Faculty of Engineering and the Built Environment

Department of Electrical Engineering

2023 GA Course Handout: **EEE3096S**

Course Name:	**EEE3096S EMBEDDED SYSTEMS II**
SAQA Credits:	16 (18 for EEE3095S)
Pre-requisites:	EEE2046F (Embedded Systems I)
Co-requisites:	None

Course convenor:	Dr Yaaseen Martin
Email address:	yaaseen.martin@uct.ac.za
Office location:	Room 4.08, Menzies Building
Consultation hours:	Email to request a consultation
Course lecturer:	Dr Yaaseen Martin and Mr Stanley Mbewe
Teaching assistant:	Mr William Bourn (brnwil012@myuct.ac.za)

Lecture venue:	Beattie B114
Lecture days and time:	**Lectures days:** Tuesdays, Wednesdays at 14h00 – 15h45 Four lecture slots per week, but one of these will likely be used as a tutorial/practical/study/consultation slot every week as based on student needs and course content.
Laboratory venue	**White Lab, Menzies Building**
Laboratory days and times	ME students sign up for Lab Groups 1 or 2 (Monday 14h00 to 16h00 or 16h00 to 18h00) EC students sign up for Lab Groups 3 or 4 (Tuesday 16h00 to 18h00 or Wednesday 16h00 to 18h00) Tutors will be available in laboratory venues for help in completing the practicals. Completion of practicals must be demonstrated to tutors on-campus during these sessions. An announcement will be made to indicate when a new practical is released.

Course objectives

This course integrates aspects of embedded systems and computer architecture by building on EEE2046S (2nd year Embedded Systems 1), which provided introductory knowledge of digital electronics, microcontrollers, and C programming. Embedded Systems 2 introduces students to: microprocessor architecture fundamentals, embedded operating systems architectures, Linux, memory technologies, a bit of Hardware Descriptive Languages (HDL), and common embedded communication protocols. The course additionally expands students' knowledge of: Assembly and C for embedded systems, digital components including memories and timers used in cyber-physical interfaces, and theories related to Analog to Digital Converters (ADCs), Digital to Analog converters (DACs) and Hardware/Software Interfacing.

Practicals will involve using a single board computer embedded platform (the "UCT STM" board), and include work in benchmarking, simulation, programming, and cyber-physical system design. The course is broken into various modules and will include four practical preparatory tutorials and four practical exercises. There is also a project released in the fourth term that will need to be completed by all Computer Science students (EEE3095S) to account for the two additional credits for that version of the course. The recommended textbook for this course is: "Computer Organization and Design ARM Edition: The Hardware Software Interface" (The Morgan Kaufmann Series in Computer Architecture and Design) 1st by Ed. D. Patterson and J. Hennessy (available on Amazon at https://www.amazon.com/Computer-Organization-Design-ARM-Architecture/dp/0128017333)

Learning outcomes

Students successfully completing this course will have the following:	Exit level	LO1	LO2	LO3	LO4	LO5	LO6	LO7	LO8	LO9	LO10	LO11
A. Knowledge (Information plus understanding)												
1. Knowledge of the design lifecycle of an embedded system	Y			X		X						
2. Knowledge of computer and microprocessor architectures pertinent to use in low-power, wireless, and or otherwise edge devices	Y			X		X						
3. Knowledge of an operating systems architecture	N					X						
4. Knowledge of techniques for the analysis of embedded software designs and performance	Y			X		X						
5. Knowledge of communication and interfacing techniques for embedded systems	Y					X						
6. Knowledge of current changes and trends regarding how embedded systems are being used in a wide range of sectors of society	Y					X		X				
7. Knowledge of how ADCs and DACs function and their use in embedded systems.	Y					X						
B. Skills (Application of knowledge)												
1. Modelling embedded system operation	Y		X	X		X						
2. Algorithmic state machine design	Y		X	X		X						

3. Theoretical and practical techniques to model and analyse the performance of computer architecture and embedded systems	Y		X	X		X						
4. Configuring a Linux kernel	N		X			X						
5. Use of a cross-compiler	N		X			X						
6. Practical use of standard embedded systems communication protocol	Y		X			X						
7. Debugging techniques for embedded systems	N		X			X						
8. Designing gateware for programmable logic devices / FPGAs	N		X			X						
9. Using HDL synthesis & analysis dev. tools (e.g. Quartus II or Vivado)	N		X			X						
11. Simulation & benchmarking techniques, use HDL simulator (e.g. iVerilog)	N		X	X		X						
12. Understanding and writing basic assembly language programs	N		X			X						
C. Values and attitudes												
1. Recognition of characteristics of quality embedded system designs	N											
2. Importance of design documentation	N			X			X					
3. Value of thorough testing .	N			X								
4. Software engineering design principles for real-time systems	N			X								
5. Appreciation of the value of teamwork in embedded system design & implementation.	N						X		X			
6. Appreciation for the use of appropriate tools for collaborative and complex design work	N						X		X			

The course addresses the following Exit Level Graduate Attributes

Main graduate attribute	Activities, skills or abilities contributing to the main outcome/objective	Alignment of assessment with relevant GA descriptions
Exit level outcome 5: Engineering methods, skills and tools, including Information Technology Demonstrate competence to use appropriate engineering methods, skills and tools, including those based on information technology.	***GA tests being reworked to streamline the process*** *A. Knowledge (Information plus Understanding)* A.1: Understanding of fundamental tools used in the design process for Embedded Systems A.2: Understanding of computer and microprocessor architectures pertinent to use in low power, wireless, and or otherwise edge devices. A.3: Knowledge of techniques for the analysis of embedded software designs and performance.	**A1** Attempt 1: Prac 1 mark Attempt 2: Prac 2 mark Attempt 3: Prac 3 mark **A2** Attempt 1: Prac 1 mark Attempt 2: Prac 2 mark Attempt 3: Prac 3 mark **A3** Attempt 1: Prac 1 mark Attempt 2: Prac 2 mark Attempt 3: Prac 3 mark

	B. Skills (Application of Knowledge)	
	B.1 Embedded System design as a complex process.	**B1** Attempt 1: Class Test 1 Attempt 2: Separate GA test
	B.2 Use of design diagrams, including Finite State Machine (FSM) & Algorithmic State Machine (ASM).	**B2** Attempt 1: Class Test 1 Attempt 2: Separate GA test
	B.5 Use (cross-)compiler for developing ES software.	**B5** Attempt 1: Prac 1 mark Attempt 2: Separate GA test
	B.6 Use of standard embedded systems communication protocols.	**B6** Attempt 1: Class Test 1 Attempt 2: Project mark or Separate GA test

IMPORTANT: All EEE students (EEE3096S) are required to pass each of these GAs in order to obtain DP for the course. CS students (EEE3095S) will also be assessed on all of these GAs but are **not** required to pass them to obtain DP.

For EEE students, the recommended approach is to:

- Pass Practical 1 (covers attributes A1, A2, A3, and B5)
- Pass each part of Class Test 1 (a written GA test that covers attributes B1, B2, and B6)

If any of these attributes are **failed** by a EEE student, they will be given a **second attempt** to pass their missing GAs either through a follow-up written GA test or one of the subsequent practicals. By the time the DP list is released, every EEE student must have passed all of the above GAs or they will **not be allowed to write the final exam**.

Detailed course content

Summary: As described in the objectives, the course extends students' knowledge in design and use of embedded systems by introducing new theory, expanding their knowledge in range of areas, and providing a learning environment within which to practically apply both theory and knowledge. The course is delivered in multiple modules, each focused on a core theme and separated between the two terms, Term 3 and Term 4, as follows:

- Term 3: Lectures cover cyber-physical systems, communication protocols, embedded system optimisation, and ARM architectures, and C and Assembly coding. Two practicals will be released in this term and will cover some of the applications of these tools, including benchmarking, emulation, embedded communications, and basic input-output interfacing with the STM board.
- Term 4: Lectures focus on ADCs, DACs, CPU architecture, Verilog, state machines, and interrupts, with practicals making use of these aspects and covering: ADCs, PWM, DACs, LCD interfacing, GPIO circuitry, and interrupts.

Details of the six learning modules of this course is provided below:

1. **Embedded C and debugging**
    a. Cyber Physical systems (CPS)
    b. Real-time systems (RTS)
    c. Development & Execution Environment
    d. Embedded C
    e. Make and compiler flags (cmake optional reading)
    f. GDB Debugging and using Bash
2. **ARM, Assembly, Optimization and Benchmarking**
    a. Intro to ARM. RISC vs. CISC processor architectures.
    b. ARM Core Families.
    c. Modes of operation, ARM registers
    d. ARM instructions, Exceptions and Branching
    e. Intro to ARM assembly language coding.
    f. Mixing assembly and C code
    g. Stack, Special Registers, Calling conventions
    h. Compiler Optimization and GCC opt. flags
    i. The Heap and related functions
    j. Handling exceptions and interrupts in assembly and C
    k. Benchmarking and Code Reviews
3. **Embedded Communications**
    a. STM architectures (and case study of Raspberry Pi and)
    b. I/O and communication in industry
    c. SPI, I2C
    d. UART, USB, JTAG
    e. Timers, EEPROM
    f. PWM, debouncing
    g. An embedded OSI stack
4. **Sampling & ADCs, Signal Generation and DACs**
    a. Basics of timers
    b. Information theory and DAC signal generation
    c. ADCs sampling
    d. ADC metrics
    e. Exceptions and interrupts
    f. Filtering
5. **Introduction to Verilog**
    a. Introduction to Verilog
    b. Types of FPGA
    c. Mealy and Moore HDL FSMs
    d. Recap on Adders, Comparators, FFs
    e. Recap on Registers, Shifters, Memory Technologies
    f. Cache hierarchy
    g. Shared memory
6. **Computer Architecture** (time permitting)
    a. CPU Architecture types
    b. ALU and data path
    c. Data and Control path

Knowledge areas

Maths Sciences	Natural Sciences	Engineering Sciences	Design & Synthesis	Complementary Studies
-	-	60	40	-

Learning environment

Multimodal. Combination of laboratory work involving hardware, lecture sessions, and course material (lecture slides, links to other learning resources, etc). Students will be expected to engage with material in various forms in a variety of different ways, but have much flexibility in where and how this material is accessed.

Suggested time allocation

Learning Activity	**No./ week**	**Time in hours**	**Contact time Multiplier**	**Total no of hours**
Number of lectures per week (for 12 weeks)	2	3	2	72
Number of tutorials per week (for 12 weeks)	0.33	2	3	24
Total practical/lab periods (for 12 weeks)	0.33	2	5	40
Total other contact periods				
Total assignment non-contact hours				15
Assessment hours (Tests, Exam)		5	2	10
Number of weeks the course lasts (extra week for revision, extensions, public holidays)	13			
Total hours				**161**

General assessment strategy

Assessment Task	%	The following DP rules apply:
Tutorials	10%	Completion of all tutorial and practical assignments. All EEE students also need to pass all ECSA Graduate Attributes. **Late penalties will be applied for assignments.**
Labs	20%	
Class Tests	20%	
Exam	50%	
Total	**100%**	

Books/Reading Materials/Notes

Prescribed textbook: “Computer Organization and Design ARM Edition: The Hardware Software Interface” (The Morgan Kaufmann Series in Computer Architecture and Design) 1st Edition. Authors: David A. Patterson and John L. Hennessy.
On Amazon at: https://www.amazon.com/Computer-Organization-Design-ARM-Architecture/dp/0128017333

Supplementary (optional) textbook: “Embedded System Design: Embedded Systems Foundations of Cyber-Physical Systems, and the Internet of Things” 3rd Edition by Peter Marwedel. Publisher: Springer. 2018.

Online slide / additional notes available on Amathuba

Amathuba: See the EEE3095/6S site

Absence: If a class test is missed, the final exam will be scaled up in its weighting to accommodate this change. E.g. missing a class test that counts for 10% of the final mark will result in the student's exam being weighted 10% extra in the final mark.

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.