Lesson 02:
Pipeline Performance
Objective

- Learn how to compute the instruction pipeline performance and throughput
- Learn datapath logic circuit uneven latencies,
- Understand the effect of buffers and
- Understand the effect of large number of stages on performance
Cycle Time
First impression about cycle time of pipelined processor

- First impression—The number of cycles required to execute a given set of instructions is now also five, total clock period of all 5 stages same as before for each instruction
- Each instruction taking 5 clock cycles to complete in five stage pipeline
- Pipelining doesn’t increase the performance of the processor
First impression about Cycle Time of Pipelined Processor

- Pipelining a processor generally increases the number of clock cycles it takes to execute a program.
- Some instructions get held up in the pipeline waiting for the instructions that generate their inputs to execute.
Unpipelined processor

- Fetched an instruction
- Decode instruction
- Read the instruction’s inputs
- Perform the computations
- Write the result back

One clock Cycle
Pipelined five stages processor

- Fetched an instruction
- Decode
- Read Inputs
- Compute
- Write Result

Pipeline Stages:
- One clock Cycle
- One clock Cycle
- One clock Cycle
- One clock Cycle
- One clock Cycle
Facts about cycle Time of Pipelined Processor

- The performance benefit of pipelining comes from the fact that, because less number of the logic gates in the datapath gets executed in a single cycle in a pipelined processor.
- Single Clock Cycle of period
- Cycle Time $T_{\text{unpipelined}} = T$
- Because less number of the logic gates in the datapath there are though five Clock Cycles but each has period Cycle Time $T'$ about $T = T' \div 5$
Cycle Time of Pipelined Processor
Cycle Time of Pipelined Processor

- Pipelined processors can be clocked at a fast clock rate and thus can have reduced cycle times (more cycles/second by a fast clock) than unpipelined implementations of the same processor.
Throughput

• Since the pipelined processor has a throughput of one instruction per cycle, the total number of instructions executed per unit time is higher in the pipelined processor, giving better performance.
Factors for Cycle time of a pipelined processor

(i) The cycle time of the unpipelined diversion of the processor
(ii) The number of pipeline stages
(iii) How evenly the datapath logic divided among the stages
(iv) The latency of the pipeline latches (buffers)
Clock Period

- Assume the logic divided evenly among the pipeline stages
- Cycle Time_{pipelined} = (Cycle Time_{unpipelined} ÷ Number of Pipeline Stages) + Pipeline Latch Latency
A slightly higher clock rate

- Can be achieved by taking advantage of the fact that a \( n \)-stage pipeline requires only \( n - 1 \) pipeline latches.
- Use enough additional logic to the last stage of the pipeline to make its total latency equal to that of the other pipeline stages including their pipeline latches.
Effect of Increasing Number of pipeline stages
Number of pipeline stages

- Each stage contains the same fraction of the original logic, plus one pipeline latch (buffer)
Increasing Number of pipeline stages

- The pipeline latch (buffer) latency becomes a greater and greater fraction of the cycle time
- Limiting the benefit of dividing a processor into a very large number of pipeline stages
Example

- Assume—an unpipelined processor version
  cycle time = 25 ns
- Assume—evenly divided pipeline stages
- Assume—each pipeline latch (buffer) has a latency of 1 ns?
Cycle time of the 5 stage pipelined version of the processor

- Cycle time for the 5-stage pipeline = \((25 \text{ ns}/5) + 1 \text{ ns} = 6 \text{ ns}\)
Cycle time of the 50 stage pipelined version of the processor

- For the 50-stage pipeline, cycle time = (25 ns/50) + 1 ns = 1.5 ns
Result Analysis

- In the 5-stage pipeline, the pipeline latch (buffer) latency is only $\frac{1}{6}$th of the overall cycle time.
- Pipeline latch (buffer) latency is $\frac{2}{3}$ of the total cycle time in the 50-stage pipeline.
Another way of looking at the result

- 50-stage pipeline cycle time = one-fourth that of the 5-stage pipeline, at a cost of 10 times as many pipeline latches
Data Path Logic
Data Path Logic

- Often, the datapath logic cannot easily be divided into equal-latency pipeline stages
Example of Data Path Logic

- Accessing the register from a register file in a processor might take 3 ns
- Decoding an instruction might take 4 ns
Dividing a datapath into pipeline stages

- Designer balances the need to have each stage have the same latency with the difficulty of dividing the datapath into pipeline stages at different points.
Dividing a datapath into pipeline stages

- Designer determines the amount of space that the latch (buffer) takes up on the chip
- Designer balance the amount of data that has to be stored in the pipeline latch (buffer)
Instruction decode logic

- Irregular, making it hard to split them into stages
- Other stages may logic generate a large number of intermediate data values that would have to be stored in the pipeline latch (buffer)
Data Path Logic sections efficient way

- Designer place the pipeline latch (buffer) at a point where there are fewer intermediate results, and thus fewer bits that have to be stored in the latch (buffer)
Effect of Data Path Logic on Clock cycle time of the processor

Schaum’s Outline of Theory and Problems of Computer Architecture
Copyright © The McGraw-Hill Companies Inc. Indian Special Edition 2009
Unequal-latency pipeline stages

- Clock cycle time of the processor = equal to the latency of the longest pipeline stage plus the pipeline latch (buffer) delay
- Since the cycle time has to be long enough for the longest pipeline stage to complete and store its result in the pipeline latch (buffer) between it and the next stage
Total latency of a longest pipeline stage
Example

- Assume—An unpipelined processor with a 25-ns cycle time is divided into 5 pipeline stages with latencies of 5, 7, 3, 6, and 4 ns
- Assume—the pipeline latch (buffer) latency = 1 ns
Solution

• The longest pipeline stage = 7 ns
• Adding a 1 ns pipeline latch (buffer) to this stage gives a total latency of 8 ns, which is the cycle time
Pipeline Latency of all stages
Pipeline Latency

- While pipelining reduces a processor's cycle time, thereby increase instruction throughput, it also increases the latency of the processor by at least the sum of all the pipeline latch latencies.
Pipeline Latency

• The amount of time that a single instruction takes to pass through the pipeline
• Product of the number of pipeline stages and the clock cycle time
Example

• Assume—An unpipelined processor a cycle time = 25 ns
• Assume—evenly divided into 5 pipeline stages using pipeline latches with 1-ns latency
Cycle time of the 5-stage pipeline

- Cycle time = \((25 \text{ ns}/5) + 1 \text{ ns} = 6 \text{ ns}\)
- Computing the latency of each pipeline—
  Multiply the cycle time by the \(n\)
- Latency = \(6 \text{ ns} \times 5 = 30\text{ ns}\) for the \(n = 5\)
Cycle time of the 50-stage pipeline

- Cycle time = (25 ns/50) + 1 ns = 1.5 ns
- Computing the latency of each pipeline—Multiply the cycle time by the $n$
- Latency = 1.5 ns $\times$ 50 = 75 ns for the $n = 50$
Result

- Shows the impact pipelining can have on latency, particularly as the number of stages grows.
- The $n = 5$ pipeline has a latency of 30 ns, 20 percent longer than the original 25-ns unpipelined processor.
- The $n = 50$ has a latency of 75 ns, 3 times that of the original processor.
Formula for Pipelines with uneven pipeline stages
Formula for Pipelines with uneven pipeline stages

- Same formula, although they see an even greater increase in latency, because the cycle time must be long enough to accommodate the longest stage of the pipeline, even if the other stages are much shorter
Example

- Assume—an unpipelined processor with a 25-ns cycle time divided into 5 pipeline stages with latencies of 5, 7, 3, 6, and 4 ns.
- Assume—the pipeline latch latency = 1 ns.
Latency of the pipeline

- Longest stage latency $= 7 \text{ ns}$
- Thus cycle time of $7 \text{ ns} + 1 \text{ ns} = 8 \text{ ns}$
- Total latency of the pipeline for 5 stages $= 8 \text{ ns} \times 5 = 40 \text{ ns}$
Summary
We learnt

- Compute the instruction pipeline performance and throughput
- Datapath logic circuit and uneven latencies
- Effect of buffers
- Effect of large number of stages on performance
End of Lesson 02
Pipeline Performance