

Parallelism via Multithreaded and Multicore CPUs

Bradley Dutton

March 29, 2010

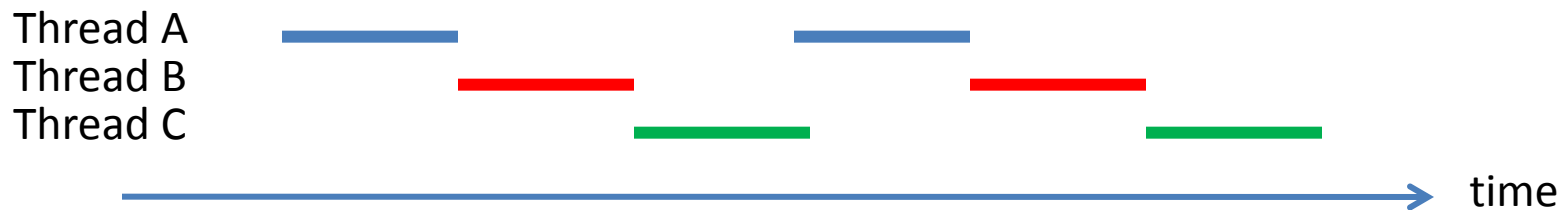
ELEC 6200

Outline

- Multithreading
 - Hardware vs. Software definition
- Hardware multithreading
 - Simple multithreading
 - Interleaved multithreading
 - Simultaneous multithreading
- Multicore CPUs (a.k.a. multiprocessing)
- Advantage and Disadvantages
- Conclusions

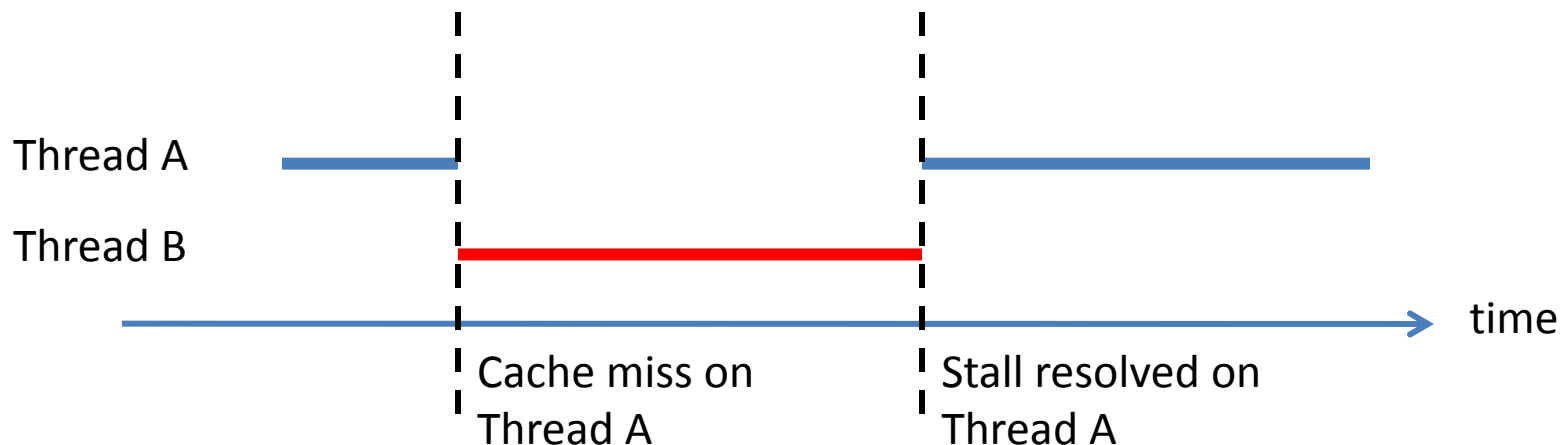
Multithreading (software perspective)

- On a single processor, multithreading occurs by time division multiplexing
 - The processor switches between different threads
 - Commonly called a context switch
 - Operating systems (OS) use context switching to give the illusion of multiple processes executing at once



Simple Multithreading CPUs

- Simplest form of multithreading occurs when a thread runs until it is blocked by an event that causes the pipeline to stall
 - e.g. cache miss could takes 100s of clock cycles
 - Switching to a ready-to-go thread can cover the latency, increasing throughput



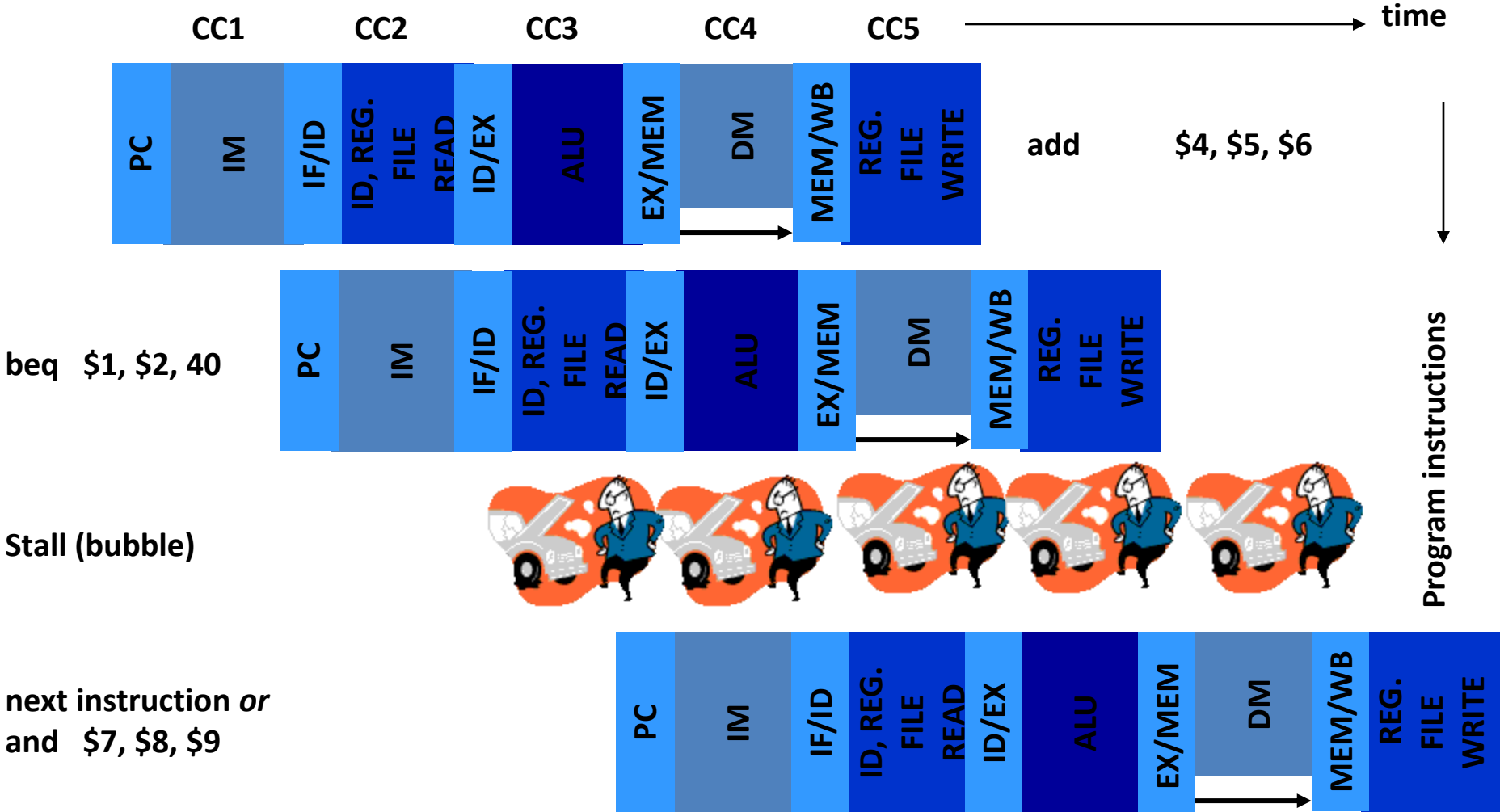
Simple Multithreading Cost

- Multithreading hardware allows fast switching between a stalled thread and a ready-to-go thread
 - Cost is duplication of all state registers (PC, general purpose registers, control registers)
 - Combinational logic (ALU, etc...) is shared
 - For a processor which executes two threads, two copies of all CPU registers are present
 - It appears to each thread that it is executing alone
 - Context switch can be done in 1 clock cycle
- Thread scheduler chooses from among list of ready-to-go threads and maintains list of stalled threads

Interleaved Multithreading

- Interleaved multithreading (a.k.a. pre-emptive, fine-grained or time-sliced)
 - Cycle i : issue an instruction from Thread A
 - Cycle $i+1$: issue an instruction from Thread B
- Idea is to remove all control hazards
 - Recall that branch requires a stall or flush
 - With IMT, the next instruction in the pipeline is always the correct instruction for the other thread
- Additional cost: each pipeline stage must track the thread ID of instruction it is processing
 - Which copy of the register file does my data belong to?

Stall on Branch



Source: V.D. Agrawal, "Pipelining (Chapter 4)," ELEC 5200/6200 Spring 2010

Simultaneous Multithreading (SMT)

- Applies only to superscalar processors
 - e.g. Intel HyperThreading, STI Cell (PlayStation 3), IBM Power5, Sun UltraSPARC T2
- Normal superscalar issues multiple instructions from a single thread per cycle to multiple copies of hardware
 - With SMT, multiple instruction are issued from multiple threads every cycle

Aside: Superscalar CPUs

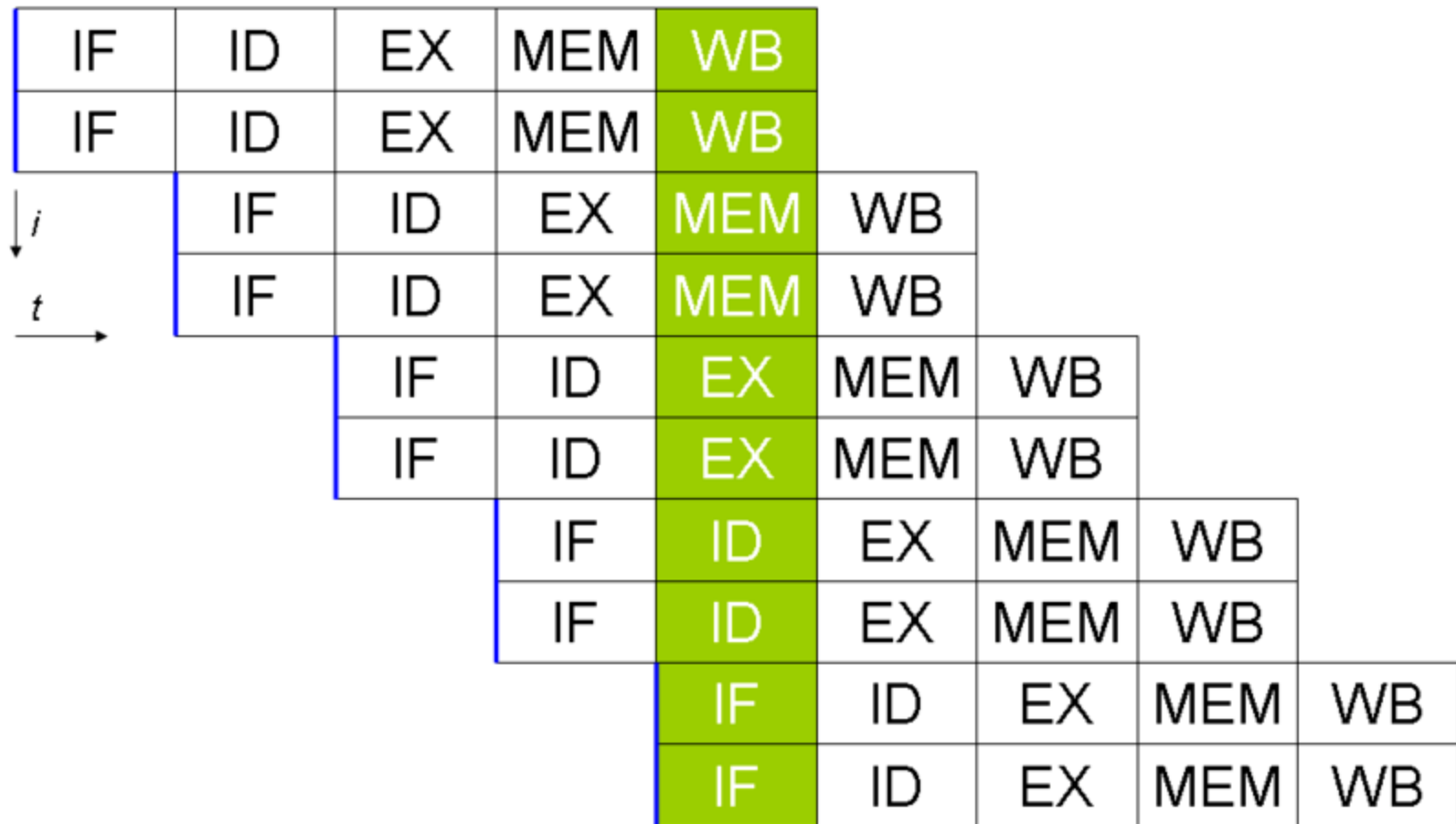
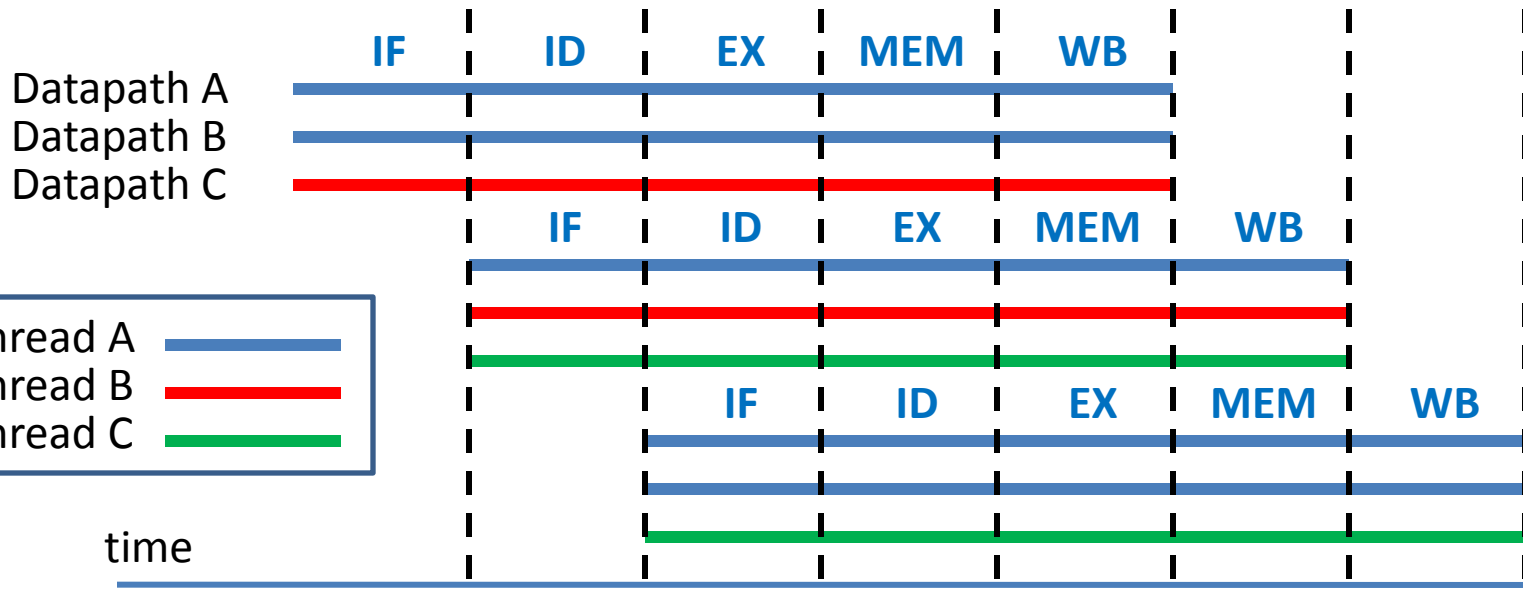


Image source: <http://en.wikipedia.org/wiki/File:Superscalarpipeline.png>

Simultaneous Multithreading (SMT)

- Example of SMT
 - Cycle i : instructions j and $j+1$ from thread A; instruction k from thread B all issued
 - Cycle $i+1$: instruction $j+2$ from thread A; instruction $k+1$ from thread B; instruction m from thread C all issued
 - Cycle $i+2$: instruction $j+3$ from thread A; instructions $m+1$ and $m+2$ from thread C all issued



Multithreading Conclusion

- Hardware cost can be small to add a thread
 - Only 5% area overhead for Intel Xeon to support a second hardware thread with 33% greater throughput
 - Area grows approximately linearly for up to 8 threads
 - Returns are diminishing for increasing number of threads on-chip due to data contentions
 - All of the complexity of the original processor is kept
 - In commercial processors, # on-chip threads vary
 - 2 (e.g. Intel Xeon)
 - 8 (e.g. Sun UltraSPARC T2)

Multicore CPUs

- A simple multicore design pairs two or more single-core chip designs
 - Only the control and execution units are replicated
 - Cache, memory controller, and secondary processing units (such as FPUs) are shared
- Each core executes one or more threads (i.e. each core may also be multithreaded)



Modern Multicore CPUs

- Aside from hardware parallelism, caches are the most important feature of modern CPUs to enhance performance
 - But software threads which do not share much data will compete for the cache (lots of stalls)
- Solution: Private cache
 - Modern multicore designs employ a hierarchical cache design, where L1 is private and L2 can be shared or private
 - Multiple threads per-core share L1 cache
- Additional multithreading per-core can cover some of this latency

Advantages/Disadvantages

- Power and performance concerns have cause interests to shift towards designs with many simpler cores
 - Power is reduced linearly with reduced complexity
 - Cores may be individually power-tuned
 - Some cores may be turned off
 - Or core clocks frequencies may be scaled individually
- Large serial applications may benefit from additional complexity and speed of single-core designs

Conclusions

- Multithreaded and Multicore CPUs can significantly improve processor throughput
 - Multicore performance is primarily limited by memory bandwidth
 - More cores have to be kept busy with more data
 - Adding multithreading to each core can improve performance over multicore single-thread
- Conclusion: no clear winner
 - A combination of multicore and multithreaded provides the best power/performance characteristics for most general computing applications

References

- Angela C. Sodan, Jacob Machina, Arash Deshmeh, Kevin Macnaughton, Bryan Esbaugh, "Parallelism via Multithreaded and Multicore CPUs," *Computer*, pp. 24-32, March, 2010