CSE 548 Computer Architecture

Clock Rate *vs* IPC

V. Agarwal, M. S. Hrishikesh, S. W. Kechler. D. Burger

Presented by: Ning Chen

Transistor Changes

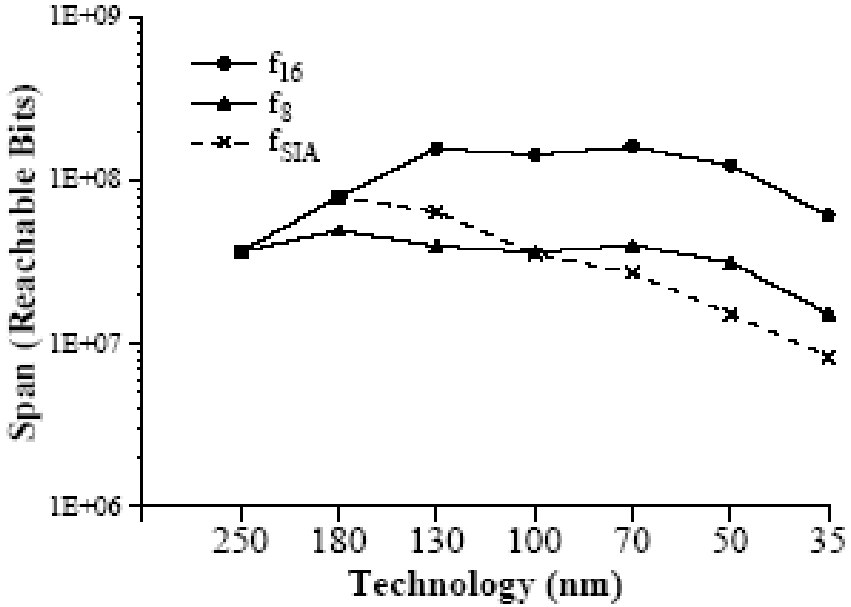
- Development of silicon fabrication technology caused transistor sizes to decrease.
- Benefits:
  - provide area for more complex microarchitectures
  - reduce transistor switching time
- Impact:
  - result in larger wire resistance
  - however, wire capacitance has not increased proportionally

Delay of a Wire

Gate Length (nm)	Dielectric Constant κ	Metal ρ ($\mu\Omega$-cm)	Mid-Level Metal				Top-Level Metal			
			Width (nm)	Aspect Ratio	R_{wire} ($m\Omega/\mu m$)	C_{wire} ($fF/\mu m$)	Width (nm)	Aspect Ratio	R_{wire} ($m\Omega/\mu m$)	C_{wire} ($fF/\mu m$)
250	3.9	3.3	500	1.4	107	0.215	700	2.0	34	0.265
180	2.7	2.2	320	2.0	107	0.253	530	2.2	36	0.270
130	2.7	2.2	230	2.2	188	0.263	380	2.5	61	0.285
100	1.6	2.2	170	2.4	316	0.273	280	2.7	103	0.296
70	1.5	1.8	120	2.5	500	0.278	200	2.8	164	0.296
50	1.5	1.8	80	2.7	1020	0.294	140	2.9	321	0.301
35	1.5	1.8	60	2.9	1760	0.300	90	3.0	714	0.317

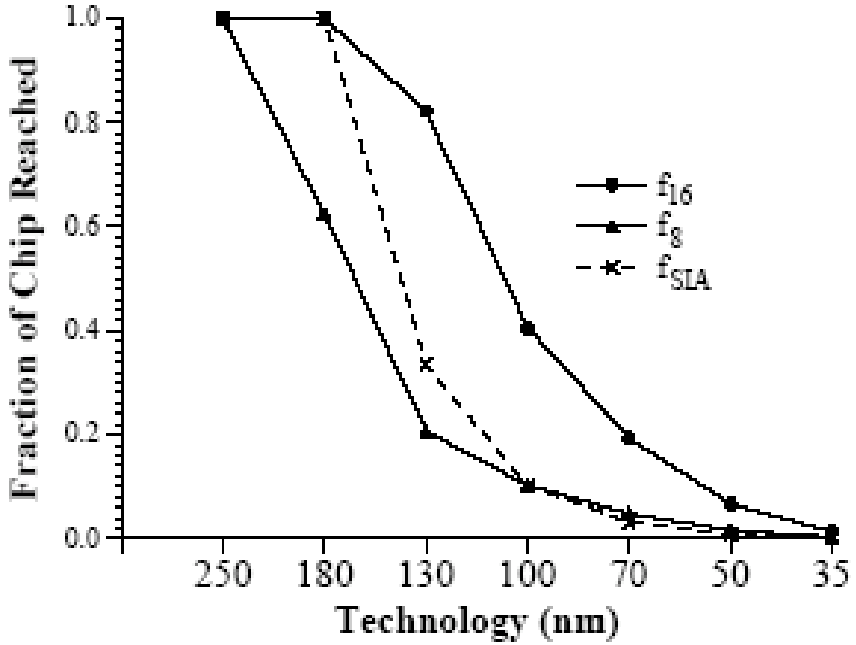
Clock Scaling

- With increasing clock rates,
  - ✓ the distance that a signal can travel in a single clock cycle decreases.
  - ✓ The absolute # of bits that can be reached in a single clock cycle increases



Clock Scaling

- With increasing clock rates, fraction of the total chip area that can be reached in a single clock cycle increases.



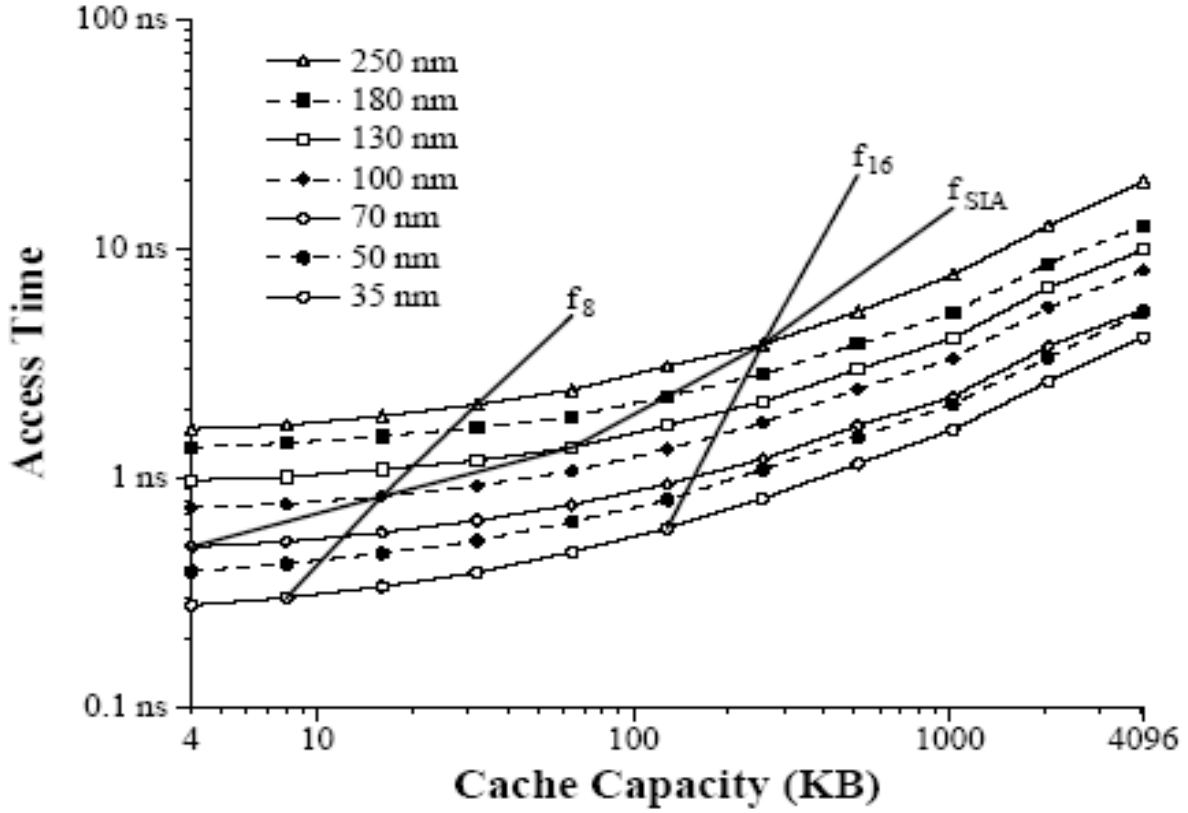
Conclusion (1)

- Due to increasing clock frequencies, wire delays are increasing at a high rate.
- Chip performance will no longer be determined solely by the # of transistors, but will depend on the amount of state and logic that can be reached in a sufficiently small # of clock cycles.
- With future wire delays, structure size will be limited and the time to bypass results between pipeline stages will grow.

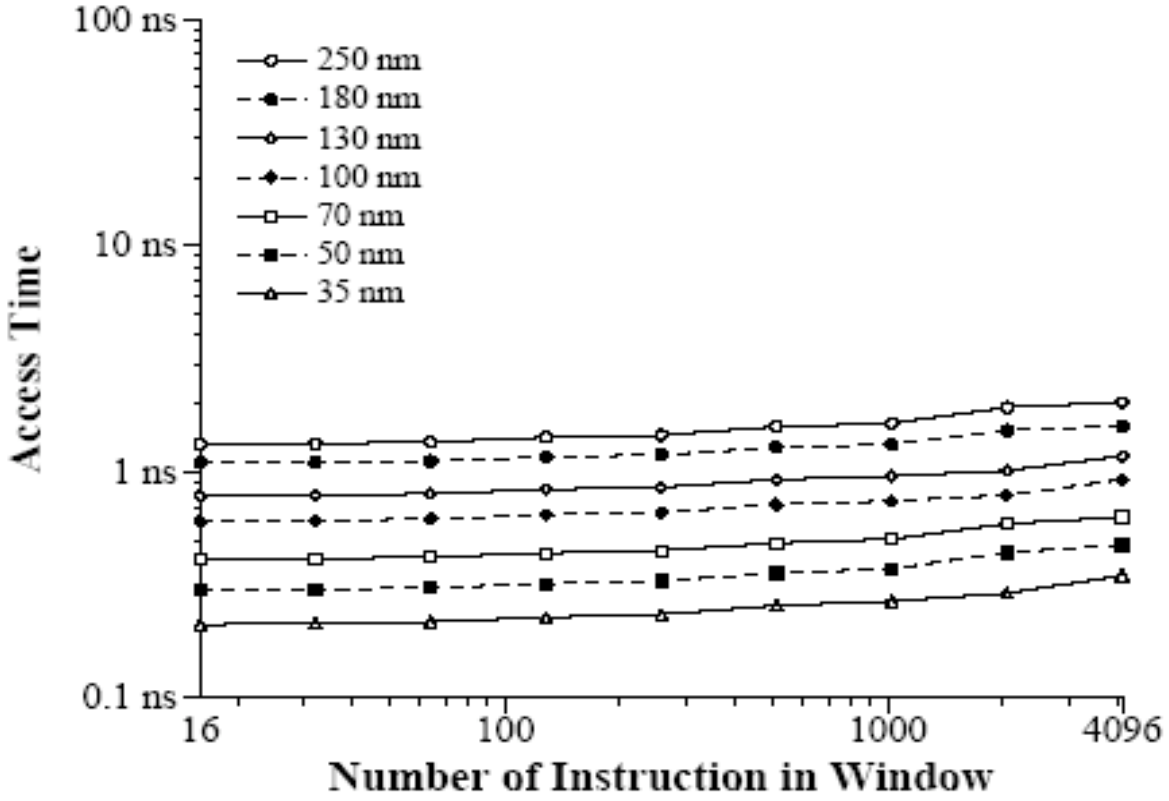
Access Time

- Factors affect memory structure access time
  - cache capacity
  - block size
  - associativity
  - number of ports
  - process technology

Access Time and Capacity



Access Time and Instruction Window



Conclusion (2)

- To access caches, register files, branch prediction tables, and instruction windows in a single cycle will require the capacity of these structures to decrease as clock rates increase.
- The # of cycles needed to access the structures

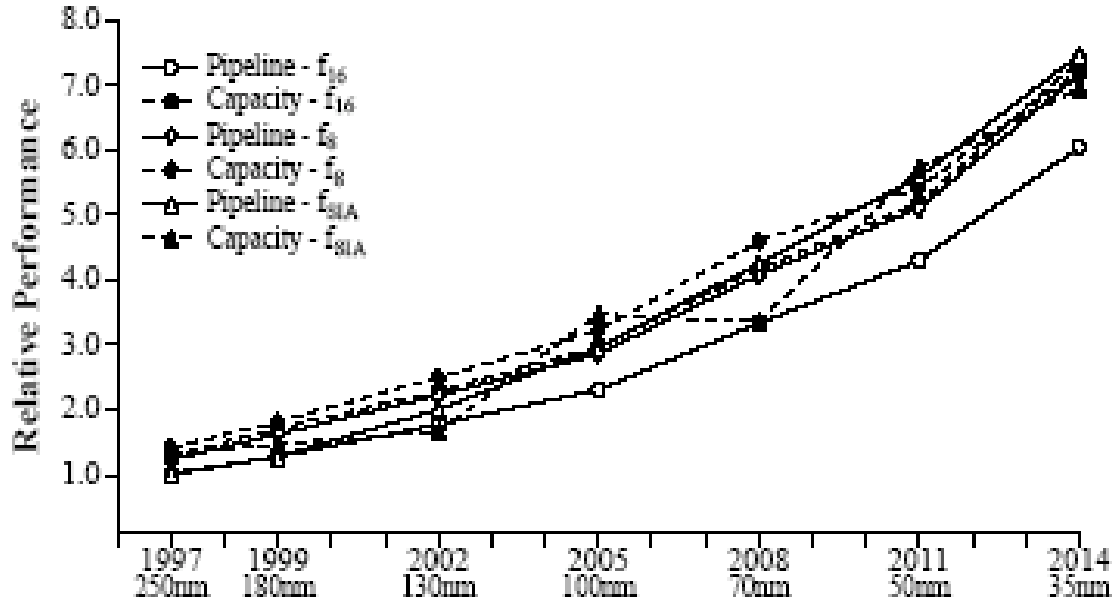
Structure Name	f_{SIA}	f_8	f_{16}
L1 cache 64K (2 ports)	7	5	3
Integer register file 64 entry (10 ports)	3	2	1
Integer issue window 20 entry (8 ports)	3	2	1
Reorder buffer 64 entry (8 ports)	3	2	1

Performance Analysis

- Approaches
  - *Capacity scaling*: shrink the microarchitectural structures sufficiently so that their access penalties are constant across technologies, where *access penalty* is the access time for a structure measured in clock cycles.
  - *Pipeline scaling*: hold the capacity of a structure constant and increase the pipeline depth as necessary to cover the increased latency across technologies.

Capacity Scaling *vs* Pipeline Scaling

- Performance increases for different scaling strategies.

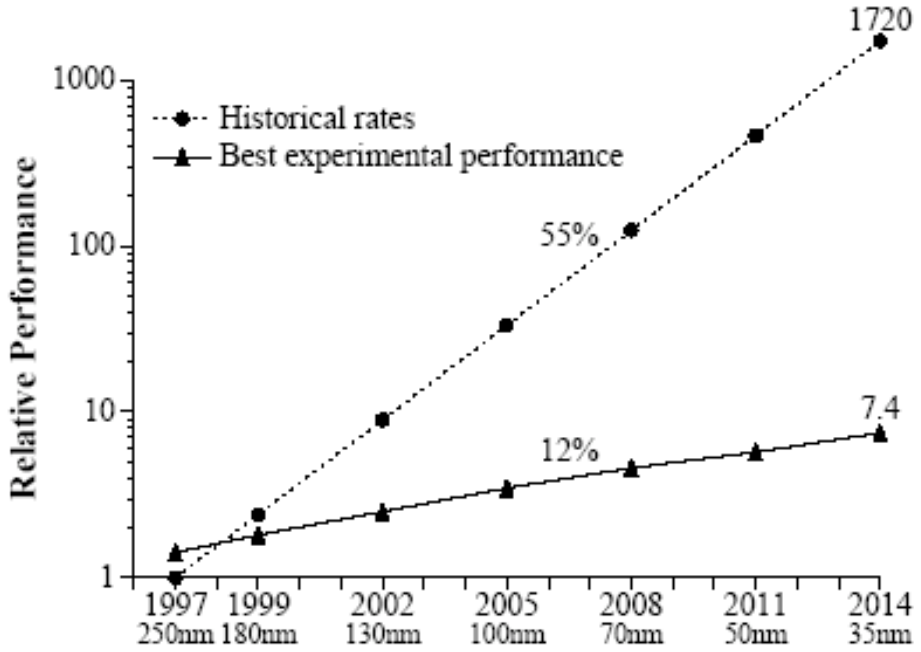


Conclusion (3)

- The overall performance for both clocks at both scaling methodologies is nearly identical.
- The maximal performance increase is a factor of 7.4, which corresponds to a 12.5% annual improvement over that 17-year span.

Conclusion (4)

- No scaling strategy permits annual performance improvements of better than 12.5%, which is far worse than the annual 50-60% to which we have grown accustomed.



CSE 548

Optimizing Pipelines for Power and Performance

Presented by: Anna Cavender

Terminology

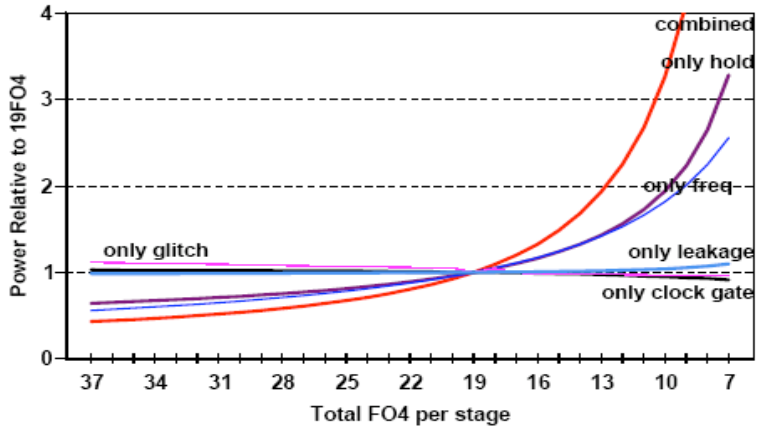
- Latch = a logic circuit.
  - Input: data and clock
  - Output: data
  - Transfers data to the output when signaled from the clock.
- Clock gating = logic units that aren't being used, don't need to receive signals from the clock. Power can be saved by only sending a clock signal when necessary.

Analytical Model – an English Translation

- Basic idea: Model the throughput of the machine in terms of the pipeline stages.
- Related to the number of stalls that occur in the pipelines:
  - Stalls due to data dependence:
    - Split the load store pipe into two: one for cache hits the other for cache misses.
  - Stalls due to instruction fetch delay:
    - Modeled this given pipeline utilization and total time per stage of the pipe.

Performance and Power Methodology

- Two types of power:
  - Dynamic Power
    - Hold Power = power when no switching is occurring
    - Switching Power = logic and data.
  - Leakage Power



- Increase pipeline depth = increased power usage

Results from Simulator

SPEC2000 = standard benchmarks. Optimal pipeline depth changes based on what metric you choose.

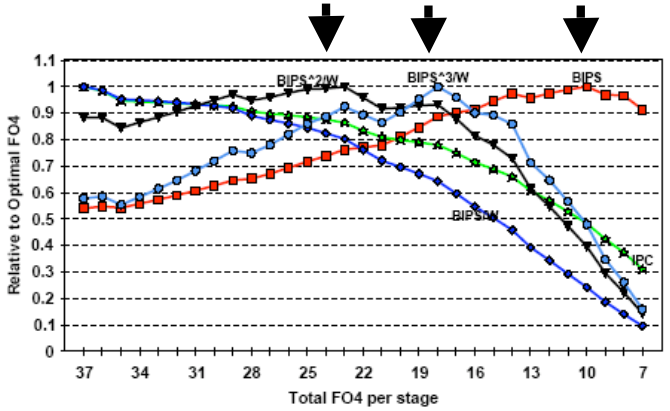


Figure 7. Simulation Results for SPEC2000

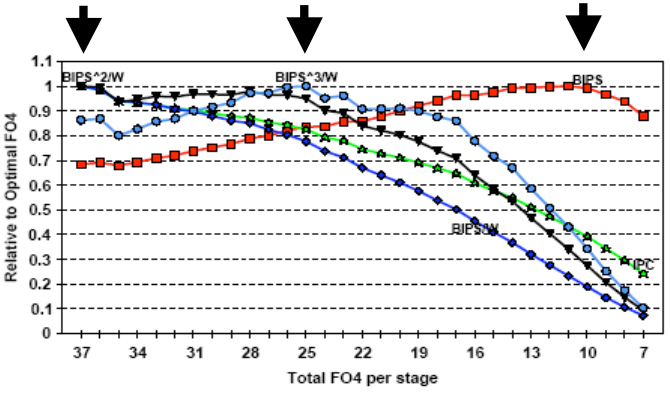


Figure 8. Simulation Results for TPC-C

TPC-C = transaction processing benchmarks. Optimal pipeline depth shifts because BIPS decrease more dramatically, so ratio peaks sooner.

Sensitivity Analysis

- Finding the optimal pipeline depth is sensitive to many parameters:
  - Latch Growth Factor
    - Number of latches increases with Pipeline depth.

      Latches added to break up logic into more stages.

      Determined by logic shape.
  - **Favors shallower pipeline.**

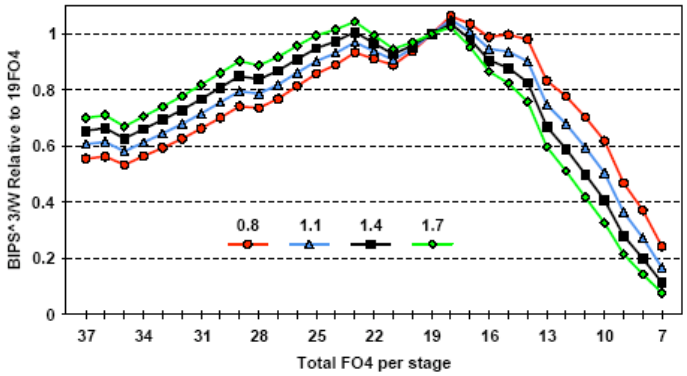


Figure 11. BIPS^3/W varying $LatchGrowthFactor$

Graph shows 4 different growth factors going from easier to harder to pull apart. They all result in same estimate for optimal pipeline depth.
But, it shows that there is a range where we can increase performance without loosing too much in the power world.

Sensitivity Analysis

- Latch Power Ratio
  - Ratio of hold power to total power.
  - **Favors deeper pipeline**
- Latch Insertion Delay
  - More latches needed for more pipe stages
  - **Favors shallower pipeline for lower-power latches**
- Glitch Factor
  - Difference in delay from latch output to gate.
  - Linearly dependent on the logic depth.
  - **Favors deeper pipeline**
- Leakage Factor
  - **Favors deeper pipeline**

Conclusion

- Modeled and simulated power-performance trade-offs.
- Optimal size of pipeline stages is around 18 FO4 with a little wiggle room to achieve better performance with small sacrifice in power usage.
- This optimal is shallower than if performance was our only concern.

Questions

- What technology is being developed to make sure we keep getting really good performance? (well, WaveScalar, and what else?)
- More local communication and optimized layout (e.g. circular with shared units in the middle) could help. Aren't there tools for optimization?
- The clock is one cause of all this; any research of new asynchronous cores (in part, or completely)?