

# Inf2C - Computer Systems

## Lecture 1

### Course overview & the big picture

---

Boris Grot

School of Informatics

University of Edinburgh



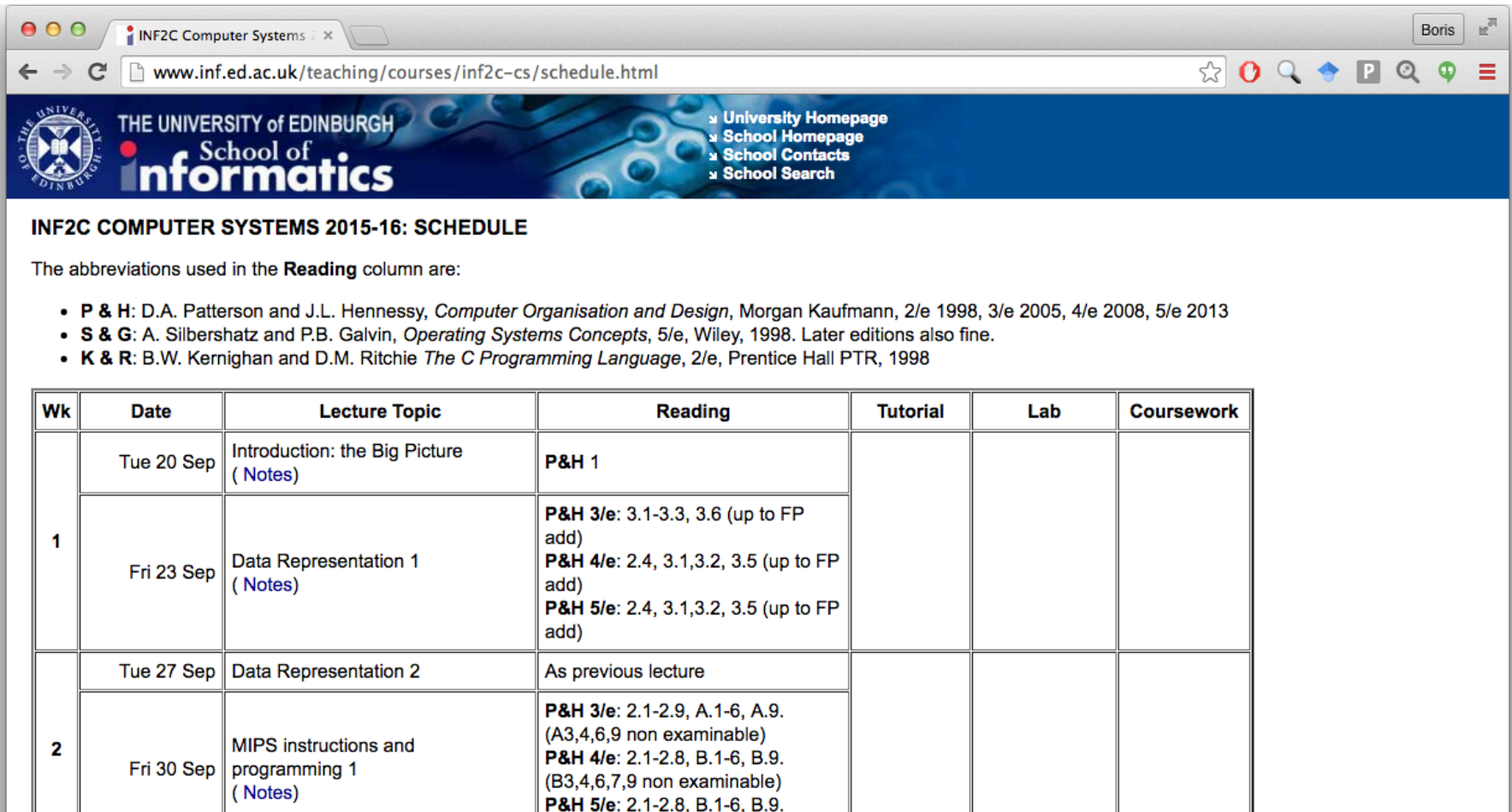
# Practicalities

---

- Lectures:
  - Tue & Fri, David Hume Tower, Lec. Hall A @ 15:10– 16:00
- Tutorials: weeks 3, 5, 7, 9
- Labs: Drop-in format.
  - Demonstrators available to help
- Online discussion forum: ASK <https://ask.sli.is.ed.ac.uk>
  - Primary means to Q&A outside of class.
- Study resources: slides, book, lecture notes (online)
- All material are/will be on the course web-page:  
<http://www.inf.ed.ac.uk/teaching/courses/inf2c-cs>



# Lecture schedule, slides, notes



The screenshot shows a web browser window with the URL [www.inf.ed.ac.uk/teaching/courses/inf2c-cs/schedule.html](http://www.inf.ed.ac.uk/teaching/courses/inf2c-cs/schedule.html). The page header includes the University of Edinburgh School of Informatics logo and navigation links: University Homepage, School Homepage, School Contacts, and School Search. The main heading is "INF2C COMPUTER SYSTEMS 2015-16: SCHEDULE". Below this, a note states: "The abbreviations used in the Reading column are: P & H: D.A. Patterson and J.L. Hennessy, *Computer Organisation and Design*, Morgan Kaufmann, 2/e 1998, 3/e 2005, 4/e 2008, 5/e 2013; S & G: A. Silberschatz and P.B. Galvin, *Operating Systems Concepts*, 5/e, Wiley, 1998. Later editions also fine; K & R: B.W. Kernighan and D.M. Ritchie *The C Programming Language*, 2/e, Prentice Hall PTR, 1998". The schedule table follows, with columns for Week, Date, Lecture Topic, Reading, Tutorial, Lab, and Coursework.

Wk	Date	Lecture Topic	Reading	Tutorial	Lab	Coursework
1	Tue 20 Sep	Introduction: the Big Picture ( <a href="#">Notes</a> )	P&H 1			
	Fri 23 Sep	Data Representation 1 ( <a href="#">Notes</a> )	P&H 3/e: 3.1-3.3, 3.6 (up to FP add) P&H 4/e: 2.4, 3.1, 3.2, 3.5 (up to FP add) P&H 5/e: 2.4, 3.1, 3.2, 3.5 (up to FP add)			
2	Tue 27 Sep	Data Representation 2	As previous lecture			
	Fri 30 Sep	MIPS instructions and programming 1 ( <a href="#">Notes</a> )	P&H 3/e: 2.1-2.9, A.1-6, A.9. (A3,4,6,9 non examinable) P&H 4/e: 2.1-2.8, B.1-6, B.9. (B3,4,6,7,9 non examinable) P&H 5/e: 2.1-2.8, B.1-6, B.9.			

**Schedule will drift. It's OK.**



# Books

---

- **Required:** *Patterson & Hennessy: Computer Organization and Design, Morgan Kaufmann*
  - 5<sup>th</sup> or 4<sup>th</sup> ed recommended
  - Several copies on reserve in the library and online
- *Silberschatz, Galvin, Gagne: Operating Systems Concepts, Wiley 9<sup>th</sup> ed*
  - Library has 5<sup>th</sup> and 7<sup>th</sup> ed ebook (both OK)
  - Only a few sections needed for this course
- *Kernighan and Ritchie. The C Programming Language, Prentice Hall 2<sup>nd</sup> ed*
  - Generally useful, but not mandatory for this course



# Exam and Coursework

---



- Exam - 60%
  - In December; exact date not available yet.
  - Must achieve at least 40/100 to pass the course
- Coursework – 40%
  1. MIPS assembly programming
    - Out: Tue 11 Oct (week 4)
    - Due: Wed 26 Oct (week 6) @ 4pm
  2. TBD (will involve C programming)
    - Out: Tue 8 Nov (week 8)
    - Due: Wed 23 Nov (week 10) @ 4pm
- Must achieve at least 40% in total to pass the course



# Late coursework

---

- School-wide consistent policy:  
**Normally, you will not be allowed to submit coursework late**
- If you have a **really good reason** to submit late, contact the ITO via their Support Form.
  - The ITO will log the report and pass it on to the UG2 Course/Year Organiser (Dr. Rik Sarkar)
  - Only in exceptional circumstances (*e.g.*, illness that stopped you getting to email), would an extension be granted after a deadline has passed
- See the online Undergraduate Year 2 Handbook for details



# What is “good reason”?

---

Something that, in the judgement of the member of staff responsible, would prevent a competent, well-organised, conscientious student from being able to submit on time.

## Examples:

- Significant illness
- Serious personal problems

## Non-examples:

- Difficult cluster of deadlines
- Last-minute computer problems, dog ate your homework, ...

# Plagiarism (aka Cheating)

---

- How cheating is defined in this course
  - Any similarity in the submitted code
- How we check for cheating:
  - Pairwise comparisons between all submissions
  - Two tools: MOSS (commercial grade) and an internal tool
    - Rely on sophisticated “fingerprinting” techniques that are rename and reorder proof
- I have a great track record of hunting down cheaters
  - Will go to great lengths to prosecute to the max extent



**DON'T DO IT!!!**



# How to do well in this class

---

- Get started on assignments early
  - They are intended to be work-intensive
- Take advantage of labs
  - But don't wait till the last day → demonstrators will be swamped and will not have time for all
- Don't ignore tutorials
  - Advance prep will allow you to focus on nuances
- Keep up with the reading and the lectures
- Don't cheat/plagiarise





# So what is this course about?

---



# Syllabus Overview

---

## ■ Hardware:

- Data representation and operations
- Design of (very) simple circuits
- Processor organisation
- Exceptions and interrupts
- The memory subsystem
- Input/Output (I/O)

## ■ Software:

- Low-level (assembly) programming
- Operating systems basics
- Introduction to C programming



# Let's dig in...

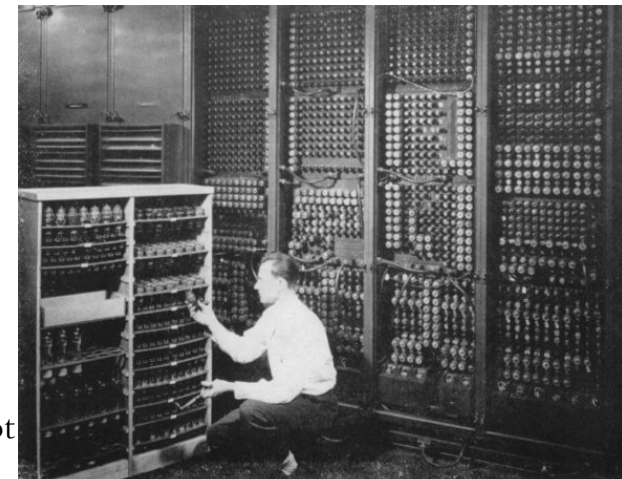
---



# Evolution of computers

---

- Early computers had their programs set up by plugging cables and setting switches
- **John von Neumann** first proposed to store the program in the computer's memory
- All computers since then (~1945) are stored-program machines

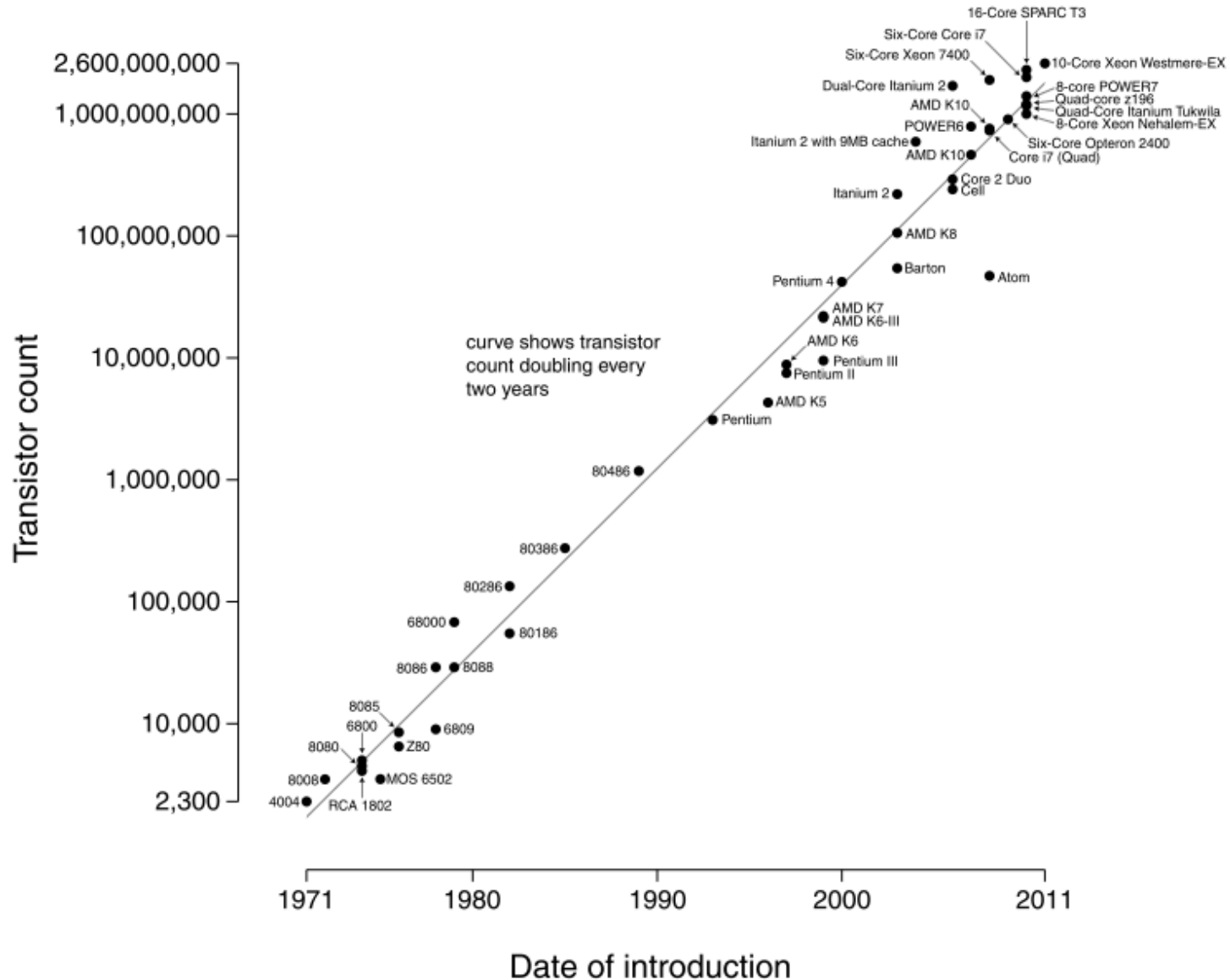


# Evolution of computers

---

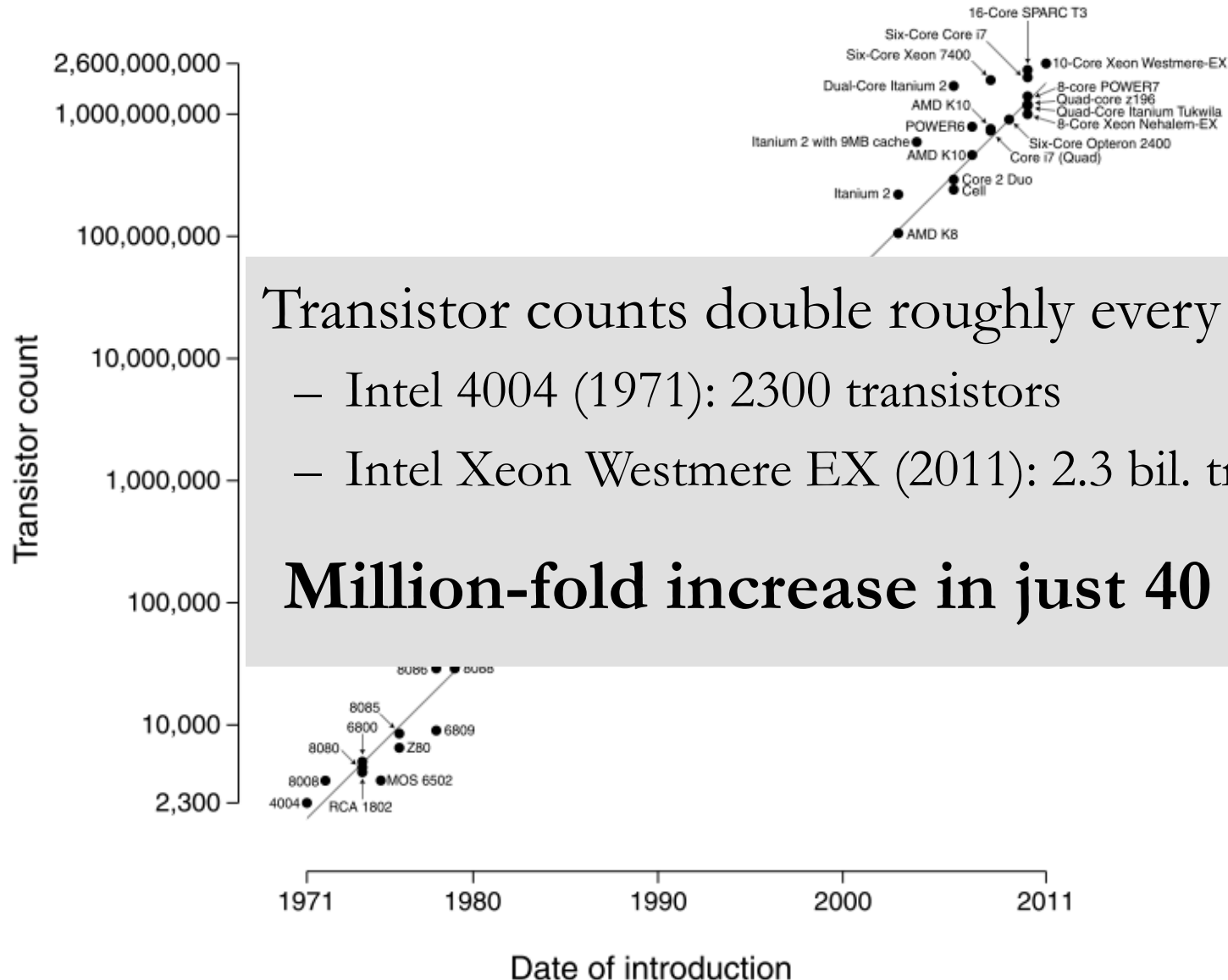
- What has changed is the number of transistors (electronic switches) and their speed
- Implementation technology progressed from vacuum tubes to discrete bipolar transistors to (eventually) Integrated Circuits (a.k.a. chips) made with complementary metal-oxide semiconductor (CMOS) technology.
- At the same time, the cost per transistor has been dropping

# Moore's law





# Moore's law



# 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

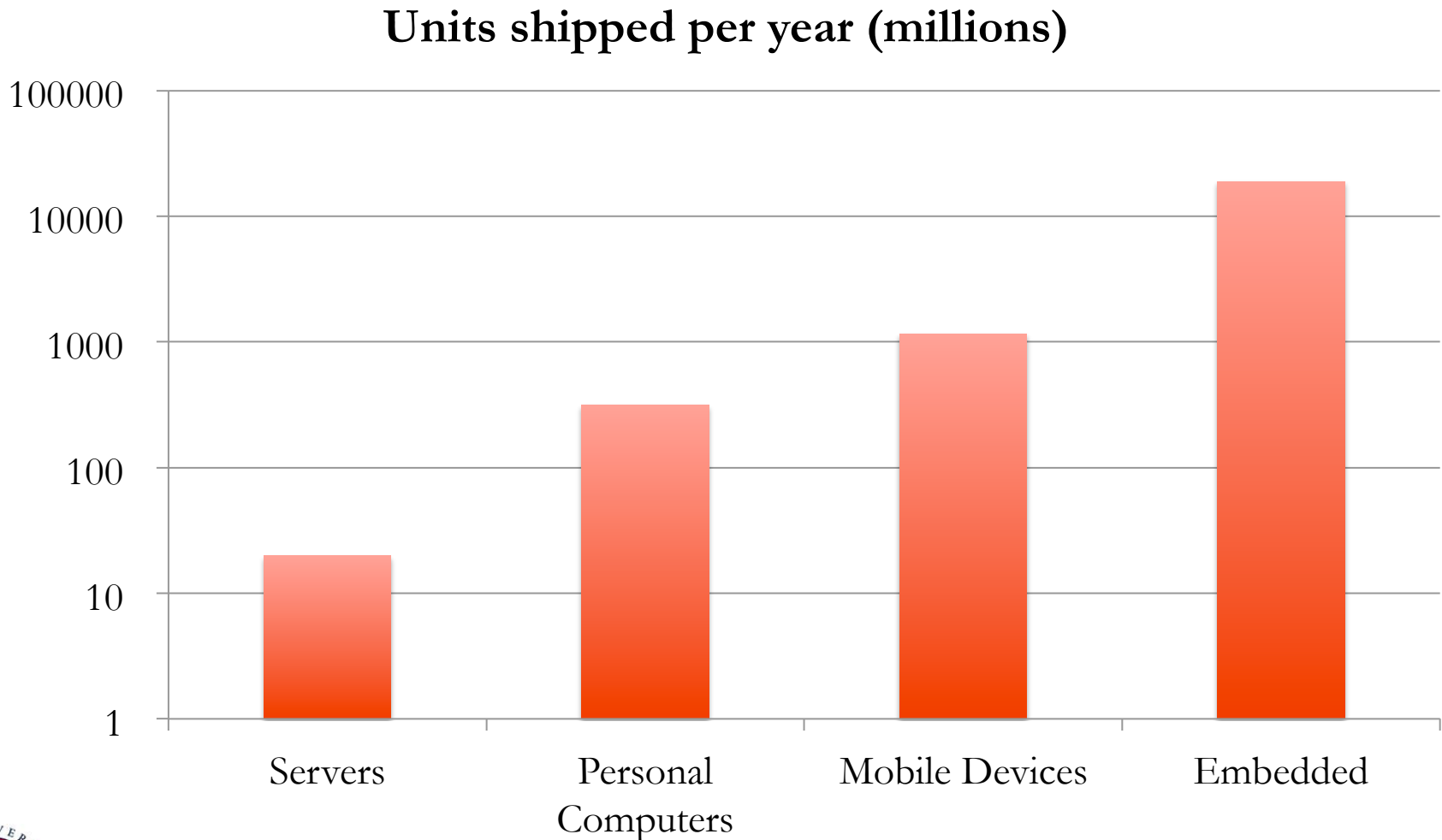
# Types of computer systems (con'd)

---

- 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

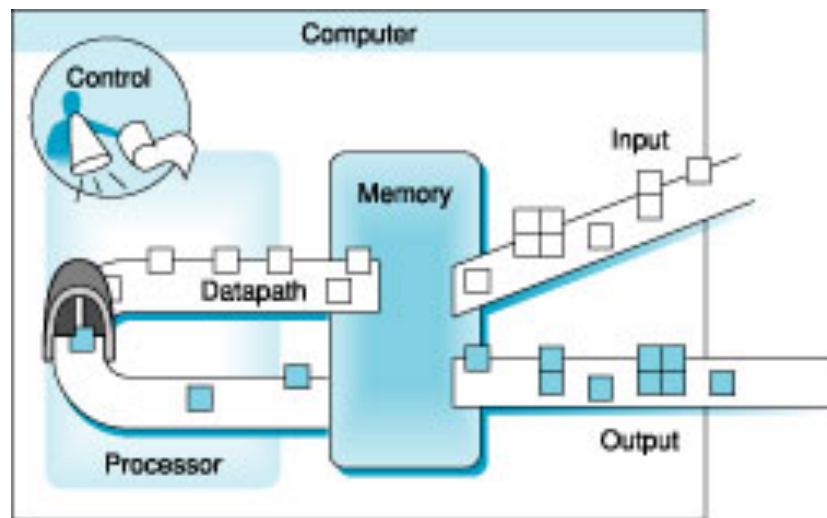
# Which computer system category is the largest?

---

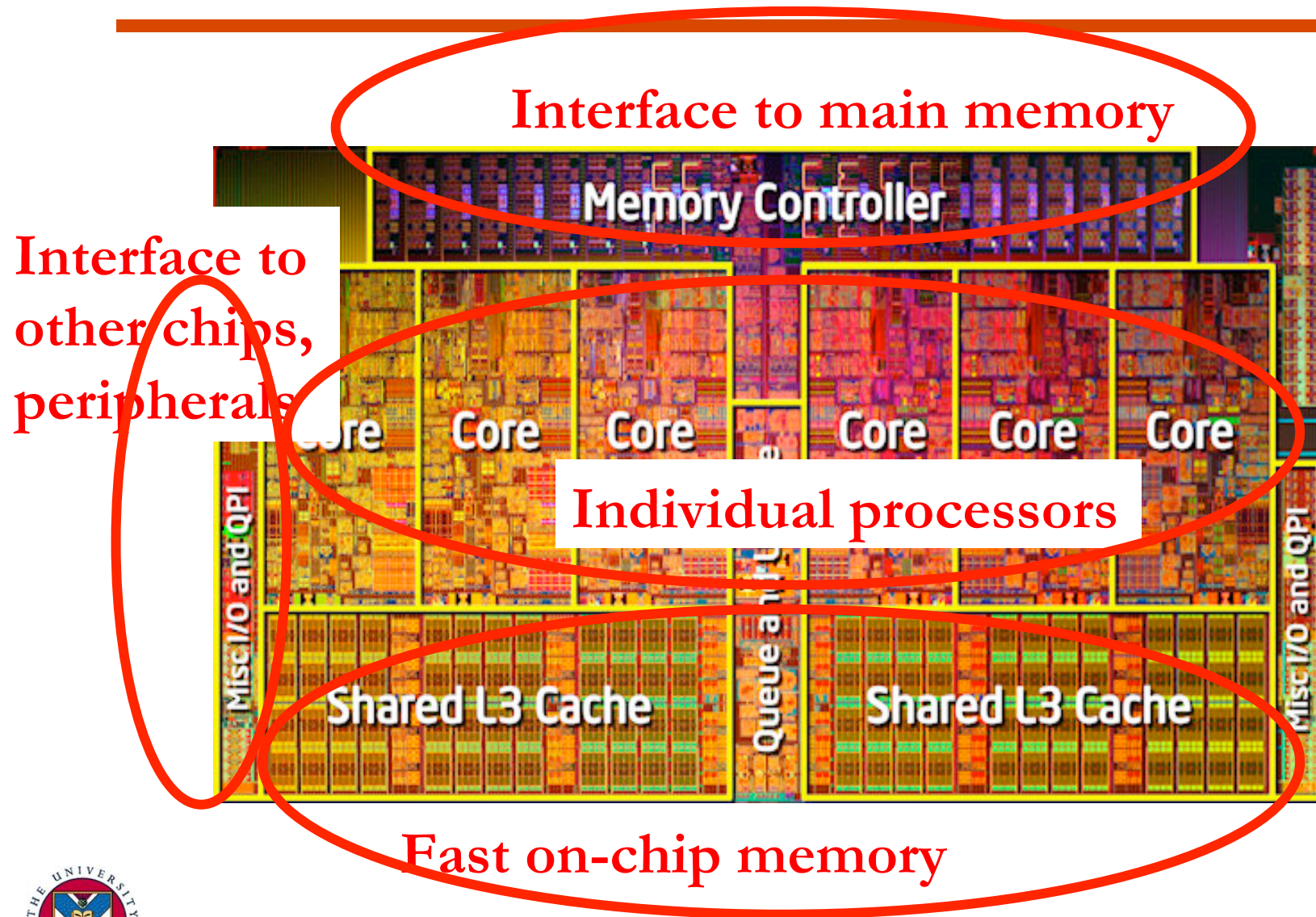


# Computer components

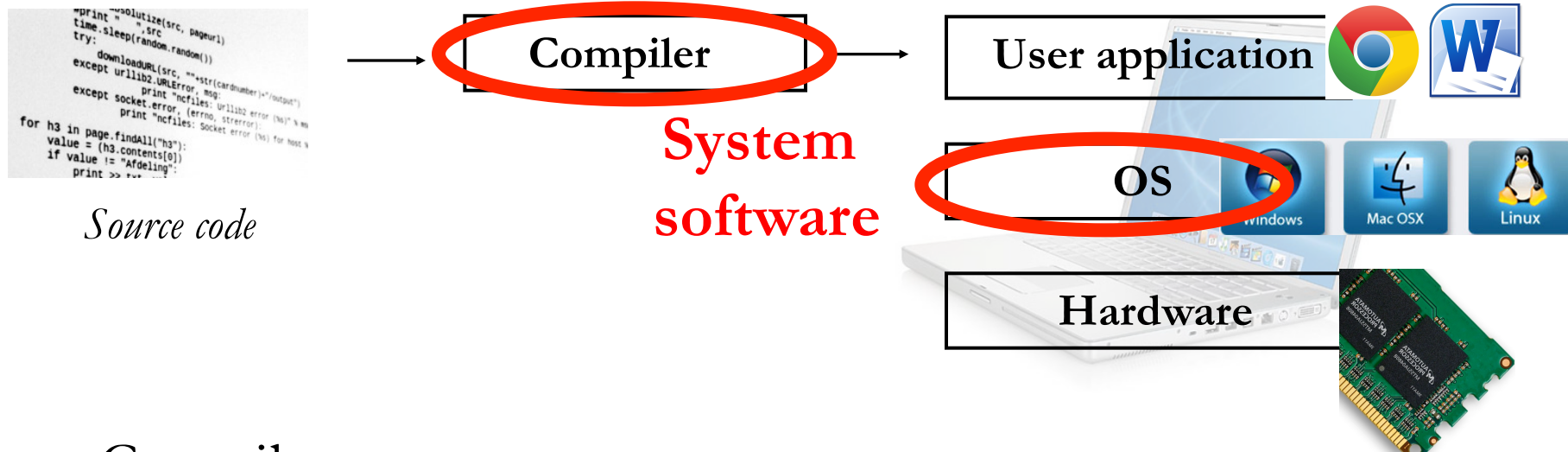
- Data path
    - Performs actual operations on data
  - Control path
    - Fetches instructions from program in memory
    - Controls the flow of data through the data path
- } Processor
- Memory
    - Stores data and instructions
  - Input/Output
    - Interfaces with other devices for getting/giving data



# A modern processor



# Modern computer system



- Compiler
  - Translates **High Level Language (HLL)** into **machine language** or **byte code**
- Operating System (OS)
  - Mediates access to hardware resources (CPU, Memory, I/O)
  - Schedules applications



# Words of wisdom & caution

---

- This class covers a lot of material
  - Keeping up will require effort on your part
  - This ain't no INF1!
- Attend all lectures, tutorials and labs
- Get started on assignments early
  - They are meant to be work-intensive
- ASK QUESTIONS!

**Reward: you will learn a lot!**





