IN1006 Systems Architecture 2024 - 2025

Tutorial 6: Pipelining and Parallelism

Solutions

1 *Does pipelining improve latency or throughput or both? Why?*

Pipelining does not change the time it takes a given instruction to execute (latency) but it does allow more instructions to be pushed thorough the processor in a given time (throughput) as the stages are more fully utilised.

2 *Describe the three types of hazards that can reduce the effectiveness of pipelined processors.*

- Structural Hazards - Hardware cannot support a combination of instructions in the same clock cycle, e.g. two simultaneous memory accesses.
- Control Hazards - Loading instructions into pipeline before the result of a decision is known, e.g. loading instructions after a branch.
- Data Hazards - Instruction depends upon the results of a previous instruction still in the pipeline, e.g. compound math expressions.

3 *What type of hazard does the technique called Branch Prediction overcome in a pipelined processor? How does it achieve this?*

It is a way of overcoming control hazards. Assume an outcome for decisions (e.g. branch always fails) and load pipeline accordingly.

- If the prediction is correct then the pipeline remains full at all times and the decision costs nothing.
- If the prediction is incorrect then the pipeline will stall (as before) and new instructions have to be loaded.

4 *What type of hazard does the technique called Delayed Decision overcome in a pipelined processor? How does it achieve this?*

Delayed Decision rearranges instructions such that an instruction that does not affect the decision is executed after the decision filling the bubble that would occur if the pipeline were to stall. It may be useful to refer to the example in the lecture slides.

5 *A processor designer is considering whether to use a shallow or deep pipeline. What are the (dis)advantages of these two decisions?*

The deeper the pipeline is, the higher the throughput. However a deeper pipeline may require more transistors (as the datapath is more involved). Also the effects of hazards are more severe on deep pipelines, thus needing even more transistors to implement say branch prediction, to achieve full use of the pipeline.

6 *The two sequential instructions from a program (shown below) are to be executed on a MIPS pipelined processor; they create a problem for the pipeline. What is the problem?*

```
add    $s0, $t0, $t1
add    $t2, $s0, $t3
```

Additionally name & explain two techniques used to overcome the problem?

This is an example of a data hazard. The pipeline must be **stalled** and the problem arises **because** of **instruction dependencies** within the pipeline (the second add needs the value in $s0, but this will only become available at the end of the 'fetch-execute-store' cycle.

The first way of dealing with this is by having the compiler reorder instructions so that cases such as the above are reduced in frequency (that that this requires the complier to make assumptions about the pipeline structure that may not be true in other members of the same processor family).

The problem can also be **overcome by forwarding** where extra circuitry allows the intermediate result from executing the first add instruction is passed onto the execute phase for the second add instruction (i.e. allowing the second instruction to continue without stalling the pipeline or waiting for the first instruction to complete.)

7 *Consider a dual-core processor. What effect on execution time and throughput would each of the following have on this processor, compared with a single core processor?*

a) One single-threaded program.
b) Two single-threaded programs.
c) One multi-threaded program.

a) No effect on either – it can only execute on one core
b) No effect on execution time (each core is no faster), but since each program can run concurrently on each core, throughput is increased.
c) Increases both execution time and throughput, as both cores can work on the

8 *What is a SIMD unit? What types of program would benefit from this?*

SIMD = Single Instruction Multiple Data
One common application of SIMD is for matrix calculations that suit the data parallel nature of SIMD. Examples include graphics and scientific computing.

9 **(Advanced):** *Rewrite your most complex assembly language program in MIPS ASM. Look at the code and work out how much parallelism could be exploited and at what level. Analyse the code for possible pipeline hazards assuming a 5 stage pipe.*

No solution given since this is an application of many different aspects of the lectures and will be specific to the individual code. Come talk with me if you want feedback on what you've done.