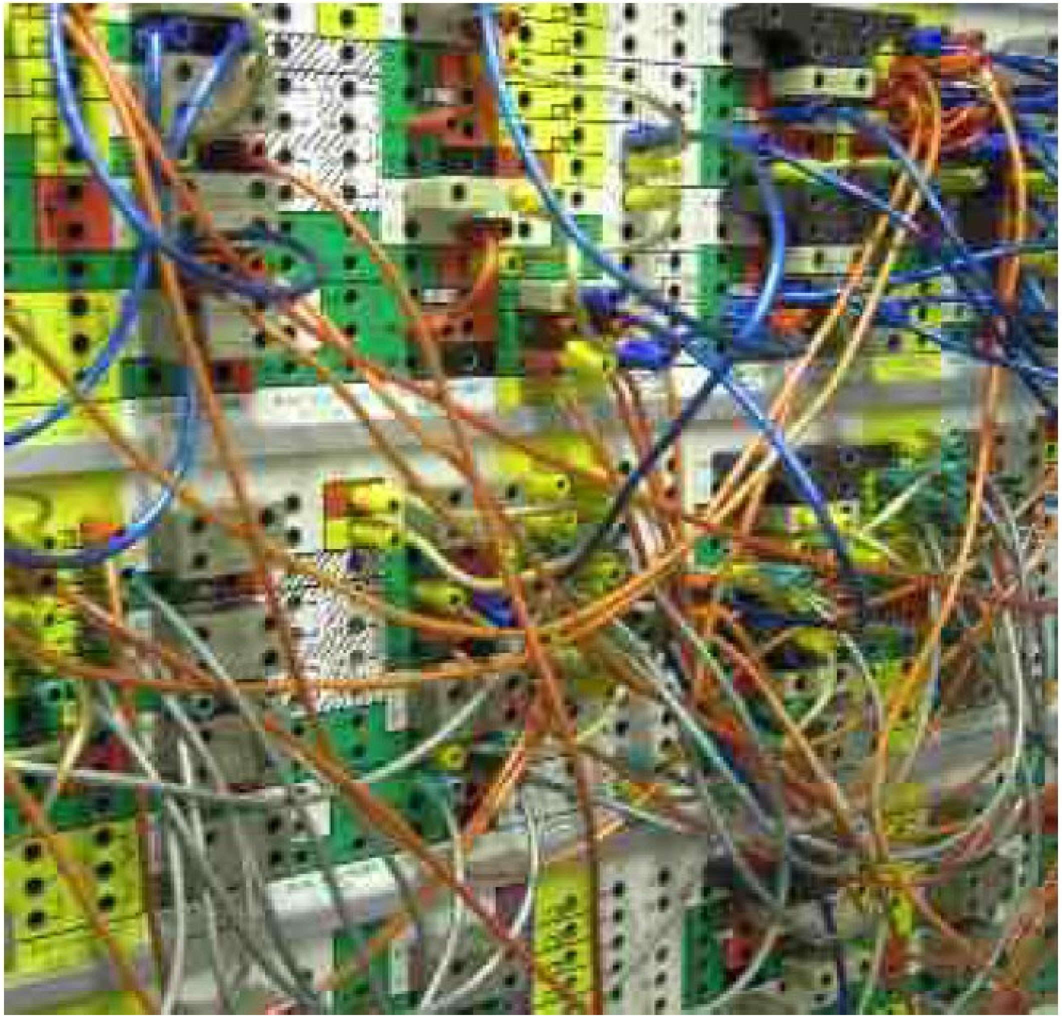
- **Idea**: make the variables in your problem proportional to a physical measure
  - Length, angle, rotation, voltage, etc

- For example, to represent the real number x, use
  - x volts
  - x meters of distance
  - x degrees of rotation

- The physical measure is an “analogy” or “analogue” to the measures in your problem
  - Example: an electronic analogue computer solving a differential equation (say, the heat equation) would make the voltage in the computer proportional to the heat in the object

Electronic analogue computer

Programming by wiring together functional units



Assessment of analogue

- Can be extremely fast
  - Example: differential equations

- But there are serious disadvantages…
  - Limited precision
  - After lots of calculations, errors will accumulate
  - It's hard to represent data other than real numbers (integers, strings, etc)
  - Large numbers can be shocking!

Digital computing

- Perhaps even older than analogue computing!
- How do you add two numbers (if you don't know the addition table)?
- Count on your fingers!
- This is digital computing
- The word digit means finger

Advantages of digital representation

- You can get as much precision as you want, just by using more digits to represent a number
- Good immunity to noise
- Errors don't accumulate after lots of calculations
- All datatypes can be represented
  - Integers, strings, objects, and much more

Binary Digits: Bits

- Voltage is used to represent information in digital circuits
- We call one of the standard voltages 0 and the other 1
- This unit of information is a Binary Digit, a Bit

Digital computing with bits

- Use just two voltages, it doesn't matter what they are, for example
  - TTL circuits: 0 volts and 5 volts
  - CMOS circuits: 2.5 volts and 2.5 volts

- The exact voltages are unimportant: what matters is that we can distinguish them!

- Circuits are simpler and more reliable if we just use two different voltages

- Use these two values to represent digits, and form numbers from strings of digits

Flip Flop

- A flip flop is a basic digital circuit that can
  - Take a data bit and store it (“remember it”)
  - Remember the value indefinitely
  - Read out the stored value

- The computer memory consists of a large number of these basic circuits

Bytes

- A byte is a string of 8 bits
- A byte is represented in the computer by 8 copies of the basic bit storage circuit
- Examples:
  - 0000 0000 0
  - 0110 1001 105
  - 1110 0100 228
  - 1111 1111 255
- We use spaces (not commas) to break it up into groups of 4 bits, to make it more readable

Information capacity of a byte

- There are exactly $2^8 = 256$ distinct values that can be represented in a byte
  - 00000000, 00000001, 00000010, 00000011, ... , 11111111
- There are many different ways to utilise the information capacity of a byte
- Each basic datatype (integer, character, etc) needs a representation method
  - That is, a way to represent that type using bits

Words

- A word is similar to a byte, but a larger amount of information

Short word	16 bits	2 bytes
Word	32 bits	4 bytes
Long word	64 bits	8 bytes

- The term “64-bit architecture” means that the internal hardware uses (mostly) 64-bit words

- Generally, a larger word size yields faster performance
  - Example: to do 64-bit addition on a 32-bit machine, you need two add instructions

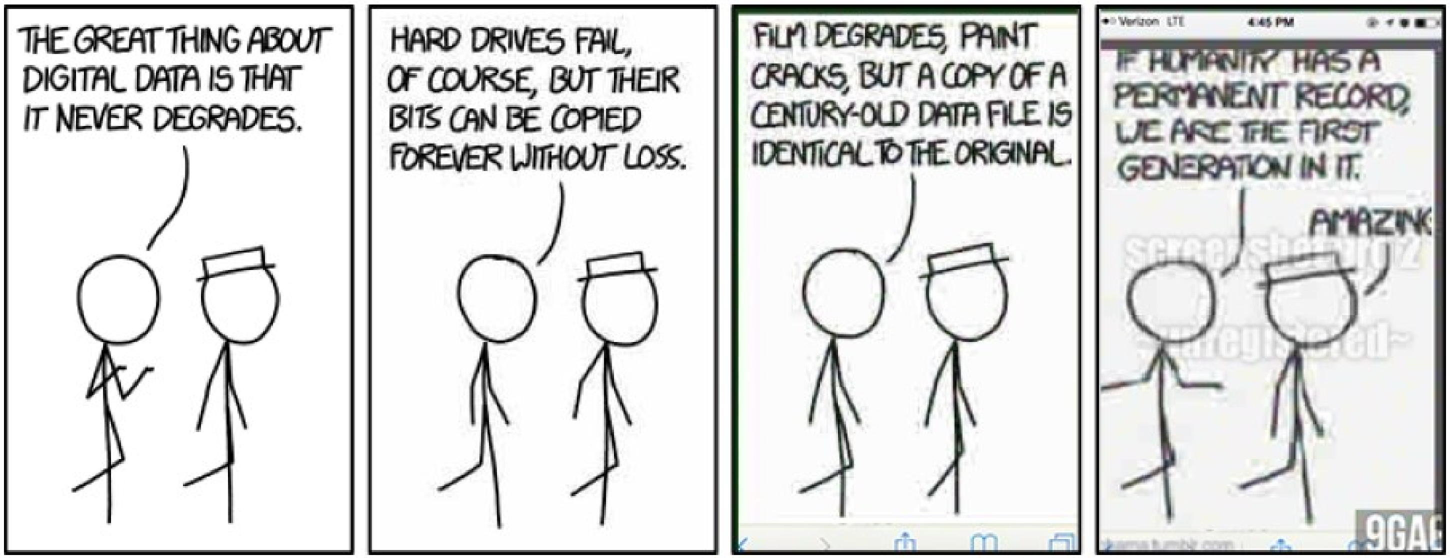
Information capacity of a word

- A word containing k bits is called a k-bit word, and can represent 2^k distinct values

Word size	Number of values
8	$2^8 = 256$
16	$2^{16} = 65{,}536$
32	$2^{32} = 4{,}294{,}967{,}296$
64	2^{64} = gigantic huge number

To do

- Check the Moodle page for this course
- Tutorial/lab starting next week, find out which group you're in
- Re-read the slides, see if you have any questions



https://xkcd.com/1683/