UNSW Course Outline

# COMP1521 Computer Systems Fundamentals - 2024

Published on the 17 May 2024

## General Course Information

**Course Code :** COMP1521
**Year :** 2024
**Term :** Term 2
**Teaching Period :** T2
**Is a multi-term course? :** No
**Faculty :** Faculty of Engineering
**Academic Unit :** School of Computer Science and Engineering
**Delivery Mode :** In Person
**Delivery Format :** Standard
**Delivery Location :** Kensington
**Campus :** Sydney
**Study Level :** Postgraduate, Undergraduate
**Units of Credit :** 6

**Useful Links**

Handbook Class Timetable

## Course Details & Outcomes

### Course Description

This course introduces students to how computer systems are structured in terms of basic electronic components, how they are used to implement procedural programs, and how they are structured as a collection of software layers. It introduces students to low-level software layers

such as operating systems, and introduces concurrency concepts. The goal is to give students a solid understanding of what happens when high-level programs are executed, as a basis for further study in important areas of computing such as computer architecture, operating systems.

## Course Aims

- Provide a programmer's view on how a computer system executes programs, manipulates data and communicates
- Enable students to become effective programmers in dealing with issues of performance, portability, and robustness
- Serve as a foundation for later courses on networks, operating systems, computer architecture and compilers, where a deeper understanding of systems-level issues is required

## Relationship to Other Courses

Before commencing this course, students should be able to ...

- write simple programs in the C programming language
- define and invoke functions and return results in C
- define and manipulate structured data in C
- use pointers to access data objects

These are assumed to have been acquired in COMP1511 or COMP1911.

# Course Learning Outcomes

| Course Learning Outcomes |
| --- |
| CLO1 : Describe the architectural layers (fundamental parts) of a modern computer systems from hardware device (chip) levels upwards |
| CLO2 : Describe the principles of memory management and explain the workings of a system with virtual memory management |
| CLO3 : Explain how the major components of a CPU work together, including how data and instructions are represented in a computer |
| CLO4 : Design, implement, and analyse small programs at the assembly/machine level |
| CLO5 : Describe the relationship between a high-level procedural language (C) and assembly (machine language) which implements it, including how a compiled program is executed in a classical von Neumann machine |
| CLO6 : Describe the components comprising, and the services offered by, an operating system |
| CLO7 : Implement simple programs involving communication and concurrency |

| Course Learning Outcomes | Assessment Item |
| --- | --- |
| CLO1 : Describe the architectural layers (fundamental parts) of a modern computer systems from hardware device (chip) levels upwards | • Labs<br>• Tests<br>• Final Exam |
| CLO2 : Describe the principles of memory management and explain the workings of a system with virtual memory management | • Assignments 1 and 2<br>• Labs<br>• Tests<br>• Final Exam |
| CLO3 : Explain how the major components of a CPU work together, including how data and instructions are represented in a computer | • Labs<br>• Tests<br>• Final Exam |
| CLO4 : Design, implement, and analyse small programs at the assembly/machine level | • Assignments 1 and 2<br>• Labs<br>• Tests<br>• Final Exam |
| CLO5 : Describe the relationship between a high-level procedural language (C) and assembly (machine language) which implements it, including how a compiled program is executed in a classical von Neumann machine | • Assignments 1 and 2<br>• Labs<br>• Tests<br>• Final Exam |
| CLO6 : Describe the components comprising, and the services offered by, an operating system | • Tests<br>• Final Exam |
| CLO7 : Implement simple programs involving communication and concurrency | • Tests<br>• Final Exam |

## Learning and Teaching Technologies

Moodle - Learning Management System | EdStem | custom LMS | Blackboard Collaborate

## Learning and Teaching in this course

This course uses the standard set of practice-focussed teaching strategies employed by most CSE foundational courses:

- Lectures … introduce concepts, show examples
- Tutorials … reinforce concepts and provide additional examples
- Lab Work … provide examples of using various technologies
- Assignments … allow you to solve larger problems

Having said that, the second half of the course is more discursive than other CSE foundational courses.

This course is taught the way it is because it aims to give a broad view of many topics in computer systems, to provide a foundation for further study in later systems-related courses. At the same time, it provides further practice in developing software, but at a level closer to the machine than other foundational courses.

## Other Professional Outcomes

This course contributes to the development of the following graduate capabilities:

| | |
|---|---|
| scholarship: understanding of their discipline in its interdisciplinary context | lectures, assignments |
| scholarship: capable of independent and collaborative enquiry | lab work, assignments |
| scholarship: rigorous in their analysis, critique, and reflection | tutorials |
| scholarship: able to apply their knowledge and skills to solving problems | tutorials, lab work, assignments |
| scholarship: ethical practitioners | all course-work, by doing it yourself |
| scholarship: capable of effective communication | tutorials |
| scholarship: digitally literate | everywhere in CSE |
| leadership: enterprising, innovative and creative | assignments |
| leadership: collaborative team workers | lab work, assignments |
| professionalism: capable of operating within an agreed Code of Practice | all prac work |

# Assessments

## Assessment Structure

| Assessment Item | Weight | Relevant Dates |
|---|---|---|
| Labs<br>Assessment Format: Individual | 15% | Start Date: Week 1 - Week 10, Except Week 6<br>Due Date: Week 3 - Week 10, Except Week 6 |
| Tests<br>Assessment Format: Individual | 10% | Start Date: Week 3 - Week 10<br>Due Date: Week 3 - Week 10 |
| Assignments 1 and 2<br>Assessment Format: Individual | 30% | Start Date: Not Applicable<br>Due Date: Not Applicable |
| Final Exam<br>Assessment Format: Individual | 45% | Start Date: During Exam Period<br>Due Date: During Exam Period |

## Assessment Details

### Labs

Assessment Overview

Following the tutorial class each week, there will be a two-hour laboratory class, during which you will work on a variety of small practical problems involving the tools introduced in lectures. Because this course has a significant practical component, laboratory classes are **important**. If you do not put a good amount of effort into the lab classes, you risk failing the final exam.

Each week, there will be one or more exercises to work on. These exercises will be released in the week preceding the lab class.

During the lab, your tutor will 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.

Completed exercises need to be submitted. You must submit exercises before the deadline using **give** to obtain a mark for a lab exercise, The usual lab exercise submission deadline will be 12:00 (midday) Monday some lab exercises may have an extended deadline.

The lab exercises for each week are worth in total 2 marks. All of your lab marks will be summed 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 COMP1521, attempting some challenge exercises is highly recommended.

If your goal is just to master the core material and pass COMP1521, you can ignore challenge exercises.

**Course Learning Outcomes**

- CLO1 : Describe the architectural layers (fundamental parts) of a modern computer systems from hardware device (chip) levels upwards
- CLO2 : Describe the principles of memory management and explain the workings of a system with virtual memory management
- CLO3 : Explain how the major components of a CPU work together, including how data and instructions are represented in a computer
- CLO4 : Design, implement, and analyse small programs at the assembly/machine level
- CLO5 : Describe the relationship between a high-level procedural language (C) and assembly (machine language) which implements it, including how a compiled program is executed in a classical von Neumann machine

**Assignment submission Turnitin type**

This is not a Turnitin assignment

## Tests

**Assessment Overview**

There will be weekly tests from weeks 3–10 designed to give you timely and realistic feedback of your understanding of the course material. Tests may be programming exercises, multiple choice questions, or both.

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:

1. no assistance permitted from any person;
2. a time limit;
3. no access to materials (written or online) except specified language documentation or man pages.

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 8 test marks.

Course Learning Outcomes

- CLO1 : Describe the architectural layers (fundamental parts) of a modern computer systems from hardware device (chip) levels upwards
- CLO2 : Describe the principles of memory management and explain the workings of a system with virtual memory management
- CLO3 : Explain how the major components of a CPU work together, including how data and instructions are represented in a computer
- CLO4 : Design, implement, and analyse small programs at the assembly/machine level
- CLO5 : Describe the relationship between a high-level procedural language (C) and assembly (machine language) which implements it, including how a compiled program is executed in a classical von Neumann machine
- CLO6 : Describe the components comprising, and the services offered by, an operating system
- CLO7 : Implement simple programs involving communication and concurrency

Assignment submission Turnitin type

This is not a Turnitin assignment

## Assignments 1 and 2

Assessment Overview

There are two assessable programming assignments. Assignments give you the chance to practice what you have learnt on relatively large problems (compared to the small exercises in the labs). Assignments are a *very important* part of this course, therefore it is essential that you attempt them yourself.

- Assignment 1, on Assembly programming (MIPS) worth 15%
- Assignment 2, on System programming worth 15%

Assignments are primarily automarked, with feedback from tutors on programming style.

Course Learning Outcomes

- CLO2 : Describe the principles of memory management and explain the workings of a system with virtual memory management
- CLO4 : Design, implement, and analyse small programs at the assembly/machine level
- CLO5 : Describe the relationship between a high-level procedural language (C) and assembly (machine language) which implements it, including how a compiled program is executed in a classical von Neumann machine

Detailed Assessment Description

- Assignment 1, on Assembly programming (MIPS), worth 15% - Due End of Week 5
- Assignment 2, on System programming, worth 15% - Due Week 10

**Assignment submission Turnitin type**

This is not a Turnitin assignment

## Final Exam

**Assessment Overview**

There will be a three-hour final exam, held in the CSE labs during the exam period. The exam runs in a closed environment (no internet) but with relevant documentation provided.

During this exam you will need to execute, debug and test your answers to implementation tasks which 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% (18.0/45) on the exam (after scaling), 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.

**Course Learning Outcomes**

- CLO1 : Describe the architectural layers (fundamental parts) of a modern computer systems from hardware device (chip) levels upwards
- CLO2 : Describe the principles of memory management and explain the workings of a system with virtual memory management
- CLO3 : Explain how the major components of a CPU work together, including how data and instructions are represented in a computer
- CLO4 : Design, implement, and analyse small programs at the assembly/machine level
- CLO5 : Describe the relationship between a high-level procedural language (C) and assembly (machine language) which implements it, including how a compiled program is executed in a classical von Neumann machine
- CLO6 : Describe the components comprising, and the services offered by, an operating system
- CLO7 : Implement simple programs involving communication and concurrency

**Assessment Length**

3 hours

**Assignment submission Turnitin type**

This is not a Turnitin assignment

**Hurdle rules**

There is a hurdle requirement on the final exam. If you do not score at least 40% (18.0/45) on the exam (after scaling), 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.

# General Assessment Information

Laboratory Classes

Following the tutorial class each week, there will be a two-hour laboratory class, during which you will work on a variety of small practical problems involving the tools introduced in lectures. Because this course has a significant practical component, laboratory classes are **important**. If you do not put a good amount of effort into the lab classes, you risk failing the final exam.

Each week, there will be one or more exercises to work on. These exercises will be released in the week preceding the lab class.

During the lab, your tutor will 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.

Completed exercises need to be submitted. You must submit exercises before the deadline using **give** to obtain a mark for a lab exercise, The usual lab exercise submission deadline will be 12:00 (midday) Monday some lab exercises may have an extended deadline

The lab exercises for each week are worth in total 2 marks. All of your lab marks will be summed 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 COMP1521, attempting some challenge exercises is highly recommended.

If your goal is just to master the core material and pass COMP1521, you can ignore challenge exercises.

Assignments

There are two assessable programming assignments. Assignments give you the chance to practice what you have learnt on relatively large problems (compared to the small exercises in

the labs). Assignments are a *very important* part of this course, therefore it is essential that you attempt them yourself.

- Assignment 1, on Assembly programming (MIPS); due Friday Week 5; worth 15%
- Assignment 2, on System programming; due Friday Week 9; worth 15%

Late assignments submissions will be penalised. The exact penalty will be the standard UNSW late penalty of a mark reduction equal to 5% of the maximum assessment mark per day late, or a zero mark after 5 days (i.e. 120 hours).

Weekly Tests

There will be weekly tests from weeks 3–10 designed to give you timely and realistic feedback of your understanding of the course material. Tests may be programming exercises, multiple choice questions, or both.

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:

1. no assistance permitted from any person;
2. a time limit;
3. no access to materials (written or online) except specified language documentation or man pages.

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 8 test marks.

Final Exam

There will be a three-hour final exam, held in the CSE labs during the exam period. This will be centrally timetabled, and appear in your UNSW exam timetable.

During this exam you will need to execute, debug and test your answers to implementation tasks which 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% (18.0/45) on the exam (after scaling), 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.

Grading Basis

Standard

Requirements to pass course

There is a hurdle requirement on the final exam. If you do not score at least 40% (18.0/45) on the exam (after scaling), 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.

# Course Schedule

| Teaching Week/Module | Activity Type | Content |
|---|---|---|
| Week 1 : 27 May - 2 June | Lecture | course intro; MIPS assembly programming |
| | Tut-Lab | C revision |
| Week 2 : 3 June - 9 June | Lecture | MIPS assembly programming |
| | Tut-Lab | MIPS assembly programming |
| Week 3 : 10 June - 16 June | Lecture | MIPS assembly programming |
| | Tut-Lab | MIPS assembly programming |
| Week 4 : 17 June - 23 June | Lecture | MIPS assembly programming |
| | Tut-Lab | MIPS assembly programming |
| Week 5 : 24 June - 30 June | Lecture | Bit manipulations; Integer representations |
| | Tut-Lab | MIPS assembly programming |
| Week 6 : 1 July - 7 July | Other | flexibility week |
| Week 7 : 8 July - 14 July | Lecture | IEEE-754; UTF-8 |
| | Tut-Lab | Bit manipulations; Integer representations |
| Week 8 : 15 July - 21 July | Lecture | file metadata; file systems |
| | Tut-Lab | IEEE-754; UTF-8 |
| Week 9 : 22 July - 28 July | Lecture | concurrency, parallelism, threads |
| | Tut-Lab | manipulating files, metadata, and directories |
| Week 10 : 29 July - 4 August | Lecture | hardware caching, final exam |
| | Tut-Lab | using threads in C |

## Attendance Requirements

Students are strongly encouraged to attend all classes and review lecture recordings.

# Course Resources

## Recommended Resources

There is no single 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 architecture (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
- *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.

## Course Evaluation and Development

Every term, student feedback is requested in a survey using UNSW's myExperience online survey system where the feedback will be used to make improvements to the course.

# Staff Details

| Position | Name | Email | Location | Phone | Availability | Equitable Learning Services Contact | Primary Contact |
|---|---|---|---|---|---|---|---|
| Convenor | Hammond Pearce | | | | | Yes | No |
| Administrator | COURSE EMAIL | | | | | Yes | Yes |

# Other Useful Information

## Academic Information

### I. Special consideration and supplementary assessment

If you have experienced an illness or misadventure beyond your control that will interfere with your assessment performance, you are eligible to apply for Special Consideration prior to, or

within 3 working days of, submitting an assessment or sitting an exam.

Please note that UNSW has a Fit to Sit rule, which means that if you sit an exam, you are declaring yourself fit enough to do so and cannot later apply for Special Consideration.

For details of applying for Special Consideration and conditions for the award of supplementary assessment, please see the information on UNSW's Special Consideration page.

**II. Administrative matters and links**

All students are expected to read and be familiar with UNSW guidelines and polices. In particular, students should be familiar with the following:

- Attendance
- UNSW Email Address
- Special Consideration
- Exams
- Approved Calculators
- Academic Honesty and Plagiarism
- Equitable Learning Services

**III. Equity and diversity**

Those students who have a disability that requires some adjustment in their teaching or learning environment are encouraged to discuss their study needs with the course convener prior to, or at the commencement of, their course, or with the Equity Officer (Disability) in the Equitable Learning Services. Issues to be discussed may include access to materials, signers or note-takers, the provision of services and additional exam and assessment arrangements. Early notification is essential to enable any necessary adjustments to be made.

**IV. Professional Outcomes and Program Design**

Students are able to review the relevant professional outcomes and program designs for their streams by going to the following link: https://www.unsw.edu.au/engineering/student-life/student-resources/program-design.

*Note: This course outline sets out the description of classes at the date the Course Outline is published. The nature of classes may change during the Term after the Course Outline is published. Moodle or your primary learning management system (LMS) should be consulted for the up-to-date class descriptions. If there is any inconsistency in the description of activities between the*

*University timetable and the Course Outline/Moodle/LMS, the description in the Course Outline/ Moodle/LMS applies.*

## Academic Honesty and Plagarism

UNSW has an ongoing commitment to fostering a culture of learning informed by academic integrity. All UNSW students have a responsibility to adhere to this principle of academic integrity. Plagiarism undermines academic integrity and is not tolerated at UNSW. *Plagiarism at UNSW is defined as using the words or ideas of others and passing them off as your own.*

Plagiarism is a type of intellectual theft. It can take many forms, from deliberate cheating to accidentally copying from a source without acknowledgement. UNSW has produced a website with a wealth of resources to support students to understand and avoid plagiarism, visit: student.unsw.edu.au/plagiarism. The Learning Centre assists students with understanding academic integrity and how not to plagiarise. They also hold workshops and can help students one-on-one.

You are also reminded that careful time management is an important part of study and one of the identified causes of plagiarism is poor time management. Students should allow sufficient time for research, drafting and the proper referencing of sources in preparing all assessment tasks.

Repeated plagiarism (even in first year), plagiarism after first year, or serious instances, may also be investigated under the Student Misconduct Procedures. The penalties under the procedures can include a reduction in marks, failing a course or for the most serious matters (like plagiarism in an honours thesis or contract cheating) even suspension from the university. The Student Misconduct Procedures are available here:

www.gs.unsw.edu.au/policy/documents/studentmisconductprocedures.pdf

## Submission of Assessment Tasks

Work submitted late without an approved extension by the course coordinator or delegated authority is subject to a late penalty of five percent (5%) of the maximum mark possible for that assessment item, per calendar day.

The late penalty is applied per calendar day (including weekends and public holidays) that the assessment is overdue. There is no pro-rata of the late penalty for submissions made part way

through a day. This is for all assessments where a penalty applies.

Work submitted after five days (120 hours) will not be accepted and a mark of zero will be awarded for that assessment item.

For some assessment items, a late penalty may not be appropriate. These will be clearly indicated in the course outline, and such assessments will receive a mark of zero if not completed by the specified date. Examples include:

- Weekly online tests or laboratory work worth a small proportion of the subject mark;
- Exams, peer feedback and team evaluation surveys;
- Online quizzes where answers are released to students on completion;
- Professional assessment tasks, where the intention is to create an authentic assessment that has an absolute submission date; and,
- Pass/Fail assessment tasks.

## Faculty-specific Information

Engineering Student Support Services – The Nucleus - enrolment, progression checks, clash requests, course issues or program-related queries

Engineering Industrial Training – Industrial training questions

UNSW Study Abroad – study abroad student enquiries (for inbound students)

UNSW Exchange – student exchange enquiries (for inbound students)

UNSW Future Students – potential student enquiries e.g. admissions, fees, programs, credit transfer

**Phone**

(+61 2) 9385 8500 – Nucleus Student Hub

(+61 2) 9385 7661 – Engineering Industrial Training

(+61 2) 9385 3179 – UNSW Study Abroad and UNSW Exchange (for inbound students)

## School Contact Information

**CSE Help! - on the Ground Floor of K17**

- For assistance with coursework assessments.

**The Nucleus Student Hub** - https://nucleus.unsw.edu.au/en/contact-us

- Course enrolment queries.

**Grievance Officer** - grievance-officer@cse.unsw.edu.au

- If the course convenor gives an inadequate response to a query or when the courses convenor does not respond to a query about assessment.

**Student Reps** - stureps@cse.unsw.edu.au

- If some aspect of a course needs urgent improvement. (e.g. Nobody responding to forum queries, cannot understand the lecturer)

You should **never** contact any of the following people directly:

- Vice Chancellor

- Pro-vice Chancellor Education (PVCE)

- Head of School

- CSE administrative staff

- CSE teaching support staff

They will simply bounce the email to one of the above, thereby creating an unnecessary level of indirection and a delay in the response.