THE UNIVERSITY *of* EDINBURGH
informatics

Computer Architecture and Design

INFR10076

Lecture 1 – Introduction

Nigel Topham

Institute for Computing Systems Architecture

School of Informatics

The University of Edinburgh

Overview

- What is Computer Architecture and Design?
- Course introduction and administrative information
- The technological context for Computer Architecture and Design

What is Computer Architecture and Design?

- **Architecture**
  - Science and art of interconnecting building **materials** to construct various **buildings**, subject to **constraints**
  - Materials: brick, concrete, glass etc.
  - Buildings: house, office, auditorium etc.
  - Constraints: cost, safety, time etc.

- **Design**
  - How to turn a blueprint into a reality

- In a **computational** context...
  - Concerned with science and art of interconnecting **logic and memory structures** to make **computers**...
  - and how to design those structures
  - Constrained by cost, power, performance requirements, etc.

The Hierarchy of Abstraction in Computer Architecture

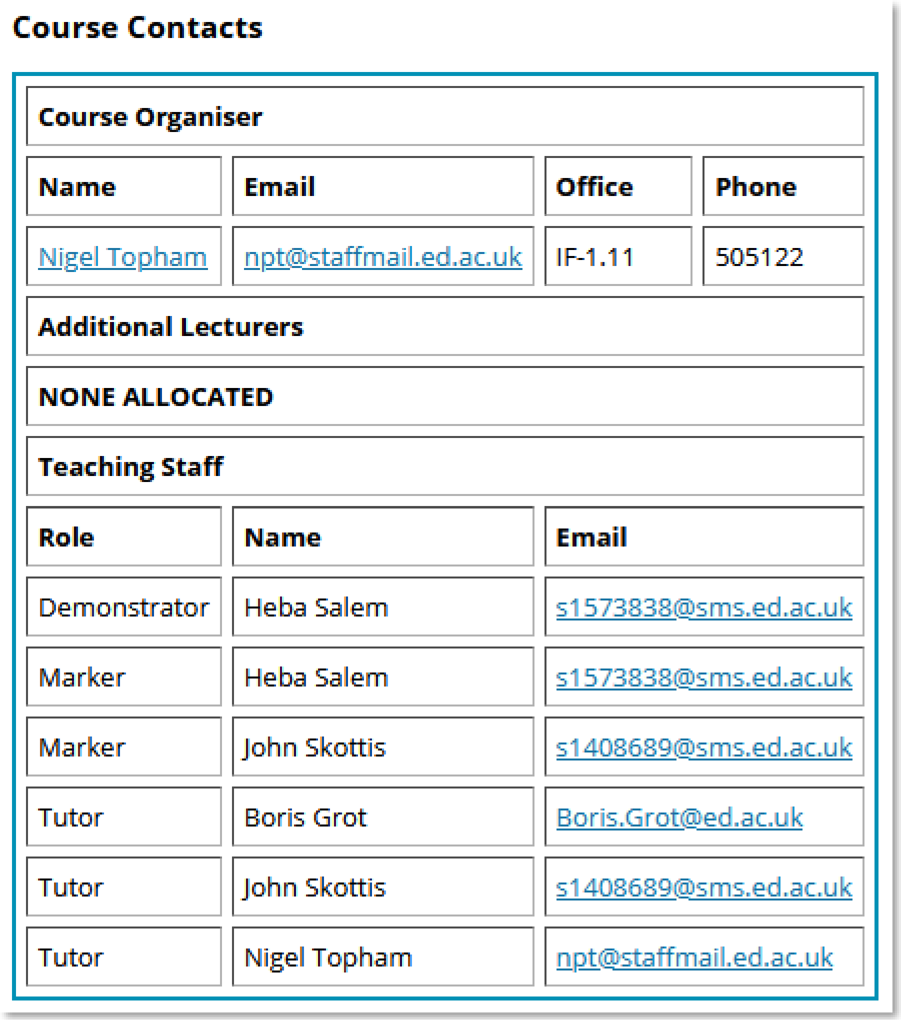
- Instruction Set Architecture (ISA)
  - Programmer's / compiler's view
  - Instructions visible to the (system) programmer
  - Opcodes, architectural registers, address translation, etc.
- Microarchitecture
  - Processor designer's view
  - Structures that implements the ISA
  - Pipelining, functional units, caches, registers, etc.
  - Computer arithmetic, memory systems, bus protocols, etc.
  - Hardware Description Languages, synthesis, timing, etc.
- Circuits
  - Circuit/chip designer's view
  - Gates, cells, CMOS process, packaging, etc.

Our focus

Administrative Stuff

- Teaching team
- Lectures
- Assessment
- Syllabus
- Learning outcomes
- Textbooks
- Coursework assignments
- Tutorials and informal homework exercises

Teaching Team



Course Contacts

Course Organiser

Name	Email	Office	Phone
Nigel Topham	npt@staffmail.ed.ac.uk	IF-1.11	505122

Additional Lecturers

NONE ALLOCATED

Teaching Staff

Role	Name	Email
Demonstrator	Heba Salem	s1573838@sms.ed.ac.uk
Marker	Heba Salem	s1573838@sms.ed.ac.uk
Marker	John Skottis	s1408689@sms.ed.ac.uk
Tutor	Boris Grot	Boris.Grot@ed.ac.uk
Tutor	John Skottis	s1408689@sms.ed.ac.uk
Tutor	Nigel Topham	npt@staffmail.ed.ac.uk

CArD - Course Activities

- Three lectures per week in weeks 1 – 6

Mon.	12:10 – 13:00
Thur.	12:10 – 13:00
Fri.	14:10 – 15:00

- Two lectures per week in weeks 7 – 10 (Mon & Thur only)
- No hand-outs, the lecture slides are available online
- Lectures focus on concepts and principles
- Practical lab exercises focus on processor design
- Piazza discussion forum – please sign up here!

  http://piazza.com/ed.ac.uk/fall2021/infr10076

7020

- See the CArD course on Learn for schedule details

Assessment

- The written exam is in December
- Summative assessment
  - Written exam = 60%
  - Practical assignments = 40%
- Formative assessment
  - Laboratory instruction and feedback
  - Tutorials
  - Feedback on exercise sheets

Syllabus – full details...

- **Fundamentals:**
  - Review of logic design and implementation technologies
  - From simple combinational logic to state machines for sequential circuits
  - Logic design using Verilog and introduction to FPGAs.
  - Register Transfer Level design principles; registers, clocks, timing budgets, setup and hold margins, clock skew, clock-domain crossing and synchronization, metastability.
  - Quantitative computer architecture; performance evaluation methods and metrics, principles of high-performance design.
- **Processor Architecture:**
  - Instruction Set Architecture (ISA) design; instruction set classes, registers, memory addressing.
  - RISC vs CISC, how the ISA supports high-level languages, quantitative approach to ISA design.
  - Example ISAs (e.g. MIPS, RISC-V).
  - ISA requirements for embedded systems.
  - Pipelined processor design; pipeline hazards and interlocks, control prediction techniques and their usage.
  - Out-of-order execution; scoreboards, reservation stations, register renaming, quantitative analysis of performance.
- **Computer Arithmetic and ALU Design:**
  - Introduction to binary arithmetic functions; fixed-point addition, subtraction, multiplication and division.
  - Advanced techniques in computer arithmetic; carry-look ahead adders, parallel-prefix adders, Booth-coded multipliers, Wallace and Dadda trees.
  - Sub-word parallelism.
  - fractional fixed-point multiply- accumulate operations.
  - Floating-point computations; IEEE standard, floating-point addition and multiplication, high-performance fused-multiply-add architectures.
- **Memory System Design:**
  - Memory hierarchies; review of principles, quantitative analysis of memory hierarchy performance; exploring the design space of cache parameters.
  - Cache coherence in multi-core architectures; protocols and implementation techniques.
  - Main memory design; Interfacing between processor and memory, synchronous and asynchronous bus protocols.
  - Error detection and correction schemes; parity, Hamming codes, SECDED.

Learning Outcomes

- Describe the structure and operating characteristics of a high-performance microprocessor, and explain the principles of: orthogonal instruction set design; pipeline hazards and interlocks; branch prediction (both static and dynamic); out-of-order execution.
- Explain the design and operating principles of arithmetic units including: high-speed adders and multipliers; dividers; and floating-point units. And also demonstrate how selected fixed-point arithmetic functions can be implemented (in a laboratory setting).
- Design and implement both combinational and synchronous digital systems using state-of-the-art FPGA design tools and hardware description languages.
- Describe the structure and operating characteristics of memory systems; demonstrate the ability to evaluate quantitatively the performance of a combined processor and memory system with respect to cycles-per-instruction (CPI) and memory bandwidth requirements; describe the operating principles of error detection and correction techniques applied to memory systems, and design a SECDED solution for a given memory system.
- Reason about the ways in which memory hierarchies can be configured to exploit locality in order to reduce average memory access times, and quantitatively evaluate the impact of varying cache design parameters (e.g. capacity, associativity, block size, and write policies) on performance; understand the operating principles of cache coherency protocols, and be able to compare and contrast different implementation techniques.

Textbooks

1. Hennessy & Patterson, Computer Architecture: A Quantitative Approach, Elsevier, 5th or 6th editions
2. Morris Mano, Michael D. Ciletti: Digital Design, 4/e, Prentice Hall, 2007.
3. Hamacher, Vranesic & Zaky: Computer Organization, McGraw-Hill, 2001.
4. Mano & Kime: Logic and Computer Design Fundamentals, Pearson, 2008.
5. Patterson & Hennessy: Computer Organization and Design, Elsevier, 2005.
6. Null & Lobur: The Essentials of Computer Organization and Architecture, Jones and Bartlett Publishers, 2003.

Please refer (frequently) to the library Resource List for CArD

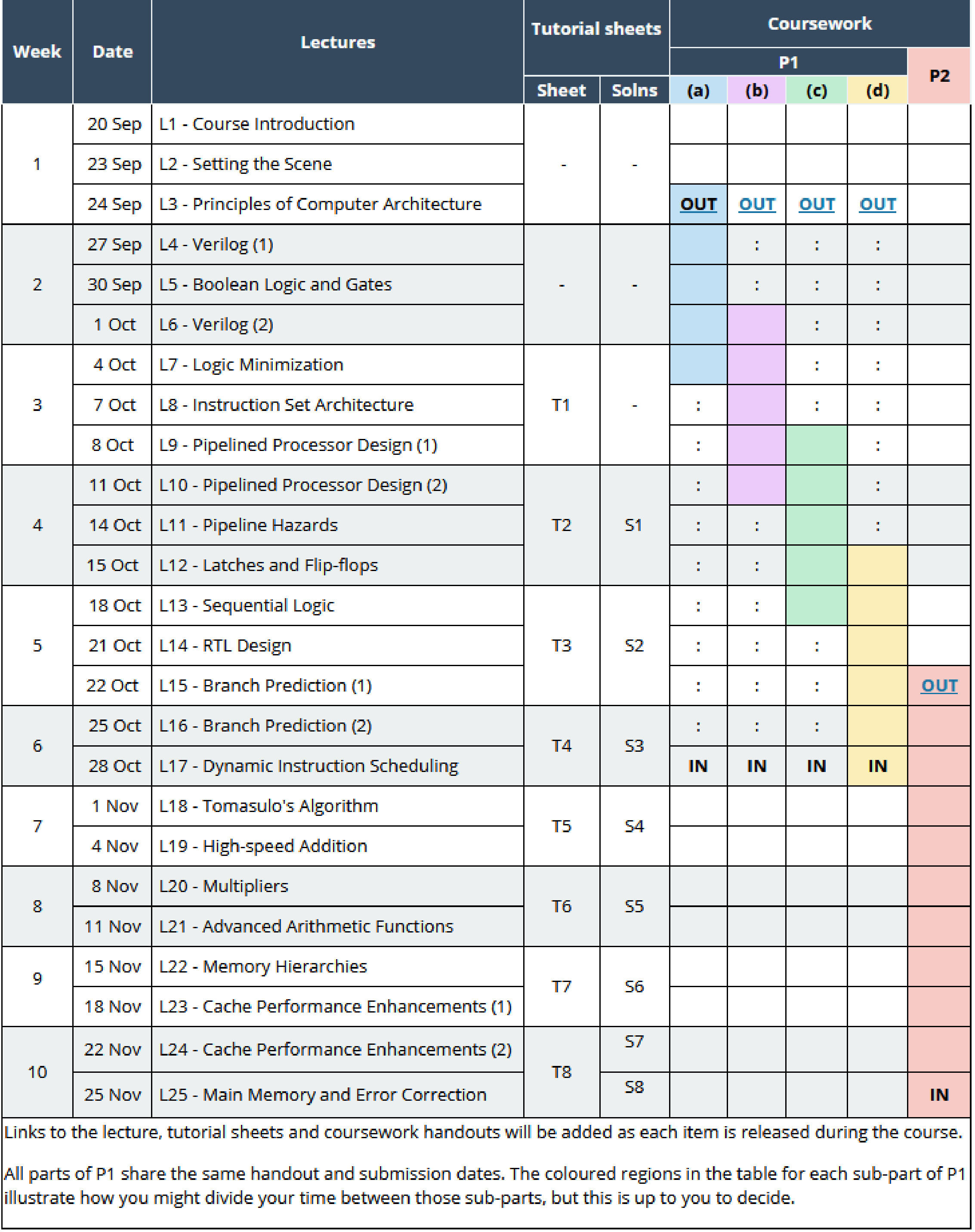
Coursework Overview

- Two practicals, P1 and P2, roughly in first and second half of the semester respectively
- P1 is a set of four computer design exercises
  - P1.(a) through P1.(d)
  - Two online drop-in sessions per week (Tue, Wed: 2-4pm)
  - In-person lab sessions will be scheduled if possible
- P2 is a simulation-based computer architecture exercise
- Consult the Schedule Table in **LEARN**

Schedule Table

- Course Materials
- At a glance
- Week by week

- Lecture slides
- Tutorial sheets
- Tutorial solutions
- Assignments
  - Handout (OUT)
  - Deadline (IN)
  - Links to details



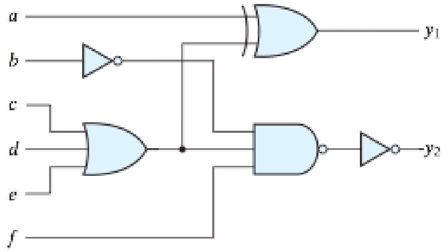
Week	Date	Lectures	Tutorial sheets		Coursework				
					P1				P2
			Sheet	Solns	(a)	(b)	(c)	(d)	
1	20 Sep	L1 - Course Introduction	-	-					
	23 Sep	L2 - Setting the Scene							
	24 Sep	L3 - Principles of Computer Architecture			OUT	OUT	OUT	OUT	
2	27 Sep	L4 - Verilog (1)	-	-	🟦	:	:	:	
	30 Sep	L5 - Boolean Logic and Gates			🟦	:	:	:	
	1 Oct	L6 - Verilog (2)			🟦	🟪	:	:	
3	4 Oct	L7 - Logic Minimization	T1	-	🟦	🟪	:	:	
	7 Oct	L8 - Instruction Set Architecture			:	🟪	:	:	
	8 Oct	L9 - Pipelined Processor Design (1)			:	🟪	🟩	:	
4	11 Oct	L10 - Pipelined Processor Design (2)	T2	S1	:	🟪	🟩	:	
	14 Oct	L11 - Pipeline Hazards			:	:	🟩	:	
	15 Oct	L12 - Latches and Flip-flops			:	:	🟩	🟨	
5	18 Oct	L13 - Sequential Logic	T3	S2	:	:	🟩	🟨	
	21 Oct	L14 - RTL Design			:	:	:	🟨	
	22 Oct	L15 - Branch Prediction (1)			:	:	:	🟨	OUT
6	25 Oct	L16 - Branch Prediction (2)	T4	S3	:	:	:	🟨	🟥
	28 Oct	L17 - Dynamic Instruction Scheduling			IN	IN	IN	IN	🟥
7	1 Nov	L18 - Tomasulo's Algorithm	T5	S4					🟥
	4 Nov	L19 - High-speed Addition							🟥
8	8 Nov	L20 - Multipliers	T6	S5					🟥
	11 Nov	L21 - Advanced Arithmetic Functions							🟥
9	15 Nov	L22 - Memory Hierarchies	T7	S6					🟥
	18 Nov	L23 - Cache Performance Enhancements (1)							🟥
10	22 Nov	L24 - Cache Performance Enhancements (2)	T8	S7					🟥
	25 Nov	L25 - Main Memory and Error Correction		S8					IN

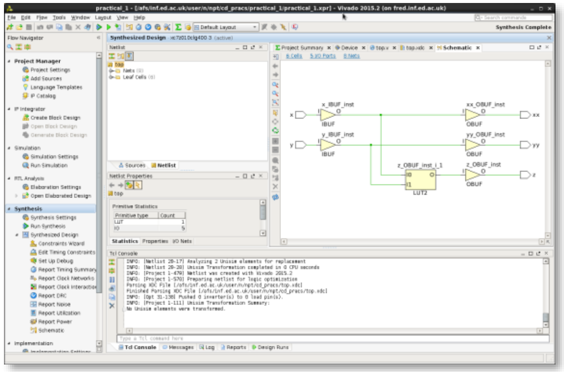
Links to the lecture, tutorial sheets and coursework handouts will be added as each item is released during the course.

All parts of P1 share the same handout and submission dates. The coloured regions in the table for each sub-part of P1 illustrate how you might divide your time between those sub-parts, but this is up to you to decide.

CArD Computer Design assignment P1

- Learn how to design logic
- Design parts of a RISC-V CPU
- Optimize the microarchitecture
- Implement your designs in an FPGA
- Evaluate your design
- Documents available from the Practical Assignments page in **LEARN**




practical_1 - [/afs/inf.ed.ac.uk/user/n/npt/cd_pracs/practical_1/practical_1.xpr] - Vivado 2015.2 (on fred.inf.ed.ac.uk)

Synthesis Complete

Synthesized Design - xc7z010clg400-3 (active)

Primitive Statistics

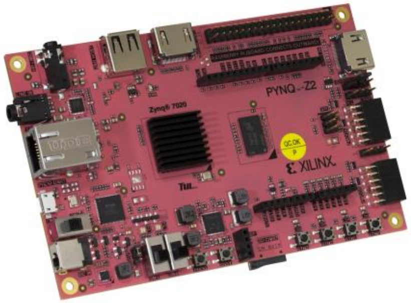
Primitive type	Count
LUT	1
IO	5

Tcl Console

```
INFO: [Netlist 29-17] Analyzing 2 Unisim elements for replacement
INFO: [Netlist 29-28] Unisim Transformation completed in 0 CPU seconds
INFO: [Project 1-479] Netlist was created with Vivado 2015.2
INFO: [Project 1-570] Preparing netlist for logic optimization
Parsing XDC File [/afs/inf.ed.ac.uk/user/n/npt/cd_pracs/top.xdc]
Finished Parsing XDC File [/afs/inf.ed.ac.uk/user/n/npt/cd_pracs/top.xdc]
INFO: [Opt 31-138] Pushed 0 inverter(s) to 0 load pin(s).
INFO: [Project 1-111] Unisim Transformation Summary:
No Unisim elements were transformed.
```

Equipment, Drop-in Sessions and Schedule for P1

- Get FPGA board, 7-segment display, USB cable from Garry Ellard
  - Available: **24, 25, 27 September, 10am-12pm, in AT 3.09**



PYNQ-Z2 board

7-segment display

Micro USB cable

- Follow instructions in handout for P1.(a) to get up and running with FPGA hardware and software at home

- Online drop-in sessions during weeks 2, 3, 4, 5 and 6
  - Tue, Wed afternoon, 2:10pm – 4pm
  - AT 3.09 lab is also open for use, without instructors, if you prefer to use DICE machines instead of your own laptop / desktop machine

- See Schedule Table for hand-out / hand-in dates

Tutorials, Example Sheets, and Solutions

- Tutorials take place online in weeks 3 – 10
- Tutorial example sheets (T1 – T8) with problem-solving exercises, related to topics in high-performance computer architecture and design
- Tutorials are also a forum for discussing assignments
- Solutions to example sheets (S1 – T8) available from the Schedule Table one week after the tutorial
- Check out the announcements on LEARN at the start of each week – these will signpost you information about that week’s activities