UNSW SYDNEY

Course Outline

May 2023

STEM Diploma

DPST1092

Computer Systems Fundamentals

1. Staff

Position	Name	Email
Course Convenor & Lecturer	Dr Angela Finlayson	A.Finlayson@unswcollege.edu.au

2. Course information

Units of credit (UOC): 6

Pre-requisite(s): CP1511

Total course contact hours: 96

2.1 Course summary

This course provides a programmer's view on how a computer system executes programs and manipulates data. It enables students to become effective programmers in dealing with issues of performance, portability, and robustness.

The course assumes that students have completed the first course in programming in the C programming language, CP1511.

2.2 Course aims

This course aims to give students an overview of the structure and behaviour of modern computer systems, and to provide a foundation for later courses on operating systems, computer architecture and compilers, where a deeper understanding of systems-level issues is required.

2.3 Course learning outcomes (CLO)

At the successful completion of this course you (the student) should be able to:

1 Describe the layers of architectures in modern computer systems from hardware device levels upwards

2 Describe the principles of memory management and explain the workings of a system with virtual memory management

3 Explain how the major components of a CPU work together, including how data (including instructions) are represented in a computer

4 Design, implement and analyse small programs at the assembly/machine level

5 Describe the relationship between high-level procedural languages (e.g., C) and assembly/machine language in the conventional machine layer, including how a compiled program is executed in a classical von Neumann machine

6 Explain how input/output operations are implemented, and describe some basic I/O devices

7 Describe the components comprising and the services offered by an operating system

8 Implement simple programs involving inter-process communication and concurrency

2.4 Relationship between course and program learning outcomes and assessments

Course Learning Outcome (CLO)	**Program Learning Outcome (PLO)**	**Related Tasks & Assessment**
CLO 1	Understanding of underpinnings (EA1.1)	Exam
CLO 2	Understanding of specialist bodies of engineering knowledge (EA1.3)	Exam, Labs
CLO 3	Understanding of specialist bodies of engineering knowledge (EA1.3)	Exam, Tests, Labs
CLO 4	Application of established engineering practice (EA2.1)	Labs, Tests, Assignments
CLO 5	Conceptual understanding of computer underpinnings (EA1.2)	Exam, Tests, Labs
CLO 6	Understanding of specialist bodies of engineering knowledge (EA1.3)	Exam, Labs
CLO 7	Understanding of specialist bodies of engineering knowledge (EA1.3)	Exam, Labs
CLO 8	Application of established engineering practice (EA2.1)	Exam

3. Strategies and approaches to learning

3.1 Learning and teaching activities

This course involves a number of teaching activities:

Lectures – 4 hours per week (weeks 1-6 and 8-12)

Lectures present theory and concepts, by way of case studies and practical examples. Lecture notes will be provided in advance of each class. There will be 4 hours of timetabled live streamed lectures each week.

Tutorials – 2 hours per week (weeks 1-6 and 8-12)

Tutorials allow students to collaboratively work through example problems to illustrate lecture ideas, and have concepts from lectures clarified by the tutor. Each student has an opportunity to lead a "code review" (discussion of a piece of software or a system) at least once during the course. There is up to 1 bonus mark available for students who give an outstanding code review.

Lab Classes – 2 hours per week (weeks 1-6 and 8-12)

Lab Classes involve small exercises where students build systems that illustrate the ideas covered in lectures.

To obtain a mark for a lab exercise you should submit it using give.

If you don't finish it during the lab, you may continue working on it during the week, you must submit it (using give) by the following Tuesday 11:59pm in order to get any marks for it.

You cannot obtain marks by e-mailing lab work to tutors.

During the lab, your tutor will give you guidance and can provide feedback on your approach to the problem and on the style of your solution. Some labs may contain exercises which will be assessed during the lab.

The lab exercises for each week are worth in total 2 marks. We will sum to best 9 lab marks to give you a mark out of 18; if their sum exceeds 15, your total mark will be capped at 15.

Most labs include one or more challenge exercises. Challenge exercises may involve concepts not covered in lectures and they range in difficulty from not-very-hard to almost-impossible.

The contribution of challenge exercises to lab marks will be limited to 20%; hence you can obtain nearly all (over 95%) marks available for the lab component without completing challenge exercises.

If you wish to obtained a high mark for DPST1092, attempting some challenge exercises is highly recommended. If your goal is just to master the core material and pass DPST1092, you can ignore challenge exercises.

Assignments – 2 during the course

Assignments are take-home problems that are larger in scope than Lab exercises and require students to use creativity to solve a challenging realistic problem. Each assignment requires students to understand the problem, design a solution, and implement and test their solution.

Weekly Tests– 8 during the course

There will be weekly tests from weeks 3-6, 8–11 designed to give you timely and realistic feedback of your understanding of the course material.

These will be conducted in your own time under self-enforced exam-like conditions. Each test will specify the conditions, but typically these will include:

no assistance permitted from any person;

a time limit;

no access to materials (written or online) except specified language documentation or man pages.

There will also be a weekly test run in one of the lecture time slots in week 12.

Each test is worth 1.7 marks, and will be automarked. Your total mark for the tests component is computed as a sum of your best 6 of 9 test marks.

Any violation of the test conditions will result in a mark of zero for the entire test component.

Apart from the week 12 test, weekly tests will be made available at 9am on the Thursday of the given week and will be due at 9am on the Thursday of the following week.

Online Forum

An online forum on teams allows students to ask and answer questions on the tutorial, lab and assignment exercises, and on lecture material.

Final Exam

There will be a final exam during the exam period. You will be required to attend the exam Face to Face if you are in the country.

It may contain a mixture of: implementation tasks (which will require you to write C and assembler programs); "theory" questions (which require analysis and written answers); multiple choice questions. During this exam you will be able to execute, debug and test your answers. The implementation tasks will be similar to those encountered in lab exercises and weekly tests.

There is a hurdle requirement on the final exam. If you do not score at least 40% on the exam, you cannot pass this course. If your overall course score exceeds 50%, despite scoring very poorly (<40%) on the exam, the hurdle will be enforced via a grade of UF. Of course, if your overall course score is less than 50%, then your grade will be FL.

3.2 Expectations of students

Students are expected to:

- attend all lectures, and ask questions
- attend all tutorials and actively participate in the discussions
- attend all lab classes and work diligently on the exercises
- do all of the assignment work themselves, asking only the forum or tutors for help

On the course forum, students should:

- use relevant/meaningful message titles on all posts
- ask questions clearly and provide sufficient background information that the question can be reasonably answered
- not post significant pieces of code, especially code for assignment

4. Course schedule and structure

This course consists of 8 hours of class contact hours per week. You are expected to take an additional 5 hours outside classes to complete assessments, readings, and exam preparation.

Week	Lectures	Tutorial and Labs	Assessment	Related CLO
Week 1	Course Introduction, bit manipulation, integer representation	Welcome, C Revision, man		1
Week 2	Instruction set architecture and MIPS assembly programming	Bit manipulation, Integers		3, 4
Week 3	MIPS assembly programming	MIPS assembly programming	Test 1	3, 4
Week 4	MIPS assembly programming	MIPS assembly programming	Test 2	4
Week 5	Data representation: IEEE-754; UTF-8	MIPS assembly programming	Assignment 1 Released Test 3	4
Week 6	File Systems	IEEE-754; UTF-8	Test 4	4
Week 7	Flexibility Week: No lectures	No Labs or Tutorials		
Week 8	File Metadata	Manipulating files	Assignment 1 Due Test 5	2,8
Week 9	Processes	Manipulating file meta data and directories	Assignment 2 Released Test 6	2,6
Week 10	Parallelism, concurrency, synchronisation, coordination, communication	Processes	Test 7	8

Week 11	Virtual Memory	Concurrency	Test 8	7
Week 12	Exam Preparation	Virtual Memory, Exam Preparation	Assignment 2 due	5,7

5. Assessment

5.1 Assessment tasks

Assessment task	Length	Weight	Due	CLOs
Weekly Tests	Throughout semester	10%	Weeks 3,4,5,6,8,9,10,11, 12	1-9
Assignment 1 (assembly language)	4 weeks	15%	Week 8	4
Assignment 2 (C programming)	4 weeks	15%	Week 12	2,5
Labs	Throughout Semester	15%	Weekly	1-7,9
Final Exam		45%	Exam period	1-9

Programming assignments are marked primarily on their correctness with respect to the assignment specification. Test cases will be provided for each assignment, and further (unseen) test cases will be used for marking. A small component of the mark will be for code quality.

Lab exercises and weekly tests are similarly marked primarily on their correctness, demonstrators will also give feedback on code quality for some labs.

Final Mark

Your final mark for this course will be computed using the above assessments as follows:

CourseWorkMark	=	TestMark + LabMark + Ass1Mark + Ass2Mark	out of 55
ExamMark			out of 45
ExamOK	=	ExamMark ≥ 18.0/45	true/false
FinalMark	=	CourseWorkMark + ExamMark	out of 100
FinalGrade	=	UF, if ! ExamOK && FinalMark ≥ 50 FL, if FinalMark < 50/100 PS, if 50/100 ≤ FinalMark < 65/100 CR, if 65/100 ≤ FinalMark < 75/100 DN, if 75/100 ≤ FinalMark < 85/100 HD, if FinalMark ≥ 85/100	

5.2 Assessment criteria and standards

In all programming work, the primary assessment criterion is correctness (i.e. does the code produce the expected output/behaviour according to the exercise specification). This will be tested by executing code against a variety of test cases, some of which are available to students, and others of which are used after submission for assessment purposes. Code is also expected to be expressed clearly, with consistent formatting and using relevant variable names.

5.3 Submission of assessment tasks

All assignments, weekly tests and labs will be submitted online via CSE's submission system. The UNSW standard late penalty for assessment is 5% per day for 5 days - this may be implemented hourly for assignments where each hour your assignment is submitted late reduces its mark by 0.2%. For example, if an assignment worth 60/100 was submitted 10 hours late, it would be awarded a mark of 58.8/100.

Beware - submissions more 5 days late will receive zero marks. This again is the UNSW standard assessment policy.

If you are unable to submit an assignment, weekly test or lab by the due date, due to medical reasons or other reasons which significantly affect your ability to carry out your work, you should submit a special consideration form and contact the lecturer as soon as possible, preferably well before the assignment deadline. If the lecturer considers that your ability to complete the assignment on time has been adversely affected, an extension may be granted to make up for the time you were unable to work on the assignment.

5.4. Feedback on assessment

Assignments will be marked after the submission deadline and annotated with comments by the tutor. You can discuss the tutor's comments in a lab class after you have received the feedback.

Weekly tests and most labs will be automarked. In some weeks lab demonstrators will discuss your lab submission with you during the lab class in the week following the submission.

6. Readings and resources

There is no single text book that covers all of the material in this course at the right level of detail and using the same technology base as we are. The lecture notes should provide sufficient detail to introduce topics, and you will then study them in further depth in the tutes, labs and assignments.

There are also many online resources available, and we will provide links to the most useful ones. Some are listed below. If you find others, please post links in the Comments section on the Course Outline page.

The following is a Recommended Reading for this course

Computer Systems: A Programmer's Perspective , by Randal E. Bryant and David R. O'Hallaron, Prentice-Hall (web site)

There are copies in the UNSW Bookstore and in the library. It covers many of the topics in the course, but uses a different machine language (i.e. not MIPS).

Some suggestions for other books that cover at least some of the topics in this course

- *Introduction to Computer Systems: From Bits and Gates to C and Beyond* , by Yale N. Patt and Sanjay J. Patel, McGraw Hill
- *nand2tetris: The Elements of Computing Systems: Building a Modern Computer from First Principles* , by Noam Nisan and Shimon Schocken, MIT Press (web site , including lecture slides)

Documentation for the various systems used in the course is linked from the course website.