4.14 Fallacies and pitfalls

Fallacy: Pipelining is easy.

Our books testify to the subtlety of correct pipeline execution. Our advanced book had a pipeline bug in its first edition, despite its being reviewed by more than 100 people and being class-tested at 18 universities. The bug was uncovered only when someone tried to build the computer in that book. The fact that the Verilog to describe a pipeline like that in the Intel Core i7 will be hundreds of thousands of lines is an indication of the complexity. Beware!

Fallacy: Pipelining ideas can be implemented independent of technology.

When the number of transistors on-chip and the speed of transistors made a five-stage pipeline the best solution, then the delayed branch (see the Elaboration in COD Section 4.3 (Building a datapath)) was a simple solution to control hazards. With longer pipelines, superscalar execution, and dynamic branch prediction, it is now redundant. In the early 1990s, dynamic pipeline scheduling took too many resources and was not required for high performance, but as transistor budgets continued to double due to **Moore's Law** and logic became much faster than memory, then multiple functional units and dynamic pipelining made more sense. Today, concerns about power are leading to less aggressive and more efficient designs.

Pitfall: Failure to consider instruction set design can adversely impact pipelining.

Many of the difficulties of pipelining arise because of instruction set complications. Here are some examples:

- Widely variable instruction lengths and running times can lead to imbalance among pipeline stages and severely complicate hazard detection in a design pipelined at the instruction set level. This problem was overcome, initially in the DEC VAX 8500 in the late 1980s, using the micro-operations and micropipelined scheme that the Intel Core i7 employs today. Of course, the overhead of translation and maintaining correspondence between the micro-operations and the actual instructions remains.
- Sophisticated-addressing modes can lead to different sorts of problems. Addressing modes that update registers complicate hazard detection. Other addressing modes that require multiple memory accesses substantially complicate pipeline control and make it difficult to keep the pipeline flowing smoothly.
- Perhaps the best example is the DEC Alpha and the DEC NVAX. In comparable technology, the newer instruction set architecture of the Alpha allowed an implementation whose performance is more than twice as fast as NVAX. In another example, Bhandarkar and Clark [1991] compared the MIPS M/2000 and the DEC VAX 8700 by counting clock cycles of the SPEC benchmarks; they concluded that although the MIPS M/2000 executes more instructions, the VAX on average executes 2.7 times as many clock cycles, so the MIPS is faster.

PARTICIPATION ACTIVITY 4.14.1: Fallacies and pitfalls.

1) Designing and implementing a pipelined computer is simple.
   - True
   - False

2) Current pipeline designs are the same as pipeline designs from the 1980s.
   - True
   - False

3) Pipelining and control hazards can be difficult when instructions vary widely in length or when addressing modes are complex.
   - True
   - False

Provide feedback on this section