ECE552 PRA0102 - Station 8
Ting-Hao (Tim) Cheng (997289090)
Isaiah Tan (999004937)

total numbers of cycles with tomasulo

benchmark	gcc.eio	go.eio	compress.eio
number of cycles	1814117	1852942	1979818

Tomasulo stage descriptions

CDB_To_retire

This function simply updates the maptable for the instruction on the CDB, and signals dependent instructions waiting in the issue stage that the output value they've been waiting for is ready. Then, free the CDB.

execute_To_CDB

This function finds all instructions that have completed execution, and for instructions that do not require the CDB, the used RS and FU is cleared (they don't have output registers, don't have dependent instructions). The oldest of instructions that require the CDB is placed on the CDB, and it's RS and FU is cleared as well.

issue_To_execute

This function first finds a free FU, and searches for instructions with no RAW dependency in the corresponding RS (int or float). The oldest of the found valid instructions is placed into the free FU, and the execution cycle is updated. This is done for both INT and FP FU/RS.

dispatch_To_issue

This function gets one instruction ready to start issuing (head of queue). If it's a control instruction, just clear the instruction from the queue, and move the head to the next instruction. If the instruction uses the floating point FU, reserve an available FP RS. use INT RS for all other instructions (load/store/int). Once a RS has been reserved, update the instruction's issue cycle, and update the output register in the maptable and collect any tags from the maptable (for input registers). Remove the instruction from queue and shift the head.

fetch

This function simply put an instruction on the queue from the trace. It is placed after the tail of the previously fetched instruction. If the instruction retrieved is a trap, ignore and get the next one. Loop around queue array when necessary.

fetch_To_dispatch

This function simply updates one instruction's dispatch cycle. The instruction is indexed in the queue, by finding the tail (which holds the last dispatched instruction) and dispatches the following instruction

ECE552 PRA0102 - Station 8
Ting-Hao (Tim) Cheng (997289090)
Isaiah Tan (999004937)

Helper functions

a helper function was created to check if a given array is empty. This is done to simplify checking if simulation is complete. (Checking that the instruction queue is empty, and that all RS and FUs are empty).

Testing correctness

We tested our code by comparing the output of our code (by using `print_all_instr()`), and comparing to a result we expected (by doing the Tomasulo algorithm on paper). We would only use up to 15-20 instructions when testing each instruction. For debugging unexpected results, we used print statements to show the RS/maptable/FU statuses after each cycle, and which instructions are in the pipeline in each cycle. This allowed us to check which functions contained bugs that produce unexpected results.

Toughest bugs

1. Segfaults from invalid pointer. Sometimes mistakenly dereferencing an invalid pointer without checks would create segfaults, which can be traced with gdb.
2. Subtle mistyped statements causing runtime bugs. Such code is hard to detect, if it compiles and runs but creates incorrect output. This can be hard to track without reading the code line by line, and finding the right line may require printing outputs (or gdb) to trace the bug.

Work Delegation

Isaiah:
coded initial CDB_To_retire
debugged execute_To_CDB, issue_To_execute, dispatch_To_issue, fetch_To_dispatch
lab report

Tim:
coded initial execute_To_CDB, issue_To_execute, dispatch_To_issue, fetch, fetch_To_dispatch
debugged execute_To_CDB, issue_To_execute