NTNU

Det skapende universitet

TDT4255 Computer Design

Lecture 0: Introduction

Joseph Rogers

Who are you and why are you taking this course?

We want to improve

We will experiment

We need feedback

TDT4255 Computer Design

- TDT4255 Computer Design is a thorough review of scalar processor core design:
  - Detailed discussion of simple pipelined CPUs
  - Introduction to advanced techniques such as out-of-order execution, branch prediction and speculative execution
- Hands-on approach (i.e., student-active learning)
  - Exam counts 100%, but lots of learning activities must be completed to be allowed to take the exam.
  - Exam will (most likely) be a school exam.
- Complementary to TDT4260 Computer Architecture
  - TDT4255 focuses on the detailed implementation of ILP-focused processor cores (ILP = Instruction Level Parallelism)
  - TDT4260 focuses on computer organization in general (e.g., multi-cores, memory-systems, accelerators, etc.). The ILP-focused processor cores we focus on in TDT4255 are key building blocks in TDT4260.

Course Staff

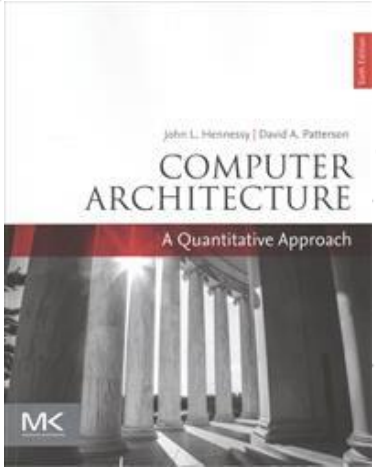
- Lecturer/Coordinator Joseph Rogers
  - PhD student since 2019, IDI, NTNU.
  - Master's 2019, Uppsala University.
  - Previously at University of Colorado, and Purdue University.
  - Research on performance modeling heterogeneous architectures, e.g., how will different applications perform on different types of devices.

- Teaching Assistant (TA) David Metz
  - PhD student

- Teaching Assistant (TA) Babis Bezaitis
  - PhD student

Preliminary Curriculum

- Hennessy and Patterson, Computer Architecture – A Quantitative Approach, 6th Edition:
  - Chapter 1
  - Chapter 2.1, 2.2 and 2.3
  - Chapter 3 (except 3.2, 3.7, 3.11 and 3.12)
  - Appendix A
  - Appendix B
  - Appendix C (except C.6)
- Course Exercise materials
- Course Lecture slides



Information and Lecture Plan

- We will use Blackboard and MS-Teams to share course information
  - We expect that you set up a notification or check manually
- Piazza for project and exercise discussions.
  - https://piazza.com/ntnu.no/fall2023/tdt4255
- Scoring:
  - Exam counts 100% of the grade
  - Comprehensive points system
    - You get points for performing learning activities
    - You need 350 points to be allowed to take the exam
  - Lecture setup:
    - An “example that explains everything”
    - Q/A session about the videos and theory questions

Why do we have the points system?

- Learning new things is uncomfortable
  - "People respond to incentives"
- What you think you learn versus what you actually learn
  - Paradox: Some students implement (nearly) perfect processors, but don't understand the theory
  - Learning things requires feedback and reflection
- The learning activities aims to be formative
  - You should learn from doing them
  - We grade based on effort (i.e., thinking) rather than "correctness" (the exception is the final delivery of the processor which is only correctness)
  - The exam is normative (i.e., we assess to what extent you have reached the learning objectives).

Course Overview

- Theory Track (8 Lecture Topics)
  1. You watch the videos in your own time at your own pace.
  2. You do a pass over the exercise assignment. Do the questions that are easy and think a bit about the ones that are hard.
  3. We meet in the lecture slot.
     - I go through a prepared example that will clarify how stuff works (which may address some issues).
     - You tell me what you are still struggling with, and I explain.
  4. You answer the remaining questions on the exercise (which have now magically become much easier) and hand in the exercise.
  5. You (anonymously) grade the assignment of some other student.
- Practical Track (2 Assignments)
  1. Chisel Introduction Exercise
  2. RISCV CPU Project
     - Monday and Wednesday Lab Sessions for help and showing progress.

Why peer evaluation?

- Learning requires thinking about things
- Receiving feedback is essential as it makes you think about things you otherwise would not have thought about
  - Rarely happens in university courses because instructors don't have time to provide feedback to all students (e.g., teaching is only 40% of my job and this course is only one of many teaching activities)
- Critically assessing one's own answers is critical for learning
  - … but student assignments are (typically) fire-and-forget
- Solution: Peer-evaluation

How to do peer evaluation

- Theory assignments should be formative
  - You should learn from doing them
- You should hence give points based on effort (i.e., thinking) rather than “correctness”
  - The exam provides normative assessment (i.e., we assess to what extent you have reached the learning objectives).
  - A solution sketch will be provided (but it will often not be the only “correct” answer)
- Be (overly) polite!
  - Statements that can be perfectly fine face-to-face can come across as borderline abusive when delivered anonymously and in writing.
- Let us know if you spot potential for improvement!

Peer Assessment Examples

- **Example 1**
  - Student A has answered Question X, which can be awarded up to 2 points, but the answer is wrong.
  - Student B hence awards 2 points and feedback explaining what went wrong and how Student A can do better next time.

- **Example 2**
  - Student A has answered one out of two subquestions in Question Y (which can be awarded 2 points).
  - Student B hence awards 1 point since Student A made no effort at answering the second subquestion and explains why 1 point was deducted in the feedback.

- **Example 3**
  - Student A has answered three out of three subquestions in Question Z and is all correct (3 points available).
  - Student B hence awards 3 points. While no comment is necessary, it is good form to provide positive feedback (e.g., "Well done!").

Practical Track

- Two exercises to solve individually:
  1. Warm-up exercise: Get used to hardware development with Chisel
     - Points awarded for demonstrating your implementation to the TA
     - Critical training for the project – take it lightly at your own risk!
  2. Project: Design your own pipelined RISC-V CPU from scratch
     - There will be three milestones where you demonstrate key functionality to the TA
       - Forces you to start early (as starting late will not work out)
       - Gets you into the habit of asking questions
     - For the final evaluation, we assess implementation quality only (i.e., features implemented and the correctness of these features)
     - **Note: You will not be allowed to take the exam without delivering as somewhat functional CPU (more about this later)**
- David and Babis will introduce the exercises in detail in the first exercise lecture (Thursday next week). Check out Blackboard and start familiarizing yourselves with the content now.

Points system

- You can get points for the following learning activities:
  - Attending a theory lecture: 10 points
  - Submitting a theory exercise: up to 10 points
  - Performing peer evaluation: 5 points
  - Attending the Chisel lecture: 20 points
  - Demonstrating the warm-up exercise: up to 20 points
  - Attending the project introduction lecture: 20 points
  - Reaching a project functionality milestone: up to 30 points
  - Correctness of your RISC-V processor: up to 100 points
- You need 350 out of 450 possible points to be allowed to take the exam

Stay at home if you are sick!

Referansegruppe!

How to Ace this Course

- Participate in the lectures and be prepared
  - Watch the videos beforehand
  - Try to answer the questions beforehand
- Read the book
  - I recommend both read relevant chapter(s) of the book and watch the videos for each topic before trying to answer the questions
- Ask for help when appropriate
  - Spend enough time on the problem before asking for help.
  - Asking to early is bad – as your question is typically imprecise and you may not fully understand the answer – and spending to much time is equally bad – as you wasted time you could have spent on learning other things.
  - Knowing when to ask for help is a trainable skill (so try and reflect).

Questions?