The End of Conventional Microprocessors

Edwin Olson

9/21/2000

Historical Growth

- Microprocessor speed increasing at a roughly 50-60% annual rate.
  - Moore’s law predicts about 58%
- Improving manufacturing processes responsible
  - Transistors switch faster
  - Increasing transistor budget enables more sophisticated architectures

Two Ways to Achieve Performance

- Braniacs: High IPC, lower clock-rate (higher FO4 delay) processors like PA-RISC
- Speed Demons: Low IPC, high clock-rate (lower FO4 delay) processors like Alpha.
- Today's designs have benefited from both approaches, which exemplifies the headroom available today in both strategies.

Today's uPs

- Today's uPs are monolithic cores which assume that signals can reach entire chip in one clock. They are *capacity* bound.
- In 0.18um, signals may not be able to travel from one corner to another in 1 cycle. uPs begin to become *communication* bound.
- WHY?

Transistor Scaling

- Good News! Switching delay of transistor proportional to λ. τ => ατ
- FO4 delay empirically estimated by
  - – 360*2λ ps (2λ is minimum gate length)
    - 0.250 : 90ps
    - 0.035nm: 12.6ps
- This is a 7.1x speed improvement.

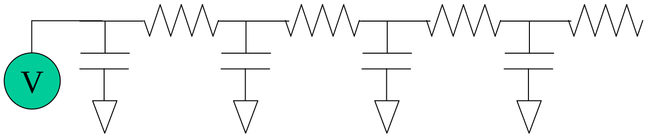
Wire Delay

- Model a wire as a distributed RC network
- Many RC delays in parallel

$$\tau = \int_0^L C_w R_w x dx = \frac{1}{2} C_w R_w L^2$$

C_w: Capacitance per unit length

R_w: Resistance per unit length



Wire Scaling

- Assume we scale an existing design down, shrinking all dimensions by α.
- $C_w = k\varepsilon_0 W/d$ (W is width of wire)
- When scaled by α ($\alpha<1$), $\tau = \int_0^L C_w R_w x\,dx = \frac{1}{2} C_w R_w L^2$
  - $W \Rightarrow W\alpha$
  - $d \Rightarrow d\alpha$
  - C_w stays the same!
  - $R \Rightarrow R/\alpha^2$ (assuming fixed aspect ratio)
    - Not quite this bad if we can increase aspect ratio some
  - $L \Rightarrow L\alpha$
  - **$\tau \Rightarrow \tau$**
- A wire is the same speed as before.

Wire Scaling

- Suppose we make our design more complex (to increase IPC). Now, L doesn't scale.
- Now, $\tau \rightarrow \frac{1}{\alpha^2}\tau$

This does not account for increasing aspect ratios and falling resistivities.

Side note

- We can design a wire with delay proportional to just L, not L^2 by using repeaters.
- Given a process-determined repeater-length, l_0, we can span a distance of L by having repeater segments joined together. Each repeater segment has a delay proportional to l_0^2/α^2.

Repeaters

$$\frac{L}{l_0}\left\{\int_0^{l_0} C_w(xR_w+R_r)dx + \rho + C_r\left(R_w l_0 + R_r\right)\right\}$$

$$= \frac{L}{l_0}\left\{\frac{1}{2}C_w R_w l_0^2 + C_w R_r l_0 + \rho + C_r(R_w l_0 + R_r)\right\}$$

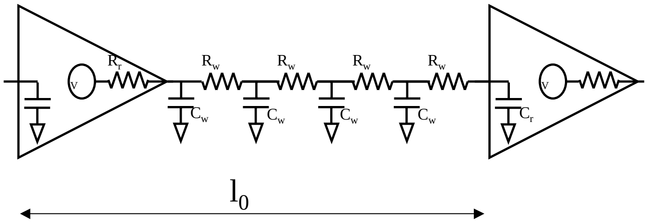
C_r=Cap. of Repeater

R_r=Res. Of Repeater

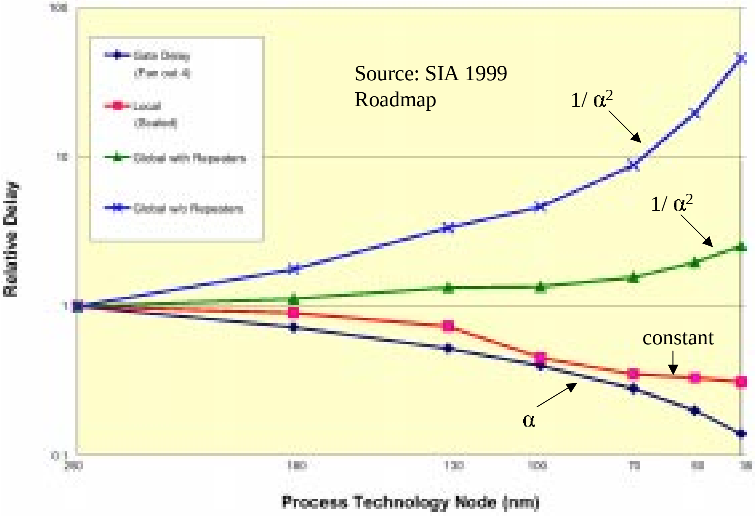
C_w=Cap/length of wire

R_w=Res/length of wire

ρ=intrinsic delay of repeater



Gates vs. Wires

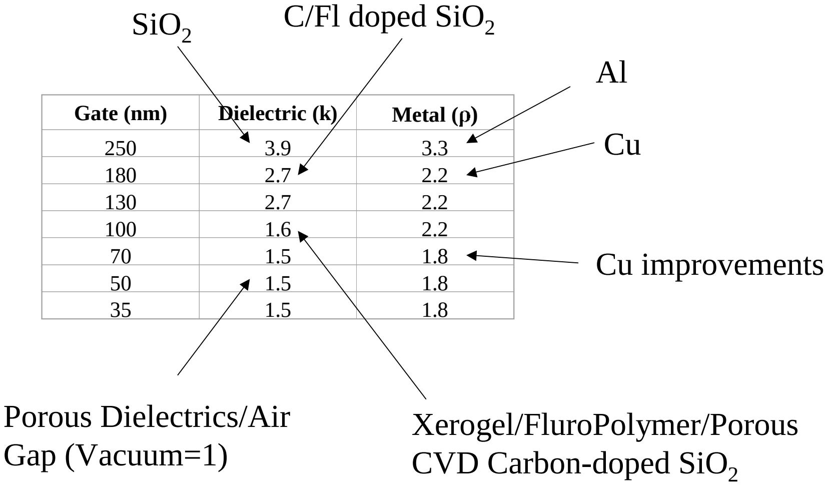


Source: SIA 1999
Roadmap

So what’s the problem?

- Transistors are getting faster
- Local wiring is staying the same speed
- Global wiring is getting really slow
- Smaller feature size only improves transistor speed. Even if the wires were infinitely fast, projected process improvements (250nm to 35nm) would yield only a 7.2x improvement through 2014 (15% annualized growth).
- We *need* global wiring to access caches and other large structures!

Material Science to the Rescue



Gate (nm)	Dielectric (k)	Metal (ρ)
250	3.9	3.3
180	2.7	2.2
130	2.7	2.2
100	1.6	2.2
70	1.5	1.8
50	1.5	1.8
35	1.5	1.8

SiO_2

C/Fl doped SiO_2

Al

Cu

Cu improvements

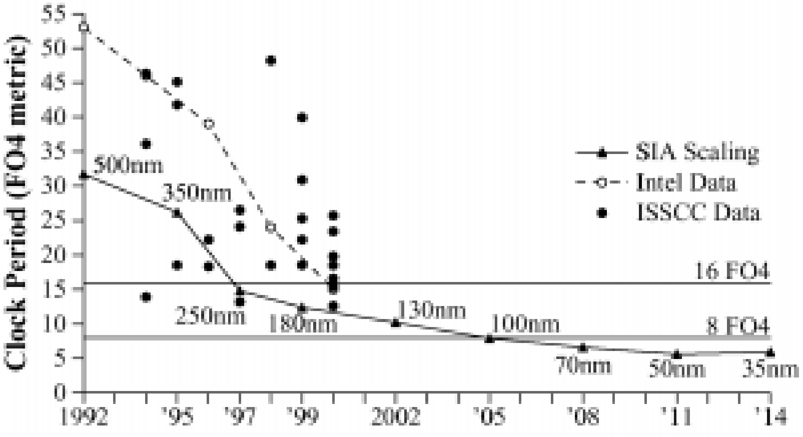
Porous Dielectrics/Air Gap (Vacuum=1)

Xerogel/FluroPolymer/Porous CVD Carbon-doped SiO_2

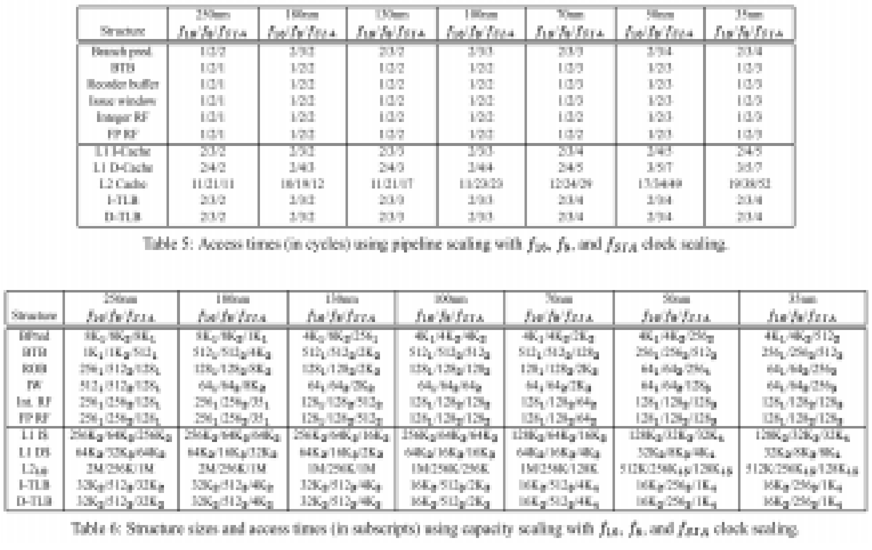
Approaches to Scaling uP designs

- We can’t increase IPC *and* clock rate.
  - – IPC increased by bigger structures, which are getting slower, not faster.
- *Capacity Scaling*: shrink structures so that they have roughly constant access penalties
- *Pipeline Scaling*: fix structure size, and increase pipeline depth to account for growing latency.

FO4 delays



Capacity and Pipeline Scaling



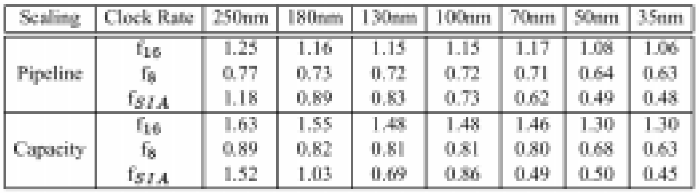
Structure	250nm	180nm	130nm	100nm	70nm	50nm	35nm
	$f_{16}/f_8/f_{SIA}$	$f_{16}/f_8/f_{SIA}$	$f_{16}/f_8/f_{SIA}$	$f_{16}/f_8/f_{SIA}$	$f_{16}/f_8/f_{SIA}$	$f_{16}/f_8/f_{SIA}$	$f_{16}/f_8/f_{SIA}$
Branch pred.	1/2/2	2/3/2	2/3/2	2/3/3	2/3/3	2/3/4	2/3/4
BTB	1/2/1	1/2/2	1/2/2	1/2/2	1/2/3	1/2/3	1/2/3
Reorder buffer	1/2/1	1/2/2	1/2/2	1/2/2	1/2/3	1/2/3	1/2/3
Issue window	1/2/1	1/2/2	1/2/2	1/2/2	1/2/3	1/2/3	1/2/3
Integer RF	1/2/1	1/2/2	1/2/2	1/2/2	1/2/2	1/2/3	1/2/3
FP RF	1/2/1	1/2/2	1/2/2	1/2/2	1/2/2	1/2/3	1/2/3
L1 I-Cache	2/3/2	2/3/2	2/3/3	2/3/3	2/3/4	2/4/5	2/4/5
L1 D-Cache	2/4/2	2/4/3	2/4/3	2/4/4	2/4/5	3/5/7	3/5/7
L2 Cache	11/21/11	10/19/12	11/21/17	11/23/23	12/24/29	17/34/49	19/38/52
I-TLB	2/3/2	2/3/2	2/3/3	2/3/3	2/3/4	2/3/4	2/3/4
D-TLB	2/3/2	2/3/2	2/3/3	2/3/3	2/3/4	2/3/4	2/3/4

Table 5: Access times (in cycles) using pipeline scaling with f_{16}, f_8, and f_{SIA} clock scaling.

Structure	250nm	180nm	130nm	100nm	70nm	50nm	35nm
	$f_{16}/f_8/f_{SIA}$	$f_{16}/f_8/f_{SIA}$	$f_{16}/f_8/f_{SIA}$	$f_{16}/f_8/f_{SIA}$	$f_{16}/f_8/f_{SIA}$	$f_{16}/f_8/f_{SIA}$	$f_{16}/f_8/f_{SIA}$
BPred	$8K_1/8K_2/8K_1$	$8K_1/8K_2/1K_1$	$4K_1/8K_2/256_1$	$4K_1/4K_2/4K_2$	$4K_1/4K_2/2K_2$	$4K_1/4K_2/256_2$	$4K_1/4K_2/512_2$
BTB	$1K_1/1K_2/512_1$	$512_1/512_2/4K_2$	$512_1/512_2/2K_2$	$512_1/512_2/512_2$	$512_1/512_2/128_2$	$256_1/256_2/512_2$	$256_1/256_2/512_2$
ROB	$256_1/512_2/128_1$	$128_1/128_2/8K_2$	$128_1/128_2/2K_2$	$128_1/128_2/128_2$	$128_1/128_2/2K_2$	$64_1/64_2/256_1$	$64_1/64_2/256_2$
IW	$512_1/512_2/128_1$	$64_1/64_2/8K_2$	$64_1/64_2/2K_2$	$64_1/64_2/64_2$	$64_1/64_2/2K_2$	$64_1/64_2/128_2$	$64_1/64_2/256_2$
Int. RF	$256_1/256_2/128_1$	$256_1/256_2/35_1$	$128_1/128_2/512_2$	$128_1/128_2/128_2$	$128_1/128_2/64_2$	$128_1/128_2/128_2$	$128_1/128_2/128_2$
FP RF	$256_1/256_2/128_1$	$256_1/256_2/35_1$	$128_1/128_2/512_2$	$128_1/128_2/128_2$	$128_1/128_2/64_2$	$128_1/128_2/128_2$	$128_1/128_2/128_2$
L1 IS	$256K_2/64K_2/256K_2$	$256K_2/64K_2/64K_2$	$256K_2/64K_2/16K_2$	$256K_2/64K_2/64K_2$	$128K_2/64K_2/16K_2$	$128K_2/32K_2/32K_4$	$128K_2/32K_2/32K_4$
L1 DS	$64K_2/32K_2/64K_2$	$64K_2/16K_2/32K_2$	$64K_2/16K_2/2K_2$	$64K_2/16K_2/16K_2$	$64K_2/16K_2/4K_2$	$32K_4/8K_4/4K_4$	$32K_4/8K_4/8K_4$
$L2_{SS}$	2M/256K/1M	2M/256K/1M	1M/256K/1M	1M/256K/256K	1M/256K/128K	$512K/256K_{18}/128K_{18}$	$512K/256K_{18}/128K_{18}$
I-TLB	$32K_2/512_2/32K_2$	$32K_2/512_2/4K_2$	$32K_2/512_2/4K_2$	$16K_2/512_2/2K_2$	$16K_2/512_2/4K_3$	$16K_2/256_2/1K_3$	$16K_2/256_2/1K_3$
D-TLB	$32K_2/512_2/32K_2$	$32K_2/512_2/4K_2$	$32K_2/512_2/4K_2$	$16K_2/512_2/2K_2$	$16K_2/512_2/4K_3$	$16K_2/256_2/1K_3$	$16K_2/256_2/1K_3$

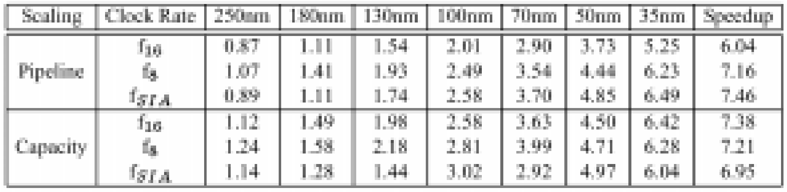
Table 6: Structure sizes and access times (in subscripts) using capacity scaling with f_{16}, f_8, and f_{SIA} clock scaling.

Capacity and Pipeline Scaling-- Performance



Scaling	Clock Rate	250nm	180nm	130nm	100nm	70nm	50nm	35nm
Pipeline	f_{16}	1.25	1.16	1.15	1.15	1.17	1.08	1.06
	f_8	0.77	0.73	0.72	0.72	0.71	0.64	0.63
	f_{SIA}	1.18	0.89	0.83	0.73	0.62	0.49	0.48
Capacity	f_{16}	1.63	1.55	1.48	1.48	1.46	1.30	1.30
	f_8	0.89	0.82	0.81	0.81	0.80	0.68	0.63
	f_{SIA}	1.52	1.03	0.69	0.86	0.49	0.50	0.45

Table 7: Geometric mean of IPC for each technology across the SPEC95 benchmarks.

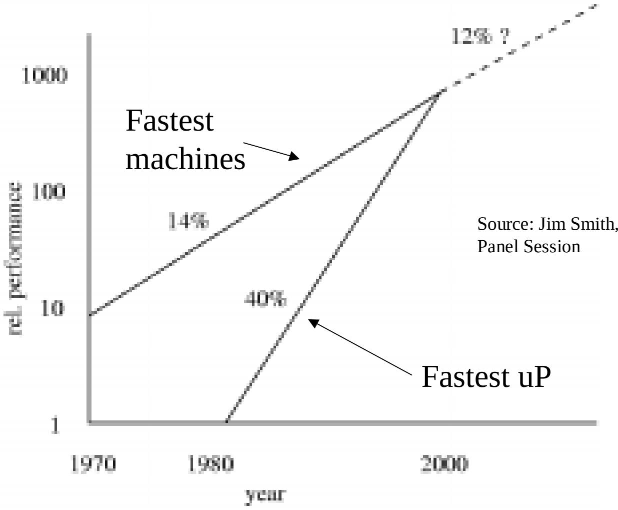


Scaling	Clock Rate	250nm	180nm	130nm	100nm	70nm	50nm	35nm	Speedup
Pipeline	f_{16}	0.87	1.11	1.54	2.01	2.90	3.73	5.25	6.04
	f_8	1.07	1.41	1.93	2.49	3.54	4.44	6.23	7.16
	f_{SIA}	0.89	1.11	1.74	2.58	3.70	4.85	6.49	7.46
Capacity	f_{16}	1.12	1.49	1.98	2.58	3.63	4.50	6.42	7.38
	f_8	1.24	1.58	2.18	2.81	3.99	4.71	6.28	7.21
	f_{SIA}	1.14	1.28	1.44	3.02	2.92	4.97	6.04	6.95

Agarwal’s Results

- Maximum speedup of 7.4 (annual gain of 12.5%)
- BUT the model they used has
  - large branch-taken penalties
  - does not use any clustering
  - Does not account for advances in compilers, microarchitecture (e.g., VLIW)

Have we really *just now* hit the wall?



Fastest machines

Fastest uP

Source: Jim Smith, ISCA 2000 Panel Session