ECE 4750 Computer Architecture, Fall 2023

Course Syllabus

School of Electrical and Computer Engineering
Cornell University

revision: 2023-08-21-12-46

1. Course Information

Cross Listed	CS 4420 Computer Architecture
Co-Meet	ECE 5740 Computer Architecture
Prereqs	ECE 3140 (cross-listed as CS 3420) or CS 3410
Instructor	Prof. José F. Martínez, 220 Carpenter Hall, `martinez@cornell.edu` Office Hours: 220 Carpenter Hall, Fri 9am–Noon
Graduate TAs MoTuWeTh 7:30-9:30pm 314 Phillips	Chris Bakhos `cmb524` Cecilio Tamarit `ct652` Amy Wang `yw575` Socrates Wong `ssw96` Kailin Yang `ky362`
Undergraduate TAs MoTuWeTh 7:30-9:30pm 314 Phillips	Will Barkoff `wb273` Peter Cao `ppc49` Angela Cui `ayc62` Steven Lambert `sal267` Aidan McNay `acm289` Eshita Sangani `ens57` Michael Wei `mw756` Evan Williams `emw236`
Lectures	B11 Kimball Hall, Monday and Wednesday, 2:55–4:10pm
Disc. Section	G01 Gates Hall, Friday, 2:30–3:20pm
Required Materials	J. L. Hennessy and D. A. Patterson "Computer Architecture: A Quantitative Approach" 5th edition, Morgan Kaufmann, 2012 Available on Canvas, Amazon ($80) Cornell library has e-book and hard copy D. M. Harris and S. L. Harris "Digital Design and Computer Architecture" 2nd edition, Morgan Kaufmann, 2012 Amazon ($80) Cornell library has e-book and hard copy "ECE 4750 Course Packet" Available on Canvas
Website	`http://www.csl.cornell.edu/courses/ece4750`
Staff Email	`ece4750-staff-l@cornell.edu`

2. Description

This course aims to provide a strong foundation for students to understand the modern eras of computer architecture (i.e., the single-core era, multi-core era, and accelerator era) and to apply these insights and principles to future computer designs. The course is structured around the three primary building blocks of general-purpose computing systems: processors, memories, and networks.

The first half of the course focuses on the fundamentals of each building block. Topics include instruction set architecture; single-cycle, FSM, and pipelined processor microarchitecture; direct-mapped vs. set-associative cache memories; memory protection, translation, and virtualization; FSM and pipelined cache microarchitecture; cache optimizations; and integrating processors, memories, and networks. The second half of the course delves into more advanced techniques and will enable students to understand how these three building blocks can be integrated to build a modern shared-memory multicore system. Topics include superscalar execution, out-of-order execution, register renaming, memory disambiguation, branch prediction, and speculative execution; multithreaded, VLIW, and SIMD processors; and memory synchronization, consistency, and coherence. Students will learn how to evaluate design decisions in the context of past, current, and future application requirements and technology constraints.

This course includes a significant project decomposed into four lab assignments. Throughout the semester, students will gradually design, implement, test, and evaluate a complete multicore system capable of running simple parallel applications at the register-transfer level.

3. Objectives

This course is meant to be a capstone course in computer engineering that draws together concepts from across the ECE curriculum including digital logic design, computer organization, system-level software, and engineering design. The course will prepare students for jobs in the computer engineering industry and can act as a springboard to more advanced material in graduate-level courses. This course can also provide a foundation for students interested in performance programming, compilers, and operating systems; and it can provide system-level context for students interested in emerging technologies and digital circuits. By the end of this course, students should be able to:

- **describe** computer architecture concepts and mechanisms related to the design of modern processors, memories, and networks and explain how these concepts and mechanisms interact.
- **apply** this understanding to new computer architecture design problems within the context of balancing application requirements against technology constraints; more specifically, quantitatively assess a design's execution time in cycles and qualitatively assess a design's cycle time, area, and energy.
- **evaluate** various design alternatives and make a compelling quantitative and/or qualitative argument for why one design is superior to the other approaches.
- **demonstrate** the ability to implement and verify designs of varying complexity at the register-transfer level.
- **create** new designs at the register-transfer-level and the associated effective testing strategies.
- **write** concise yet comprehensive technical reports that describe designs, explain the testing strategy used to verify functionality, and evaluate the designs to determine the superior approach.

4. Prerequisites

This course is targeted towards senior-level undergraduate students and M.Eng. students, although it is also appropriate for advanced juniors and first-year Ph.D. students. An introductory course on computing is required (CS 1110 or equivalent). A course in digital logic design and computer organization (ECE 2300 or equivalent) and a course in system-level programming (ECE 3140 or equivalent) are also required. CS 3410 is a suitable replacement for ECE 2300 and ECE 3140 for the purposes of satisfying the prerequisites. Students should feel comfortable working with a hardware description language such as Verilog, SystemVerilog, or VHDL and have a reasonable understanding of digital logic, assembly-level programming, storage systems, basic pipelining, and simple cache design.

M.Eng. and Ph.D. students coming from undergraduate institutions other than Cornell may want to spend additional time reviewing the secondary required textbook, *"Digital Design and Computer Architecture, 2nd edition"* by D. M. Harris and S. L. Harris (Morgan Kaufmann, 2012), to refresh their understanding of basic concepts. Students who have less experience working with Verilog, then they are strongly encouraged to read Chapter 4 in Harris and Harris on digital design with Verilog and/or to review the optional text *"Verilog HDL: A Guide to Digital Design and Synthesis, 2nd edition"* by S. Palnitkar (Prentice Hall, 2003).

5. Topics

The course includes five parts: the first three parts cover the fundamentals of processor, memory, and network design, while the final two parts cover more advanced processor and memory design. In addition, the final lecture at the end of the course will present in detail an example architecture from industry to help illustrate the concepts discussed in class. A tentative list of topics for each part is included below. The exact topics covered in the course are subject to change based on student progress and interest.

- **Part 1: Processors (6 lectures) –** instruction set architecture; single-cycle, FSM, and pipelined processor microarchitecture; resolving structural, data, control, and name hazards; and analyzing processor performance
- **Part 2: Memories (5 lectures) –** memory technology; direct-mapped vs. associative caches; write-through vs write-back caches; memory protection, translation, and virtualization; FSM and pipelined cache microarchitecture; and analyzing memory performance
- **Part 3: Integrating Processors, Memories, and Networks (2 lectures) –** processor and L1 cache interface; banked memory systems; message-passing systems; shared-memory systems
- **Part 4: Advanced Processors (12 lectures) –** superscalar execution, out-of-order execution, register renaming, memory disambiguation, branch prediction, speculative execution; multithreaded, VLIW, and SIMD processors
- **Part 5: Advanced Memories (2 lectures) –** memory synchronization, consistency, and coherence

6. Required Materials

There are three materials that students are required to have access to for the course: the primary course textbook, the secondary course textbook, and the course packet.

- **Hennessy and Patterson Textbook –** The primary required textbook for the course is *"Computer Architecture: A Quantitative Approach, 5th ed.,"* by J. L. Hennessy and D. A. Patterson (Morgan Kaufmann, 2012). This is the classic text in the field. The first chapter will be available on Canvas for download. This textbook is in Uris library, available as an e-book to any Cornell student with a valid NetID, and is also available through Canvas to all enrolled students.
- **Harris and Harris Textbook –** The secondary required textbook for the course is *"Digital Design and Computer Architecture, 2nd ed.,"* by D. M. Harris and S. L. Harris (Morgan Kaufmann, 2012). This is the primary required textbook for ECE 2300. There will be some assigned reading from this book, but a student may or may not need to purchase the actual book depending on how comfortable a student is with the more basic material. M.Eng. and Ph.D. students coming from undergraduate institutions other than Cornell may want to purchase this book to solidify their understanding of digital logic design and basic computer architecture. The book also includes some useful background information on digital design using the Verilog hardware description language. This textbook is in Uris library and is also available as an e-book to any Cornell student with a valid NetID.
- **Course Packet –** The course will use a packet of additional reading material on processors, memories, and networks that is meant to complement the course textbook. The information in the packet is either not in the textbook, or is presented in a useful alternative way. The course packet materials are posted on Canvas.

7. Optional Materials

There are a few additional books that students may find useful for providing background on Verilog and SystemVerilog.

- **Verilog Book –** *"Verilog HDL: A Guide to Digital Design and Synthesis, 2nd ed.,"* by S. Palnitkar (Prentice Hall, 2003) provides a good introduction to Verilog-2001 well suited for the beginner.
- **SystemVerilog Book –** *"SystemVerilog for Design: A Guide to Using SystemVerilog for Hardware Design and Modeling, 2nd ed.,"* by S. Sutherland, S. Davidmann, and P. Flake (Springer, 2006) provides a good introduction to the new features in SystemVerilog that can enable productive hardware design.

8. Format and Procedures

This course includes a combination of lectures, quizzes, discussion sections, readings, practice problems, lab assignments, and exams. ECE 5740 students will also be required to complete additional position papers and a position defense.

- **Lectures –** Lectures will be from 2:55pm to 4:10pm every Monday and Wednesday in B11 Kimball Hall excluding the following academic holidays: Labor Day, Fall Break, and Thanksgiving. We will start promptly at 2:55pm so please arrive on time. Students are expected to attend all lectures, be attentive during lectures, and participate in class discussions. Please turn off all cellular phones during class.
- **Quizzes –** There will be a short quiz at the beginning of some lectures. The quiz should take about five minutes and will cover some of the key topics discussed in the previous lecture. Quizzes may or may not be announced beforehand, and there are no make-up quizzes. The lowest quiz score is dropped. Students should prepare for a potential quiz by simply reviewing the material from the previous lecture before coming to class. Solutions to quizzes will be available online soon after the quiz is given for formative self-assessment.
- **Discussion Section –** The discussion section will be most Fridays from 2:30pm to 3:20pm in G01 Gates Hall. These discussion sections will be relatively informal, with the primary focus being on facilitating student's ability to complete the lab assignments and on reviewing material from lectures using problem-based learning. Many of the discussion sections will involve hands-on activities so students should bring their laptops.
- **Readings –** Students are expected to complete all of the assigned reading according to the schedule on the course website, although there is some flexibility. Some students may prefer to complete the readings before the corresponding lecture, while others may prefer to do so after the corresponding lecture. Either strategy is acceptable. The readings are contained within the course textbooks and the course packet.
- **Practice Problems –** The course will include practice problems distributed throughout the semester to help students put the concepts learned in lectures and reading into practice. Solutions will not be provided for the practice problems. Students should work either individually or collaboratively on the problem sets and then discuss their solutions with course instructors during office/lab hours.
- **Lab Assignments –** The course will include four lab assignments that allow students to incrementally design, implement, test, and evaluate a complete multicore system with an integrated collection of processors, memories, and networks. Students are expected to work in a group of two students, although groups of one or three students may be allowed with explicit instructor permission in exceptional circumstances (see Section 11.F for collaboration policy). It is suggested that students form a group early on and keep the same group throughout the semester. Students can either form their own groups or ask the instructor to form a group for them. Much more detail about the lab assignments is provided in the Lab Assignment Assessment Rubric to be posted on Canvas. Students will be using the ECE Computing Resources to complete the lab assignments, the lab code must be submitted via GitHub and the lab report must be submitted in PDF format via Canvas. No other means of submission or electronic format will be accepted. Each lab also has a lab milestone. Although the milestones are not graded, students are expected to complete the milestone and submit their code via GitHub by the milestone deadline. Code that is submitted by the deadline will be reviewed by the course staff and feedback will be provided to the students. Students who do not meet the milestone usually struggle to com-

plete the entire lab by the final lab deadline. Lab assignments are due on Thursdays at 10pm (see Section 11.D for late submission policy).

- **Exams –** The course includes two prelim exams and a cumulative final exam. The exams assess student understanding of the material presented in lectures and assigned readings. The prelim exams include a series of short answer questions. The final exam includes two parts: the first part includes a series of short answer questions, while the second part includes a detailed design problem that requires students to compare and contrast two design alternatives. The exams do *not* assess student understanding of the Linux and RTL development environment (e.g., the Linux command line, Verilog, etc.) used for the lab assignments. If students have a scheduling conflict with the exam, they must let the instructor know as soon as possible, but no later than one week before the prelim or final exam. Usually, the only acceptable scheduling conflict is due to another exam at the same time. Exceptions can only be made for a family or medical emergency. Graded exams and exam solutions are only available for review under the supervision of a course instructor. You may not remove your graded exam, nor may you remove the exam solutions.
- **Position Papers/Defense –** ECE 5740 students are required to complete three short position papers throughout the semester qualitatively comparing and contrasting alternative architectures before arguing strongly for one of the alternatives. Each position paper focuses on either processor, memory, or network architecture. Relevant readings will be assigned for students to prepare their position papers. Position papers should be submitted in PDF format via Canvas. ECE 5740 students will also be required to complete a 15-min position defense with the instructor at the end of the semester. To prepare for the position defense, the student should choose one of their position papers and be ready to make a strong and compelling argument for their chosen architecture. Students should be ready to defend their choice through questions from the instructor.

9. Assignment and Exam Schedule

The current schedule is on the course website. Most assignments are due on Thursdays at 10pm. Changes to this schedule will be posted as announcements via Ed.

Thu	Sep	7	Lab 1: Increment Milestone
Thu	Sep	14	Lab 1: Iterative Integer Multiplier
Thu	Sep	21	→ *ECE 5740: Processor Position Paper*
Thu	Sep	28	Lab 2: Incremental Milestone
Thu	Oct	5	Lab 2: Pipelined Processor
Thu	Oct	12	Prelim from 7:30–9:00pm in B11 Kimball Hall
Thu	Oct	19	→ *ECE 5740 Memory Position Paper*
Thu	Oct	26	Lab 3: Incremental Milestone
Thu	Nov	2	Lab 3: Blocking Cache
Thu	Nov	9	→ *ECE 5740 Network Position Paper*
Thu	Nov	16	Lab 4: Incremental Milestone
Tue	Nov	21	Prelim from 7:30–9:00pm in B11 Kimball Hall
Thu	Nov	30	Lab 4: Multicore System
	Dec		Final Exam (location TBD)

10. Grading Scheme

This course will adopt a philosophy of "grading for equity" where grading is exclusively used to assess mastery of the material covered in the course as opposed to rewarding effort and/or incentivizing specific behaviors. To this end, each part or criterion of every assignment is graded on a four-point scale without any curve. A score of 4.25 is an A+, 4 roughly corresponds to an A, 3 roughly corresponds to a B, 2 roughly corresponds to a C, and so on. A score of 4.0 usually indicates that the submitted work demonstrates no misunderstanding (there may be small mistakes, but these mistakes do not indicate a misunderstanding) or there may be a very small misunderstanding that is vastly outweighed by the demonstrated understanding. A score of 3.0 usually indicates that the submitted work demonstrates more understanding than misunderstanding. A score of 2.0 usually indicates that the submitted work demonstrates more misunderstanding than understanding. A score of 1.0 usually indicates that the submitted work is significantly lacking in some way. A score of 4.25 is reserved for when the submitted work is perfect with absolutely no mistakes or is exceptional in some other way.

Total scores are a weighted average of the scores for each part or criteria. Parts or criteria are usually structured to assess a student's understanding according to four kinds of knowledge: basic recall of previously seen concepts, applying concepts in new situations, qualitatively and quantitatively evaluating design alternatives, and creatively implementing new designs; these are ordered in increasing sophistication and thus increasing weight. Detailed rubrics for all quizzes and exams are provided once the assignment has been graded to enable students to easily see how the score was awarded. For lab assignments, a detailed Lab Assignment Assessment Rubric is available on Canvas.

The final grade is calculated using a weighted average of all assignments. Each lab assignment is weighted equally. All quiz grades are averaged to form a single total. Students can drop their lowest quiz score. At the instructor's discretion, additional quiz scores may be dropped depending on the total number of quizzes in the semester. The weighting for the various assignments is shown below.

	ECE 4750	ECE 5740	
Quizzes	10%	7.5%	(students can drop lowest score)
Lab Code	25%	22.5%	(weighted equally)
Lab Reports	10%	7.5%	(weighted equally)
Prelim Exams	30%	30%	(weighted equally)
Final Exam	25%	25%	
Position Papers/Defense	n/a	7.5%	(weighted equally)

Note that the prelim and final exams account for over half of a student's final grade. The exams in this course are very challenging. Successful students begin preparing for the exams far in advance by carefully reviewing the assigned readings, independently developing study problems, and participating in critical study groups.

To pass the course, a student must at a bare minimum satisfy the following requirements: (1) submit three out of the four lab assignments; (2) take the prelim exams; and (3) take the final exam. **If a student does not satisfy these criteria then that student may fail the course regardless of the student's numerical grade.**

11. Policies

This section outlines various policies concerning auditors, usage of cellular phones and laptops in lecture, audio/video recording in lecture, turning in assignments late, regrading assignments, collaboration, and accommodations for students with disabilities.

11.A Auditor Policy

Casual listeners that attend lecture but do not enroll as auditors are not allowed; you must enroll officially as an auditor. *If you would like to audit the course please talk to the instructor first!* Usually we wait until the second week of classes before allowing auditors to enroll, to ensure there is sufficient capacity in the lecture room. If you do not plan on attending the lectures, then please do not audit the course. Please note that students are not allowed to audit the course and then take it for credit in a later year unless there is some kind of truly exceptional circumstance.

11.B Cellular Phones and Laptops in Lecture Policy

Students are prohibited from using cellular phones. Although laptops are not prohibited, they are strongly discouraged, as it is not practical to take notes with a laptop for this course: students will need to write on the handouts, quickly draw pipeline diagrams, and sketch microarchitectural block diagrams during lecture. Tablets are encouraged.

11.C Photo and Audio/Video Recording in Lecture Policy

Any kind of photography and audio or video recording during lectures is strictly prohibited without express written consent from the instructor. This includes taking photos with cell phones, taking videos with cell phones, and/or using voice recorders. These kinds of activities are distracting to the other students and the instructor, and they stifle participation.

11.D Late Submission Policy

- One minute late = Not submitted = Zero
- All assignments due Thursday at 10pm
- An automatic 24-hour extension for *one* assignment is available

Lab reports and position papers must be submitted electronically in PDF format on Canvas, and lab code must be submitted electronically via GitHub (as explained in the lab handout). **No other formats will be accepted!** Everything must be submitted by 10pm on the due date, although there is an automatically granted 24-hour extension for *one* assignment. No additional extensions beyond this one-time automatic extension will be granted except for a family or medical emergency. If you submit more than one assignment past the deadline but within the following 24 hours, you will be automatically credited the one with the highest weighted grade. We will be using the online Canvas assignment submission system. You can continue to resubmit your files as many times as you would like up until the deadline, so please feel free to upload early and often. **If you submit an assignment even one minute past the deadline, then the assignment will not be accepted.**

11.E Regrade Policy

Addition errors in the total score are always applicable for regrades. Regrades concerning the actual solution should be rare and are only permitted when there is a significant error. Please only make

regrade requests when the case is strong and a significant number of points are at stake. Regrade requests should be submitted online via a private post on Ed within one week of when an assignment is returned to the student or within one week of when an exam is reviewed with the class. You must provide a justification for the regrade request.

11.F Collaboration Policy

The work you submit in this course is expected to be the result of your individual effort only, or in the case of lab assignments, the result of you and your partner's effort only. Your work should accurately demonstrate your understanding of the material. The use of a computer in no way modifies the standards of academic integrity expected under the University Code.

You are encouraged to study together and to discuss information and concepts covered in lecture with other students. You can give "consulting" help to or receive "consulting" help from other students. Students can also freely discuss basic computing skills or the course infrastructure. **However, this permissible cooperation should never involve one student (or lab group) having possession of or observing in detail a copy of all or part of work done by someone else, in the form of an email, an email attachment file, a flash drive, a hard copy, or on a computer screen.** Students are not allowed to seek consulting help from online forums outside of Cornell University. Students are not allowed to use online solutions (e.g., from ChatGPT, Course Hero, or Chegg) from previous offerings of this course. Students are encouraged to seek consulting help from their peers and from the course staff via office hours and the online Ed discussion forums. **If a student receives consulting help from anyone outside of the course staff, then the student must acknowledge this help on the submitted assignment.**

During in-class paper quizzes and examinations, you must do your own work. Talking or discussion is not permitted during the in-class paper quizzes and examinations, nor may you compare papers, copy from others, or collaborate in any way. Students must not discuss a quiz/exam's contents with other students who have not taken the quiz/exam. If prior to taking it, you are inadvertently exposed to material in a quiz/exam (by whatever means) you must immediately inform an instructor.

Should a violation of the code of academic integrity occur, then a primary hearing will be held. See `https://theuniversityfaculty.cornell.edu/academic-integrity` for more information about academic integrity proceedings.

Examples of acceptable collaboration:

- Bob and Beth are struggling to complete a lab assignment which requires implementing a direct-mapped cache. They talk with Alice and Adam and learn that both groups are really struggling. So the four students get together for a brainstorming session. They review the lecture and reading materials and then sketch on a whiteboard some ideas on how to implement a direct-mapped cache. They might also sketch out some code snippets to try and understand the best way to describe some of the hardware. Then each group independently writes the code for the assignment and includes an acknowledgment of the help they received from the other group in their submission. At no time do the groups actually share code.
- Bob and Beth are having difficulty figuring out difficult test cases for their pipelined processor. They make a post on Ed to see if anyone has some general ideas for tricky corner cases. Alice and Adam figured out an interesting test case that ensures their pipelined processor correctly forwards the address to a store instruction, so Alice and Adam post a qualitative description of this test case. Bob and Beth then independently write the code for this test case and then include an acknowledgment of the help they received from the other group in their submission. At no time do the groups actually share test code.

- Bob and Alice are seated close to each other in the final exam. Bob is struggling on a problem requiring drawing a detailed pipeline diagram. While stretching, he accidentally sees the pipeline diagram on Alice's solution. This was an unintentional mistake, but now he has seen the pipeline diagram and it has helped him remember how to correctly show data hazards. While this is not "acceptable collaboration," the right thing for Bob to do is complete the problem and include a note explaining that he briefly saw Alice's solution. Bob's note should clearly indicate how seeing Alice's solution impacted his own solution so the instructors can take that into account when assessing his understanding.
- *(ECE 5740)* Bob is preparing his position paper comparing various network topologies and he is a little confused on the trade-off between using a ring vs. crossbar topology. Bob knows Alice is also working on her position paper, so they meet to brainstorm various advantages and disadvantages of the two topologies. They discuss the associated reading and sketch on a whiteboard a list of advantages and disadvantages. Bob and Alice then independently write their position papers based on this discussion but in their own words. Both Bob and Alice include an acknowledgment in their position paper of their interactions. At no time do the students share any actual writing.

Examples of unacceptable collaboration:

- Bob and Beth are struggling to complete a lab assignment which requires implementing a direct-mapped cache. They talk with Alice and Adam and learn that both groups are really struggling. So the four students get together for a joint coding session. Each student works on one module in the cache, then they combine the modules together to create the final working direct-mapped cache. *The four students share and copy each other's code often to finish the assignment.* Each group submits the final code independently. Each group acknowledges the help it received from the other group, but it doesn't matter since they explicitly shared code.
- Bob and Beth are having difficulty figuring out difficult test cases for their pipelined processor. They make a post on Ed to see if anyone has some general ideas for tricky corner cases. Alice and Adam figured out an interesting test case that ensures their pipelined processor correctly forwards the address to a store instruction, so *Alice and Adam show their test code to Bob and Beth in the computer lab.* Bob and Beth use this test code and include it in their submission. Bob and Beth include an acknowledgment of the help they received from the other group, but it doesn't matter since they explicitly copied code.
- Bob and Alice are seated close to each other in the final exam. Bob is struggling on a problem requiring drawing a detailed pipeline diagram. While stretching, he accidentally sees the pipeline diagram on Alice's solution. This was an unintentional mistake, but now he has seen the pipeline diagram, and it has helped him remember how to correctly show data hazards. *Bob completes the problem and submits his solution representing it as his own independent work.*
- Anna cannot make today's lecture since she has an extracurricular commitment. *Anna asks Bart to complete two in-class paper quizzes and to put Anna's name on the second quiz.*
- *(ECE 5740)* Bob is preparing his position paper comparing various network topologies and he is a little confused on the trade-off between using a ring vs. crossbar topology. Bob knows Alice is also working on her position paper, so they meet to brainstorm various advantages and disadvantages of the two topologies. They discuss the associated reading and sketch on a whiteboard a list of advantages and disadvantages. Bob and Alice then independently write their position papers based on this discussion but in their own words. *They then meet to go over their position papers, and each student reads the other student's work.* They then improve their

position papers based on what they read in the other student's work. Both Bob and Alice include an acknowledgment in their position paper of their interactions, but it doesn't matter since they explicitly shared their writing.

- *(ECE 5740)* Bob has just finished his 15-minute position defense and bumps into Alice in the hall. *Bob tells Alice in detail exactly how the discussion went including the questions the instructor asked.* This gives Alice an unfair advantage and means the position defense does not accurately represent her own understanding.

Notice that **the key is that students should not share the actual solutions or code with each other unless expressly permitted by the course instructors.** Consulting with your fellow students is fine and is an important part of succeeding in this course. If the vehicle for consulting is a whiteboard (and you avoid writing the actual solution on the whiteboard) then you should be fine.

11.G Copyright Policy

11.H Accommodations for Students with Disabilities

In compliance with the Cornell University policy and equal access laws, the instructor is available to discuss appropriate academic accommodations that may be required for students with disabilities. Requests for academic accommodations are to be made during the first three weeks of the semester, except for unusual circumstances, so arrangements can be made. Students are encouraged to register with Student Disability Services to verify their eligibility for appropriate accommodations. Note that students cannot simply rely on Student Disability Services to email the instructor without actually speaking to the instructor in person. Students must speak with the instructor in person during the first three weeks to discuss their accommodation.

12. Online and Computing Resources

We will be making use of a variety of online websites and computing resources.

- **Public Course Website –** `http://www.csl.cornell.edu/courses/ece4750` is the main public course website with course details, updated schedules, reading assignments, and most handouts. We intend for all course content to always be available on Canvas. The public course website is just for public access to some of this content.
- **Canvas –** We will be using Canvas to manage course content, assignment submission, and grade distribution.
- **Ed –** We will be using Ed for all announcements and discussions on course content, lab assignments, and projects. The course staff is notified whenever anyone posts on the forum and will respond quickly. Using the forum allows other students to contribute to the discussion and to see the answers. Use common sense when posting questions such that you do not reveal solutions. Please prefer posting to Ed instead of directly emailing the course staff unless you need to discuss a personal issue.

- `ecelinux` **Servers –** The ECE department has a cluster of Linux-based servers which we will be using for the programming assignments. You can access the `ecelinux` servers remotely using PowerShell, Mac Terminal, VS Code, X2Go, MobaXterm, or Mac Terminal with X11. More information about accessing the ECE computing resources is available on Canvas.