IMPERIAL

Module Specification

Basic details

UID		Cohorts covered (Earliest cohort)	
Long title	Instruction Architectures and Compilers		
New code	ELEC50010	New short title	IAC
Brief description of module *(approx. 600 chars.)*	In this module, you will learn how to design a modern general-purpose processor, and how to write a compiler that enables your processor to run software written in a high-level programming language. Specifically, in the first half of the module (Instruction Architectures, Autumn term), you will be guided through the process of building your own processor using the Verilog hardware description language, following the RISC-V architecture. In the second half of the module (Compilers, Spring term), you will build your own compiler that is capable of translating software written in the C programming language into low-level instructions that your RISC-V processor can execute		
Available as a standalone module/ short course?	N		

Statutory details

	ECTS	CATS	Non-credit	
Credit value	10	20	N	HECOS codes
FHEQ level	Level 5			

Allocation of study hours

	Hours	
Lectures	40	*Online lectures*
Group teaching	20	*Problem classes*
Lab/ practical	30	*(Not timetabled - estimated)*
Other scheduled		*Incl. project supervision, fieldwork, external visits.*
Independent study	160	*Incl. wider reading/ practice, follow-up work, completion of assess*
Placement		*Incl. work-based learning and study that occurs overseas.*
Total hours	250	
ECTS ratio	25.00	

Project/placement activity

Is placement activity allowed?	No

Module delivery

Delivery mode	Taught/ Campus	Other	
Delivery term		Other	Term 1 + 2

Ownership

Primary department	Electrical and Electronic Engineering
Additional teaching departments	

Delivery campus	South Kensington

Collaborative delivery

Collaborative delivery?	N
External institution	N/A
External department	N/A
External campus	N/A

Associated staff

Role	CID	Given name	Surname
Module Leader		John	Wickerson
Topic Leader		Peter	Cheung

Learning and teaching

Module description

Learning outcomes	Upon successful completion of this module, you will be able to: 1. Describe how high-level programs are executed through the sequencing of instructions 2. Create a compiler from a high-level language to an instruction-based language 3. Program a functional model of a CPU 4. Optimise high-level data structures to exploit the low-level memory hierarchy 5. Design data-structures which can represent programs 6. Determine the worst-case propagation delay of a combinational circuit 7. Design arithmetic circuits to meet a specification and determine the propagation delay 8. Evaluate possible architectural solutions against a set of performance objectives 9. Discuss the relationship between throughput, latency, and pipelining
Module content	- ISAs - State machines - Data and control path - Critical paths and timing - Pipelining - Physical storage - Arithmetic - Buses - RTL (Verilog) - Memory Hierarchy - Grammars - Interpreters - ASTs
Learning and Teaching Approach	You will be taught through lectures in a large lecture theatre to the whole year group using slide presentations, and interactive teaching technology supported question and answer sessions. In tutorial groups of 3-4 students you will have the opportunity to solve a selection of problems with your personal tutor. Laboratory exercises will link the topics taught directly to their applications.

Assessment Strategy	Laboratory exercises are used to develop skills and understanding. There are two courseworks in autumn and spring term which provide an applied context for the ideas. A final exam assesses understanding of the full topic.
Feedback	Feedback will be given on laboratories either orally or through automated tests. Feedback on the coursework will be given after the assessments, both as summative results and as formative feedback. Feedback on the exams will be giving in after re-assessments have been given. This feedback will come in the form of annotated example answers. The annotations will highly where the student cohort did well and where general problems in understanding were observed.
Reading list	Compilers: principles, techniques and tools (Chapter 2).
Required equipment/ software	NA

Quality assurance

Date of first approval	
Date of last revision	June 2026
Date of this approval	
Module leader	John Wickerson
Notes/ comments	

Office use only

QA Lead	
Department staff	
Date of collection	
Date exported	
Date imported	

Template version

Assessment details

Grading method	Numeric	Pass mark	40%

Assessments

Assessment type	Assessment description	Weighting	Pass mark	Must pass?
Examination				
Coursework	Instruction Architecture coursework, autumn (small group)	25%	40%	N
Coursework	Compiler coursework, spring (small group) including a post-submission interview	25%	40%	N
Laboratories	Instructional exercises in SW tools (Spring mid-term)	5%	40%	N
Laboratories	Instructional exercises in HW tools (Autumn mid-term)	5%	40%	N
Examination	exam at start of summer term	40%	40%	N
		100%		