CSE 390B, Winter 2022

Building Academic Success Through Bottom-Up Computing

Midterm Exam Review, Project 5 Overview

Exam Review Session, Project 5 Demo, Hack CPU Logic

Lecture Outline

- **CSE 390B Midterm Brainstorm**
- CSE 390B Exam Review Session
  - Circuit Design, Writing Assembly, Tracing Assembly
- Project 5 Overview
  - Timed Mock Exam, Building a Computer, Project Tips
- Hack CPU Logic Example: writeM

CSE 390B Midterm Brainstorm

- Based on what we’ve covered thus far in class, what are topics, concepts, questions that you might expect to show-up on next week’s midterm?

Lecture Outline

- CSE 390B Midterm Brainstorm
- **CSE 390B Exam Review Session**
  - **Circuit Design, Writing Assembly, Tracing Assembly**
- Project 5 Overview
  - Timed Mock Exam, Building a Computer, Project Tips
- Hack CPU Logic Example: writeM

Review Session Activity

- Work through three practice problems in teams
  - Circuit Design, Writing Assembly, Tracing Assembly

- For each problem:
  - Step 1: Spend 7-8 minutes working on the problem individually
  - Step 2: Spend 7-8 minutes as a group comparing and contrasting how you solved the problem
  - Step 3: Spend 5 minutes as a group discussing what tips you would recommend for approaching this type of exam problem and write it on the whiteboard
  - Step 4: Identify who in your group will demonstrate solving the problem to the class and who will present tips for approaching this type of problem

What now?

- Based on your experience with this exercise, how does it inform how you approach your studying?

- What resources can you utilize to help you deepen your understanding?

Previous CSE 390B Midterms

- We have three old midterms from previous quarters
  - Spring 2020 is likely more difficult than the midterm this quarter
  - Winter 2021 & Spring 2021 are more similar to what this quarter's midterm will look like

- Spring 2020 midterm recommended to become familiar with problem types

- Winter and Spring 2021 midterms recommended for practicing a timed exam
  - Set a timer for 50 minutes and take the exam in its entirety
  - Doing so help you practice time management

Lecture Outline

- CSE 390B Midterm Brainstorm

- CSE 390B Exam Review Session
  - Circuit Design, Writing Assembly, Tracing Assembly

- **Project 5 Overview**
  - **Timed Mock Exam, Building a Computer, Project Tips**

- Hack CPU Logic Example: writeM

Project 5: Overview

- Timed Mock Exam Problem

- Build a Computer
  - `LoadAReg.hdl`, `LoadDReg.hdl` (Easier)
  - `JumpLogic.hdl` (Medium)
  - `CPU.hdl` (Harder)
  - `Computer.hdl` (Easier)

Project 5: Timed Mock Exam Problem

- Your group will meet for a 30-minute session to do one mock exam problem
    - Your group's mock exam problem will be emailed right before your session

- Your 30-min session will include:
    - Set-Up: 5 minutes
    - Mock Exam Problem: 10 minutes
    - Debrief & Reflection: 15 minutes

- Complete and submit the reflection questions

Project 5 Tips

- `CPU.hdl`: We provide an overview diagram, but there are several details to fill in, especially control
  - Crucial to draw your own detailed diagram first
  - Handling jumps will require a lot of logic; sketch out all cases
  - Chapter 4 and 5 are going to be extremely useful

- Multi-Bit Buses: MSB to the left, LSB to the right
  - Important to keep in mind when taking apart the instruction

- Debugging: Consult .out and .cmp files when getting buggy output, then look at internal wires in simulator
  - See also the "Debugging tips" section of the spec

Hack CPU Logic

- How do we determine the unimplemented logic for the CPU (all of the c’s in the diagrams)?

- Need to refer to the assembly specification

- Project 5 will requires understanding of textbook chapters to determine how to use the instruction bits to implement the control logic
  - Textbook sections 4.2.2, 4.2.3, and 5.3.1 are especially helpful

Hack CPU Logic Workflow

- Step 1: What do we pay attention to?
  - Usually, will be some combination of instruction bits or intermediate outputs
  - These are the “inputs” to your sub-problem
- Step 2: Determine logic for the part you are working on
  - Uses the “inputs” from step 1
  - Usually requires reading a relevant section of the textbook/assembly specification

Lecture Outline

- CSE 390B Midterm Brainstorm

- CSE 390B Exam Review Session
  - Circuit Design, Writing Assembly, Tracing Assembly

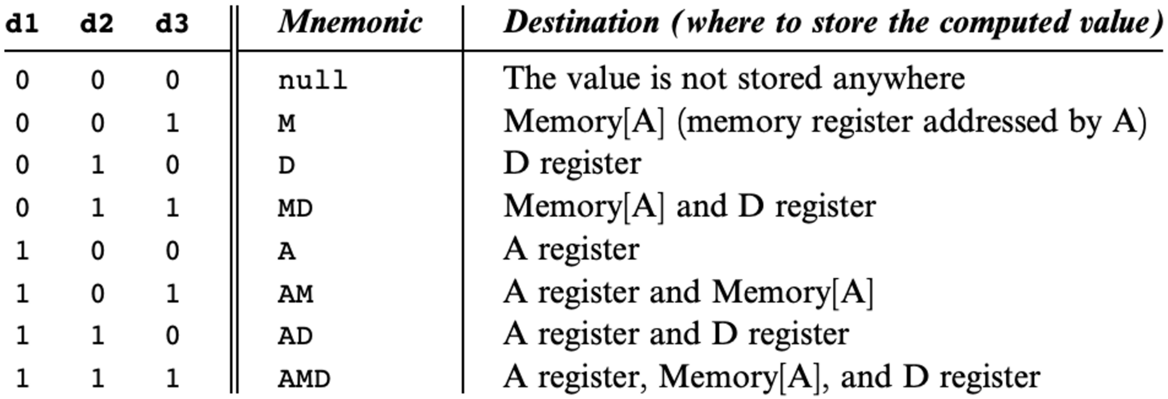
- Project 5 Overview
  - Timed Mock Exam, Building a Computer, Project Tips

- **Hack CPU Logic Example: writeM**

Hack CPU Logic Example: writeM

- Example: Determine when writeM should be set to 1

- Step 1: What do we pay attention to?
  - **`writeM`** is related to whether we write to memory or not
  - We need to look up the destination bits specification from Chapter 4



d1	d2	d3	*Mnemonic*	*Destination (where to store the computed value)*
0	0	0	null	The value is not stored anywhere
0	0	1	M	Memory[A] (memory register addressed by A)
0	1	0	D	D register
0	1	1	MD	Memory[A] and D register
1	0	0	A	A register
1	0	1	AM	A register and Memory[A]
1	1	0	AD	A register and D register
1	1	1	AMD	A register, Memory[A], and D register

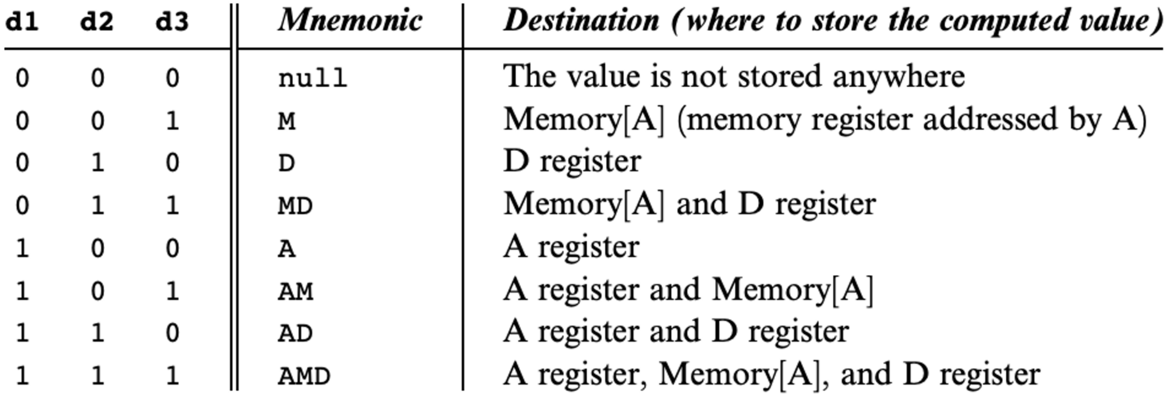
Figure 4.4 The *dest* field of the *C*-instruction.

Hack CPU Logic Example: writeM

- Example: Determine when writeM should be set to 1

- Step 2: Determine logic for specification
  - Read the “Destination Specification” section of Chapter 4
  - Instruction bits:

```
1 1 1 a c1 c2 c3 c4 c5 c6 d1 d2 d3 j1 j2 j3
```



d1	d2	d3	*Mnemonic*	*Destination (where to store the computed value)*
0	0	0	null	The value is not stored anywhere
0	0	1	M	Memory[A] (memory register addressed by A)
0	1	0	D	D register
0	1	1	MD	Memory[A] and D register
1	0	0	A	A register
1	0	1	AM	A register and Memory[A]
1	1	0	AD	A register and D register
1	1	1	AMD	A register, Memory[A], and D register

Figure 4.4 The *dest* field of the *C*-instruction.

Hack CPU Implementation: Logic sub-chips

- We provide you with 3 sub-chips and tests that implement the control logic for the A Register, D Register, and PC
  - **`LoadAReg`** contains logic for loading the A Register
  - **`LoadDReg`** contains logic for loading the D Register
  - **`JumpLogic`** contains logic for determining if the PC should load, jump, or increment
- Implement and test these first, then use them in your CPU implementation
  - Intended to help you narrow the scope of bugs

Post-Lecture 10 Reminders

- What's in store for Week 6?
  - CSE 390B mock midterm exam practice
  - **CSE 390B Midterm Exam: 2/10 at 1:30pm during in-person lecture**

- Project Reminders
  - Project 3 grades released on Gradescope
  - **Project 4 due tonight (2/3) at 11:59pm PST**
  - Project 5: Building a Computer Part II, Timed Mock Exam to be released today
    - Due in two weeks (2/17) at 11:59pm PST