CS6501: Great Works in Computer Science
January 31, 2013
Wei Wang

An Efficient Algorithm for Exploiting Multiple Arithmetic Units

Robert M. Tomasulo 1967

http://inst-tech.engin.umich.edu/media/index.php?sk=tomasulo

1. Background
    a. What are the components of a Turing machine?
        - Tape
        - Symbols
        - Configurations
    b. What are the components of the Von Neumann architecture?
        - Functional units
        - Register file
        - Memory
        - Program counters
        - Disc and I/O devices
    c. The implied rules for Turing machine and Von Neumann architecture?
        - One step / instruction at a time
        - Know that to execute after one step / instruction finishes
    d. Is Von Neumann architecture (or even Turing machine) the only way to build computing machines?
        - No, data-flow architectures. (No sure if data-flow architecture is a Turing machine, although it is definitely definitely as powerful as a Turing machine.

2. Motivation of Tomasulo’s algorithm

    a. The three optimizations

        - Read/write buffers
        - Overlapping of memory accesses and executions
        - Pipelining

    b. For smaller cycles
    Functional units should have one single functions (e.g., ADD) so that they can produce results in one small CPU cycle.

    c. Multiple functional units (FUs)
    Some functional units may be not used during execution.

    d. Software or Hardware solutions
    Hardware solutions are transparent to users.

    e. Preserving the dependencies

        - RAR (not considered by Tomasolu's algorithm)
        - RAW
        - WAW (not considered by Tomasolu's algorithm)
        - WAR

3. Definitions and pre-requisitions

    a. Floating point units

    b. The instructions: what is a sink?
    The memory where the result of an instruction is saved.

    c. Three key features

        - Honor data dependencies
        - Increase functional unit utilization
        - Pure hardware solution

    d. Three functional requirements

        - Recognize data dependencies
        - Obey data dependencies
        - Increase simultaneous executions

4. Busy bit and reservation station

    a. Busy bit scheme

    b. The software solution: why it fails?

    c. Reservation station

    d. How three functional requirement s are met?

        - Busy bits recognize data dependencies
        - Busy bits preserve data dependencies
        - User allocation registers to improve FU utilization

e. What happens if software provides no help?

5. Common data bus (CDB) and tagging

   a. The common data bus

   b. Tagging

   c. How the functional requirements are met

      - Busy bits recognize data dependencies
      - Tags preserve data dependencies
      - CBS and tags improve FU utilization

6. The limitations of Tomasulo’s algorithm

    a. The first limitation of Tomasulo’s algorithm
       Complex to impelement.

    b. The second limitation of Tomasulo’s algorithm

        - Types of interrupts that must be precise

        b..a) I/O interrupts

        b..b) page fault

        - What are stored during context switches

        b..a) Register file, PC, Memory etc.

        - How Tomasulo’s algorithm messed up?

        b..a) Instructions does not executed in order

        b..b) Register file is not updated in order

        b..c) Middle results may be discarded

        b..d) Memory is not updated in order

        - Implementing precise interrupts

        b..a) Reorder-buffer
        Results are buffered and in-orderly written back to register file and memory

        b..b) History file
        Data replaced by out-order writes are buffered in history file.

        b..c) Future file
        Add a new register file to store out-order writes. The old register file is only used for in-order writes.

c. The third limitation of Tomasulo's algorithm

- Which structures limit Tomasulo's algorithm's power?

c..a) Reservation station

c..b) Functional units

~~c..c) Register file~~

c..d) Load/store Buffers

- Making Tomasulo's algorithm better
  More FUs and larger reservation station

- Why multi-core?
  More power efficiency, and exploits Thread-Level Parallelism.

- Instruction-level parallelism: Is Tomasulo's algorithm still good?
  Instruction-level parallelism is limited. Therefore, more FUs and larger RS won't help.

- The implication of Tomasulo's algorithm for today's research
  Algorithms have to changed to exploit Thread-Level Parallelism.