Faculty of Media Engineering and Technology
Dept. of Computer Science and Engineering
Assoc. Prof. Milad Ghantous

CSEN 702: Microprocessors
*Winter 2023*

# Project requirements

You are required develop a simulator (in java, C) or any other language you're comfortable with for the Tomasulo algorithm.

- Your simulator should accept inputs (MIPS instructions in assembly format, either by writing them one by one or by loading a text file containing the code) and show **step by step** (cycle by cycle) how these instructions are executed as well as the content of each reservation station/buffer, the register file, the cache and the queue.)

- Your simulator should handle all types of codes, with and without a loop, and that might contain a combination of RAW, WAR and WAW hazards.

- A command line interface (CLI) is the minimum required and the better the UX with the CLI, the higher the grade.
- Developing a GUI is considered a bonus.

- Your simulator should include ALU ops (FP adds, subs, multiply, divide), (integer ADDI or SUBI needed for loops), loads and stores and branches (BNEZ is enough). You can take any assumptions about the cache latency. Don't worry about the cache misses. Also always consider the effective address is calculated. You can write the instructions as such L.D F2,100 where 100 is the effective address directly. Address clashes are ignored.

- The user should be able to input the latency of each type of instruction before we start simulating, EXCEPT ADDI and BNEZ, assume they take 1 cycle of execution each. No prediction is required.

- The user should be able to select the size of all stations and buffers.

- Cache and Register file can be pre-loaded with some values or allow the user to load them.

- N.B. Make all instructions (integer and floating) enter the Tomasulo architecture, such as DADD and BNEZ, which can be considered add/sub instructions and hence they enter the add/sub reservation stations.

- When two instructions wish to publish their result on the bus in the same cycle, you need to handle that and explain how you did so.

Deliverables:

- A report explaining the steps you did to develop and run and test your code. The report should explain your approach, code structure, and test cases.
- A demo (while evaluating the project with your TA).

Warning:

- **Copying a project (or sharing codes) from the internet or from each other will result in a zero for both parties, the copying and the copied.**
- **It's better to submit a non-complete project than to submit a stolen one.**

## Notes and deadlines:

The work should be divided equally among all group members according to the complexity of the project and its content. An equal grade for all group members is not guaranteed. In the evaluation, be prepared for any question. All group members should know how to answer.

**Submission: December 10, 2023.**

Project submission by email (link to a google drive folder where all your deliverables are) and not an attachment. Email address will be provided later.

**Evaluations: Starting Dec 10, 2023**

Evaluations will start on December 10, 2023 as well. A reservation system will be provided to you. While evaluating, we should be able to try out any combination of instructions, and your simulator should work.