Faculty of Engineering and the Built Environment

Department of Electrical Engineering

2023 GA/ELO Course Handout: EEE4120F as at 1 Mar 2023

Course Name:	**EEE4120F High Performance Embedded Systems**
SAQA Credits:	16 credits
Pre-requisites:	EEE3096S
Co-requisites:	None

Course convenor:	Simon **WINBERG**
Email address:	simon.winberg@uct.ac.za
Office location:	Room 6.13, Level 6, Menzies Building
Consultation hours:	Wed/Thus 10am bookings site: https://www.supersaas.com/schedule/swinberg/Consultation
Course lecturer:	Simon **WINBERG**
Teaching assistant:	Chris **HILL**

Teaching Mode:	Blended and Online-suited – a combination of Differentiated Instruction (using a variety of resources from which to learn) and Expeditionary learning (where students can try different approaches to learn a topic).
Learning activities:	A combination of methods: lecture slides, recorded annotations in lectures / supplementary podcases, example problem-solution scenarios, design case-studies. Week is planned to follow a scenario:
Weekly Routine:	Monday : Update on learning, lessons and assignment plans for the week. 11h00-13h00 : Prac Option 1 (to discuss in Lecture 1) Tuesday : 11h00-13h00 : Prac Option 2 (to discuss in Lecture 1) Wednesday: 14h00-16h00 Lecture session and learning materials explained Thursday: 14h00-15h00 Lecture session / interlude to prac / quiz 14h00-15h00 Prac Session A (main prac session, start of new prac)
Laboratory approach	Blended: Student can choose to either use labs in the department (Blue Lab) or to work on their own computers / Raspberry Pi, or to use a combination of these approaches.
Laboratory assistance	Tutors will aim to be available to assist during scheduled slots using Discord https://discord.com/
Course website	http://ocw.ee.uct.ac.za/courses/EEE4120F/ : learning, lecture, assignment and project resources https://amathuba.uct.ac.za/d2l/home/14448 : submit things here, grades

Course objectives

The objective of this course is to develop an understanding of the concepts involved in the design and development of high performance special-purpose embedded computing systems (or HPES systems), which has become known in the literature as the development practice of "High Performance Embedded Computing" (HPEC). This course incorporates a focus on the development of custom digital accelerators which are becoming more commonly use in high-speed HPEC systems. HPEC is essentially a generation beyond the more traditional subject of embedded system development – and is strongly aligned to recent development trends of university and in industry. With the increase in computing power, available in today's embedded microcontrollers and special-purpose processing system, the design of many modern embedded systems is no longer constrained to single microcontrollers or DSP chips, but are to an increasing extent harnessing the computing resources that are available in new technologies such as FPGAs, Graphics Processor Units (GPUs) and multi-core processors. Your smartphone is probably a case in point, with its multiprocessor and graphics accelerator. Previously (pre +- 2005), embedded systems and embedded software development were considered among the hardest form of computer application design – but the practices of HPEC are disrupting this view. In recent years, the complexity of the hardware and software designs of HPEC systems have surpassed that of the more traditional views of embedded systems. Examples of HPEC systems include software defined radio systems (e.g., telecoms base stations that can change its radio processing methods in software) and radar systems (e.g., array of GPUs doing simultaneous multiband detection).

Learning outcomes

Students successfully completing this course will have the following:	Exit level	LO 1	LO 2	LO 3	LO 4	LO 5	LO 6	LO 7	LO 8	LO 9	LO10	LO11
A. Knowledge (Information plus Understanding)												
1. Parallel computing fundamentals, concept of automatic parallelism; understand the difference between microprocessor-based computer systems, embedded computer systems and reconfigurable computers. Concepts of temporal and spatial programming. Current trend in high performance embedded computing (HPEC).	N									8		
2. Patterns in parallel computing and HPEC design. Knowledgeable of the steps in designing parallel programs (these steps are generic to both microprocessor- and FPGA-based/embedded parallel systems). Understanding of parallel design patterns.	N	8								8		
3. Theories of benchmarking digital systems (incl. speed and power benchmarking). Parallel programming models (data parallel model; message passing model, etc.). Understanding of the common parallel architecture models (including Von Neumann architecture, Flynn's taxonomy, etc.), memory access architecture models. Can draw on knowledge of case studies to illustrate some of these architecture models.	N	8	8							8		
4. Knowledge of computer subsystems, including DMA, bus management systems, memory types and memory management systems and interrupts among others. Fundamentals of microprocessor-based and FPGA-based reconfigurable computing. Knowledge of the fundamental reconfigurable computing building blocks. Understand theory and limitations of using an automatic C -> HDL conversion tool to translate a specialised form of C code to HDL.	N	8	8							8		
B. Skills (Application of Knowledge)												
1. Can develop parallel / threaded programs in C or C++, able to compile and run this on a standard digital computer architecture. Able to translate a complex sequential algorithm into a parallel version, and can implement and run the resultant algorithm on a computer cluster using e.g. MPI. Use of heterogeneous computer, comprising e.g. multicore standard architecture and Programming GPU running highly parallel code. Can design and code FPGA-based firmware suited for deployment on a reconfigurable computer platform (i.e., towards prototyping custom HPEC system).	N	8	8									
2. Benchmarking of parallel computer systems (microprocessor-based systems and systems making use of GPUs). Benchmarking embedded high performance computing systems (knowledgeable of how to do so for microprocessor computer systems and system incorporating FPGA-based or GPU-based accelerator module).	N	8	8									
3. Ability to describe HPEC design ideas. Can describe (in writing with figures) how to go about designing a HPEC system, or HPEC subsystems, that solves an open-ended application development problem. (Tested in Conceptual Assignment).	Y	8	8	8								
4. Able to validate and verify digital system design. Can use correlation to compare results produced by different platforms. Able to do performance analyses on the design of HPEC systems including analysis of communication performance between computer systems. Given a detailed design of a HPEC system, the student can perform paper-based calculations to estimate the performance of the HPEC system. Use of this data to identify performance bottlenecks, areas to optimise design costs, amongst suggesting possible other improvement to the given HPEC design.	Y		8									
5. Able to understand and describe the principle characteristics of a digital system, its overall design process, design software and hardware for such systems.	N			8			8					
C. Values and Attitudes												
1. Appreciation of digital system / HPEC design to support real time applications	N							8				
2. Stimulate an interest in this branch of Electrical Engineering / Computer Engineering.	N							8			8	
3. Able to contend with open-ended problems requiring thinking and intuition.	N											
4. Appreciate the need of teamwork in digital systems design and implementation.	N								8			

The course addresses the following Graduate Attribute exit level outcomes

Main outcome	Activities, skills or abilities contributing to the main outcome/objective	Alignment of assessment with relevant GA description
Graduate Attribute (GA) / Exit-level Outcome 2: **Application of scientific and engineering knowledge** Apply knowledge of mathematics, natural sciences, engineering fundamentals and an engineering specialty to solve complex engineering problems.	1. A systematic, theory-based understanding of the natural sciences applicable to the discipline; 2. Conceptually-based mathematics, numerical analysis, statistics and formal aspects of computer and information science to support analysis and modelling applicable to the discipline; 3. A systematic, theory-based formulation of engineering fundamentals required in the engineering discipline; 4. Engineering specialist knowledge that provides theoretical frameworks and bodies of knowledge for the accepted practice areas in the engineering discipline; much is at the forefront of the discipline. 5. Mathematics, natural science and engineering sciences are applied in formal analysis and modelling of engineering situations, and for reasoning about and conceptualizing engineering problems.	1. Students are tasked with understanding the concept of natural phenomenon provided in the GA2 Conceptual Assignment description. The student is provided readings and weblinks to help them assimilate the concept explained in this reading assignment. The student's comprehension of this phenomenon is assessed in the '**GA2: Comprehension Test**' based on Pracs 1 and 2, which has a time-limited test with an essay component. 2. The student is then tasked with '**GA2 Problem Solving Questions**' that lead from the GA2 Concept Assignment description, in which the natural phenomenon and concept presented for point (1) above is formulated into a slightly more refined, but still (ill-formed) product description presented to the student. The student then draws on engineering knowledge, maths and understanding of natural science and scientific method, to suggest a more refined 'proposed implementation strategy' for building and testing the product. Note: this proposal is only expressed in a document (not implemented or needing executable code provided). Part of this document is to explain how this solution can be benchmarked and validated (e.g., by model testing or setting up a 'golden measure'). *{ALTERNATIVELY:* Students may instead be asked to critique a small, ill-formed design concept (a case study). The student will need to use a scientific approach to report on their approach the anticipated performance of their solution to validate its operation (note both alternatives are more conceptual activities, not necessarily requiring complete implementation and testing of a developed prototype).} 3. Students will be required to complete the '**GA2: Validation Essay**' used o validate the student's understanding of their solution and approaches used in #2 above (*Note:* this essay test is to ensures students understood and applied themselves in parts 1 and 2 above; students that did not engage in part 2 particularly (e.g., relying on classmates to do all the work and explain what would do) are unlikely to pass this test which would result in a DPR).

Information specific to Graduate Attributes

GA Outcome 2:
Where and how is this learning outcome assessed?
Students will be individually evaluated in tests forming part of the **GA2 Conceptual Assignment** described above; these will evaluate each student's ability in the application of scientific and engineering knowledge concerned in this assignment.
What constitutes satisfactory performance?
Passing the GA2 assessment test. Students will need to pass both the 'GA2: Comprehension Test' (i.e., knowledge of the phenomenon and design approach) and 'GA2: Validation Essay' (i.e., understanding of method and tools to develop and validate a design solution). To pass the GA2 requirement (and receive a DP for this course), the student must achieve a mark of 50% or more for each part of the GA2 Conceptual Assignment. For any part that is failed, a second opportunity will be provided to pass that part but the recorded mark use in calculating the coursework grade for the course will use the mark obtained for the first attempt. Student that do not complete one of the parts will fail the assignment (and receive a DPR) unless they provide a valid excuse for having not completed the part concerned.
What strategy is to be followed should this learning outcome not be satisfactorily attained?
Students are allowed a maximum of **two** additional attempts or re-submissions of work to meet the GA, failing which they get a Duly Performed Refusal (DPR), and cannot write the course examination.

Detailed course content

The course is divided into two parts: Part A and Part B. Part A runs in the first term, and Part B runs in the second term; these are explained below:

Part A: Microprocessor-based parallel computing and theory with abstract study of special-purpose CPU/processor/SoC design. Three practicals related to parallel computing and the 'conceptual assignment' mini-project concerning partial design and theoretical analysis of a special-purpose HPEC design concept.

Part B: Reconfigurable computing and advanced theories of distributed and heterogeneous computing ecologies (including Apache Hadoop). FPGA up-skilling, and the design and development of digital logic to implement a digital accelerator (or special purpose digital SoC systems) running on FPGA platforms (these digital accelerators could also be considered to prototype functionality aspects of digital SoC system, albeit at a much larger scale, and slower performance, that what can be achieved on a silicon wafer). Considering the increasingly many solutions for HPEC development, as well as the increasing class size, this second term of the course has been adjusted to be less HDL- and FPGA-dominated and to rather provide students more variation in tools they choose to use; however a clear consistency is maintained in terms of the developing of an application accelerator as part of a component of a HPEC system, but instead of requiring all students to develop a FPGA-based accelerator, students will be given alternates such as using a GPU instead of a FPGA. All students will nevertheless learn Verilog HDL coding, building on the basic HDL introduction in Embedded Systems II (EEE3096S), to implement and test more complex HDL solutions.

We have been trying to move this course away from a lecture-dominated (transmission) approach, towards a more "expeditionary learning", where the students are provided a curated collection of learning resources, instructional guidelines on assimilating and learning from these resources, tasked with practical assignments and problem-solving tasks to gain experience in using tools for solving representative problems and building design understanding in this subject. The course has accordingly been divided into a series of topics, a new topic generally starting each week, for which the associated resources will be released, guidelines for learning from those resources will be provided, and opportunities for posing questions, or discussing solutions will be provided. Quizzes will be held to encourage students to keep up with the learning topics and to help them to assess their performance in the course. While the quizzes will generally be for marks (to encourage participation), these quizzes will have only a small weighing in the final grade.

During the first term (Part A) students focus on fundamentals of parallel computing architectures and parallel programming. During this term students focus on laboratory pracs and GA2/homework tasks (reading and possible short assignment/comprehension task). Into the third week of Term 1, the GA2 Conceptual Assignment and its associated reading task will be released. This conceptual exercise, as the name suggests, is largely a thinking exercise and high-level design conceptualization; the student's understanding of this is assessed in the first GA2 Test.

During the second term (Part B) the course has an emphasis on the use of field programmable gate arrays (FPGAs) and HDL programming in relation to design and application development for ReConfigurable (RC) hardware platforms. RC platforms, and representative case studies, are discussed in the learning resources, and in the pracs, prepared as platforms that provide a means to accelerate computation, or to offload computation from a host CPU. Students' understanding of HDL coding, hardware/software peripheral interfacing and other more embedded software related development issues are built up, to develop an understanding of using HDL to design a hardware accelerator component or specialized peripheral, that can assist the PC/host computer in the completion of tasks. These theories lead into the course project, which is called the 'Your Own Digital Accelerator' (YODA) Project, and is worked on by a team of two or three students. This purpose of the YODA project is to integrate the student's understanding and experiences of HPES learned in this course to build and test the YODA device, which is a prototyped system that incorporates an accelerator component (e.g., running on a FPGA platform or a GPU) that connects with a host-based program (note, this can be designed around use as a hardware-in-the-loop type approach where a real accelerator is used, OR it can be designed to run in a simulation environment, such as running more in 'slow time' on an accurate simulator as opposed to running in real-time). Besides HDL and other software development and simulation, the project touches on issues of problem description, problem solving, design and implementation, hardware/software interfacing, communications protocol design, experimentation and culminates in a final acceptance testing demonstration.

Outline of course structure
The course structure is shown on the next page. While it is generally somewhat consistent between versions of this course, the specific topics covered may change a bit each year.

Assessment Strategy for Distance Version

The assessment strategy followed in the distance version of this course is going to have some similarities to that of the on-campus version. In particular, we are aiming to have the following assessments:

Assignment	*Weighting*	*Details*
Practicals	15%	3 practicals, bridges to FPGA or OpenCL project-based learning PbL activities)
Tests	18%	4%: spot quizzes / participation marks for activities during lecture slot; 7% : Test 1 including GA2 validation test 7% : Test 2 (preparation for final exam)
GA assessment	2%	Comprehension Test. Conceptual assignment. These & validation test are DP requirement for EEE4120F.
Project	15%	Involves milestones: status update, demo, final report
Final exam	50%	Planned to be a venue-based closed-book examination

In addition to the above, we will provide learning activities to help encourage participation, and assist in self-learning of lecture topics and test preparation. Students are all encouraged to participate in spot quizzes and activities that make take place during lecture slots, and make use of Discord to ask questions of tutors / 'project managers'.

Outline of course structure

Term 1:

1. Intro to HPES and Reconfigurable Computing (13-17 Feb)
- Welcome lecture and explanation of the course
- Terminology (e.g. Golden Measure; Spatial vs. Temporal computing)
- Reading: R01 'Landscale of Parallel Computing Research'
- *Prac1:* Testing methods: OCTAVE intro and creating golden measure

2. Edge, Tools to Used, Intro Benchmarking (20-24 Feb)
- Considerations for EDGE computing & relevance of HPES to this
- Microprocessor-based vs. FPGA-based solutions for RC
- Platform & tools we will use
- GPUs (issues and benefits) and programming of these *
- OpenCL and application to many-core / GPUs programming
- Quiz 1 (Thursday 2pm, 15min)
- Performance benchmarking essentials: wall clock time
- Useful Techniques Selected topics on timing in C or other languages.
- *Prac2:* GPUs OpenCL experience, sequential vs many-core parallel
- Due: Prac 1

3. Parallel programming & Performance Benchmarks (27 Feb – 3 Mar)
- Performance benchmarking; Size, Weight and Power benchmarking
- Profiling code (*Reading:* Valgrind)
- Parallel computing fundamentals
- Large Scale Parallelism
- Important terms (contiguous, partitioned, interleaved, interlaced)
- Amdahl's Law for the Multicore Era
- Base Core Equivalents (BCE).
- Reading: Hill & Marty's "Amdahl Law in Multicore Era.pdf"

4. Design of Parallel Systems and Programs (6-10 Mar)
- Out: scientific GA2 reading
- Steps in Designing Parallel Computing (steps 2 and 3 of 3…)
- Partitioning, Decomposition & Granularity, Communications
- Parallel computing essential theories
  - Identify data dependencies
  - Synchronization
  - Load balancing, performance analysis & tuning
  - Parallel prog models: Data parallel; msg passing; shared mem; hybrid
- Quiz2 : assessing Hill & Marty's "Amdahl Law in Multicore Era.pdf"
- Automatic parallelism;

5. Communication and memory architecture (13-17 Mar)
- Cost of communication (Design of parallel systems – step 2 of 3)
- Latency, bandwidth, effective bandwidth (and related calculations)
- Blocking/non-blocking; synch/asynch; scope of communications
- Memory access architectures (intro to this, more in Weeks 10, 11)
- *GA2 Comprehension Check (attempt 1, 16 Mar 2pm venue: EM6)*

6. Verilog Recap & Vivado Intro (recap of EEE3096S) (20-24 Mar)
- YODA project begins!
- Presentation of the YODA (Your Own Digital Accelerator) project
- Whirlwind tour of some YODA topics
- Call for proposals for innovative YODA projects
- Prac3: Simulation-based IP Core Integration Using Xilinx Vivado (prac does not need FPGA kit, just uses simulator)
- Recap programmable logics, Verilog HDL and VHDL
- The lecture delves into syntax and simple examples of using Verilog
- The open-source Icarus Verilog simulator and Xilinx QSim is used.
- Leading on to FPGA application coding.
- Using IP cores / block design / adding IP in Vivado designs
- Advantages of knowing both VHDL and Verilog
- Prac4 (optional) reflections on its relevance to these topics
- Due: Prac 2 (27 March 2023)
- Due: YODA MS-0: Group formed (*choose topic by 31 Mar*)

Term 2:

7. Test 1, catchup and YODA project planning (3-7 Apr)
- Class Test 1 Including attempt 2 for *GA2 comprehension check*
- Due: Prac 3 (7 April 2023)

8. More Verilog HDL and Simulation Techniques (10-14 Apr)
(GA additional attempt) – 13 Apr 3pm
- Due: YODA Proposal Blog (12 Apr)
- Various HDL techniques (synch vs asynch, pitfalls, sequential modules)
- State-machines in Verilog
- Overview of the variety of FPGAs and tools.
- The Nexys FPGA platform (example embedded Xilinx FPGA platform)
- Licensing of tools & IP
- Practical methods: ChipScope and brief overview of other tools
- HDL Imitation Method. Relevance Amdahl's to FPGA (see also BCE).
- Simulation methods for brining in real-time and physical characteristics

9. FPGA systems and related architecture (17-21 Apr)
- Due: YODA Status Update (this week by 21 Apr)
- FPGA Performance benchmarking techniques
- Memory Control Units (part2/2), FIFOs and LIFOs in Verilog
- Some useful operators. Doing math, developing a multiplier.
- Workings of a PLL
- Theoretical speed calculation of a logic design.
- Comparing FPGA-based execution time to CPU execution time.

10. Config architecture, Interconnects, Memory (24-28 Apr)
- Configuration HW architectures (where is the bitstream stored etc)
- Robust interfaces and handshaking
- Designs concerns, the 'design space' & various (real-world) vignettes (changes somewhat each year)
- Interconnection on-chip bus topologies (extends 'handshaking')
- Memory types, Memory Control Units (part1/2), MCU in Verilog
- Due: YODA Draft Paper (by 28 Apr)

11. YODA Conference & Demos Week (1-5 May)
- Data Path, Instructure Path
- Cache Hierarchy
- Shared/Not Memory
- Instruction Set Arch, micro instructions
- Processor aspects - e.g. CISC vs RISC (*Reading:* RISC-V scenarios)
- aim to end main content lectures 5 May

12. Foundational Computer Architecture (8-12 May)
- YODA Conference
- Due: YODA MS-4 / conf. presentation by 12 May

13. More HDL methods; RC Design Process (15-19 May)
- Due: YODA MS-5: Final Project Report & Code
- *additional/advanced lecture topics:*
- Dealing with clock boundaries
- Latest Trends in FPGAs.
- Benchmarking using DMIPS, Dhrystone, Whetstone, Coremark
- Virtualization and other key technology factors

19 May: last teaching day of term

- *optional*
  - FPGA External Mem. Direct Memory Access (DMA) principles
  - On-chip Interfacing Standards (Wishbone, AVALON)

OPTIONAL:

14. Softcores & RC Design Process
- The FPGA Design Cycle
- Recap of design cycle and HW/SW co-design
- Quality assessment and measurement {towards HDL Kaizen}
- optional: YODA MS-3: Draft paper
- optional in 2021:
  - Methods for RC application development
  - Softcore & uCLinux O/S running on an FPGA platform
  - C→HDL automatic conversion (HandleC notation, how to write a C-style representation of a digital logic circuit).
  - Use of lookup tables, megablocks etc. (bulky and boring section for most students, will probably be left out)
  - Sourcing Intellectual Properties - *optional*

Test/Project Dates

Quizzes and Worksheets	
~~These will be announced a week before they are released (removed for 2022).~~ Worksheets instead moved to GA2 assessment.	
GA2 Comprehension Test (11 Mar, online) **Class Test 1 including GA2 Validation Test** (5 April, 1.5h, regular double lecture time 2pm in venue EM6)	
Final Exam	**TBD -** waiting to receiving exam timetable
Project *(note: can choose to submit milestones early if you choose, when relevant assignment opens. Late penalties: see rule below.)*	
Milestone 0: Group formed and topic Selection [0%]	**23 Mar**
Milestone 1: Proposal Blog [5%]	**12 Apr**
Milestone 2: Status Update [5%]	sometime in week **17-21 Apr**
Milestone 3: Draft paper [5%]	**28 Apr** *(note this submission is not optional this year, is for marks)*
Milestone 4: Demonstration / (Alpha type) Acceptance Test [20%]	**8-12 May** *(conference presentation, all team members to present)*
Milestone 5: Final Project Report [65%] (Code Hand in optional)	**15 May (submit by 23h55)**
* The final double period or two was previously set aside for the "YODA Conference" where each team is given (about 10min) to do a brief presentation on their project topic (counts as YODA presentation marks). This was restructured as Video Recordings that each YODA team prepared and upload to Vula by the due date (to be announced). Each student should look at two or more demo videos submitted by other students (who are not in the same YODA team). I'm considering making it more of a Vlog, where students can possibly use the Vula Blog add some comments other materials together with a link to their video, possibly also posting a follow-up short vlog to answer any questions that the markers may have.	

Knowledge areas

Maths Sciences	Natural Sciences	Eng Sciences	Design & Synthesis	Complm Studies
0	0	**50**	**50**	0

Learning environment

Multi-modal, online learning supported. Students will be expected to engage with material in various forms in a variety of different ways.

Suggested time allocation

Learning Activity	No./ week	Time in hours	Contact time Multiplier	Total no of hours
Number of lectures per week	3	0.75	2	54
Number of tutorials per week	---	---	---	---
Total practical/lab periods	3 pracs	3.3	3	30
Total other contact periods	---	8	---	8
Total assignment non-contact hours (Project)	---	40	---	40
Assessment hours (Quizzes/Worksheets; Exam)	---	7	4	28
Number of weeks the course lasts	12	---	---	---
Total hours	---	---	---	**160**

General assessment strategy

Assessment Task	%	The following DP rules and late penalty rules apply:
Labs	15	The following DP rules apply: 1. Minimum **40% overall class average** to write the final exam. **2.** Engineering students required to **pass ECSA GA/ELO 2** **Late penalty for assignments:** 10% a day, up to a max of -50% (no >5 days late)
Project	15	
Tests (quizzes [8%] and class test [10%])	18	
GA2 Conceptual Asg.	2	
Exam	50	
Total	**100**	

Books/Reading Materials/Notes

- A selection of prescribed academic articles (journal papers and peer reviewed conference papers) will augment the text-book (these papers or links to them will be made available on VULA or linked to on the open courseware website)
- Public / OpenUCT website: http://ocw.ee.uct.ac.za/courses/EEE4120F

Absence: The continuous assessment marks will be adjusted to allow for absence only on the following grounds:

- A medical certificate for absence of a class test or exam
- Death of an immediate family member (parent or sibling)
- Pre-arranged absence to represent a University, provincial or national team.

Academic dishonesty: Plagiarism is a very serious offence and usually leads to disciplinary action that could include expulsion from the university. Therefore, recognise the work of others in any submission. Details of referencing methods are widely available on the Web. A non-plagiarism declaration must be submitted with all work submitted for marking.