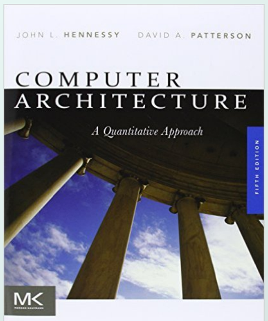


Processor Performance and Parallelism

Slides by Yashwant Malaiya

Limited content from:
Computer Architecture
A Quantitative Approach
Hennessy, Patterson

Processor Execution time

$$\text{Clock Cycles} = \text{Instruction Count} \times \text{Cycles per Instruction}$$
$$\text{CPU Time} = \text{Instruction Count} \times \text{CPI} \times \text{Clock period}$$

- The time taken by a program to execute is the product of
  - Number of machine instructions executed
  - Number of clock cycles per instruction (**CPI**)
  - Single clock period duration
- **Example**: 10,000 instructions, CPI=2, clock period = 250 ps

$$\text{CPU Time} = 10{,}000\,instructions \times 2 \times 250\,ps$$
$$= 10^4 \times 2 \times 250.10^{-12} = 5.10^{-6}\,\text{sec}.$$

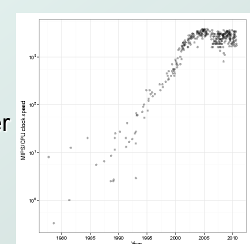
Processor Execution time

$$\text{CPU Time} = \text{Instruction Count} \times \text{CPI} \times \text{Clock Cycle Time}$$

- Instruction Count for a program
  - Determined by program, ISA and compiler
- Average Cycles per instruction (CPI)
  - Determined by CPU hardware
  - If different instructions have different CPI
    - Average CPI affected by instruction mix
- Clock cycle time (inverse of frequency)
  - Logic levels
  - technology

Reducing clock cycle time

- Has worked well for decades.
- Small transistor dimensions implied smaller delays and hence lower clock cycle time.
- Not any more.



CPI (cycles per instruction)

- What is LC-3 cycles per instruction?
- Instructions take 5-9 cycles (p. 568), assuming memory access time is one clock period.
  - LC-3 CPI may be about 6*. (ideal)

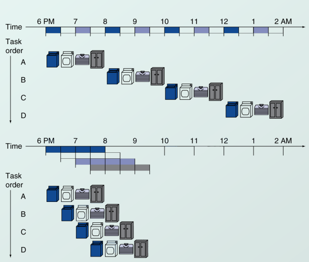
    Load/store instructions are about 20-30%
- No cache, memory access time = 100 cycles?
  - LC-3 CPI would be very high.
- Cache reduces access time to 2 cycles.
  - LC-3 CPI higher than 6, but still reasonable.

Parallelism to save time

- Do things in parallel to save time.
- Example: Pipelining
  - Divide flow into stages.
  - Let instructions flow into the pipeline.
  - At a time multiple instructions are under execution.

Pipelining Analogy

- Pipelined laundry: overlapping execution
  - Parallelism improves performance



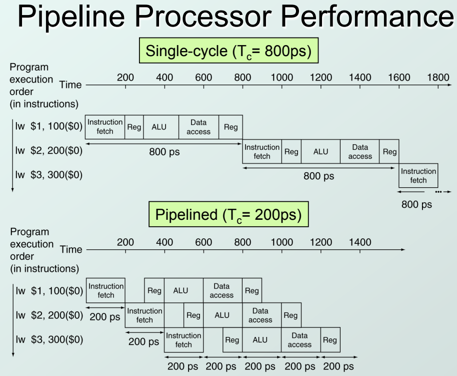
- Four loads:
  - time

    = 4x2 = 8 hours
- Pipelined:
  - Time in example

    = 7x0.5 = 3.5 hours
  - Non-stop

    = 4x0.5 = 2 hours.

Pipeline Processor Performance



Pipelining: Issues

- Cannot predict which branch will be taken.
  - Actually you may be able to make a good guess.
  - Some performance penalty for bad guesses.
- Instructions may depend on results of previous instructions.
  - There may be a way to get around that problem in some cases.

Instruction level parallelism (ILP):

- **Pipelining** is one example.
- **Multiple issue**: have multiple copies of resources
  - Multiple instructions start at the same time
  - Need careful scheduling
    - Compiler assisted scheduling
    - Hardware assisted (“**superscaler**”): “dynamic scheduling”
      - Ex: AMD Opteron x4
      - CPI can be less than 1!.

Flynn’s taxonomy

- Michael J. Flynn, 1966

		Data Streams	
		Single	Multiple
Instruction Streams	Single	**SISD**: Intel Pentium 4	**SIMD**: SSE instructions of x86
	Multiple	**MISD**: No examples today	**MIMD**: Intel Xeon e5345

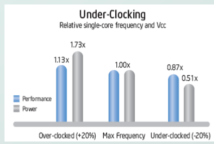
- Instruction level parallelism is still SISD
- SSE (Streaming SIMD Extensions): vector operations
- Intel Xeon e5345: 4 cores

Multi what?

- Multitasking: tasks share a processor
- Multithreading: threads share a processor
- Multiprocessors: using multiple processors
  - For example multi-core processors (multiples processors on the same chip)
  - Scheduling of tasks/subtasks needed
- Thread level parallelism:
  - multiple threads on one/more processors
- Simultaneous multi-threading:
  - multiple threads in parallel (using multiple *states*)

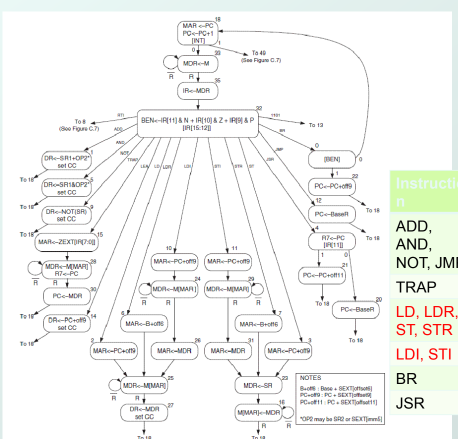
Multi-core processors

- Power consumption has become a limiting factor
- Key advantage: lower power consumption for the same performance
  - Ex: 20% lower clock frequency: 87% performance, 51% power.
- A processor can switch to lower frequency to reduce power.
- N cores: can run n or more threads.



Multi-core processors

- Cores may be identical or specialized
- Higher level caches are shared.
- Lower level *cache coherency* required.
- Cores may use superscalar or simultaneous multi-threading architectures.



LC-3 states

Instruction	Cycles
ADD, AND, NOT, JMP	5
TRAP	8
LD, LDR, ST, STR	7
LDI, STI	9
BR	5, 6
JSR	6