High Performance ULSI –
The Real Limiter To Interconnect Scaling

Ron Ho
Sun Labs
ron.ho@sun.com

The recent past (1995-2005)

“It all started” with a classic paper by Intel Fellow Mark Bohr

- *Interconnect scaling–the real limiter to high performance ULSI*
- 1995 International Electron Devices Meeting
- Called for new materials and new circuits

Circuits and process people have been very busy since

- Lots of new techniques and methodologies; some even used
- On-chip wires are well understood, arguably manageable

Let’s look ahead to the next several years

- Trends for high performance: architecture, circuits, power
- How do they impact how we think about wires?

Outline

First, review basic wire scaling trends

- Pay attention to hot-button topics like noise and inductance

Consider high-performance CPU trends

- Focus on frequency, power and cost

Look at where all of these trends are taking us

- Examine the kinds of machines VLSI *wants* us to build

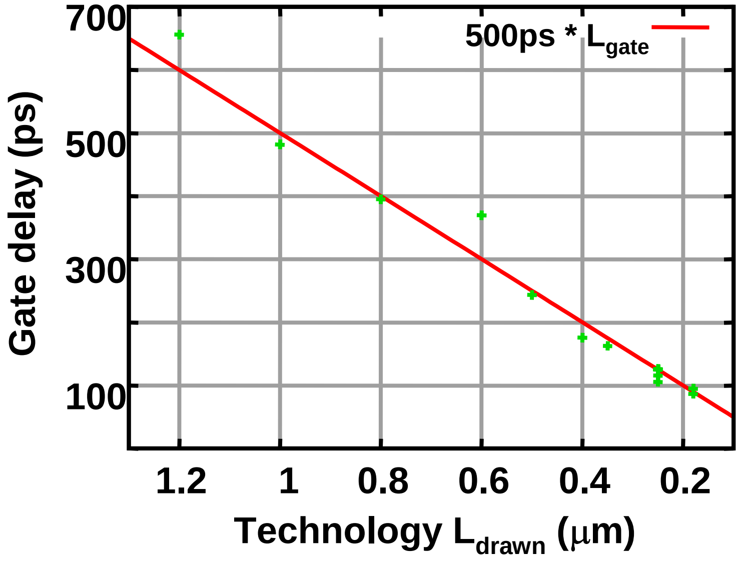
Ask: what’s wrong with this picture?

- Break the wire scaling paradigm with Proximity Communication

Start with gate delays

Wire scaling only matters if it’s slower/faster than gate scaling

- Use delay of a fanout-of-4 inverter (FO4) for normalization

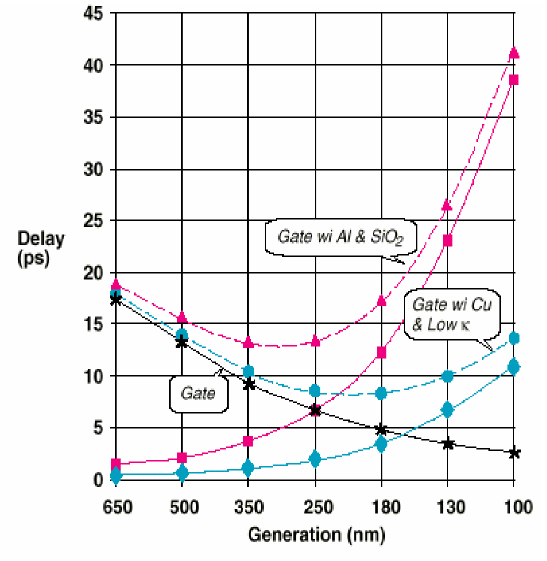


- Use FO4=L_{gate}¢500pS/μm up to 180nm
- Use FO4=L_{gate}¢250pS/μm at 130nm and beyond (poly undercut)

Gates are steadily speeding up. How about wires?

Wire scaling 101

We all remember this infamous slide from the 1997 ITRS



J. Meindl: "Ought to be the logo of the SRC"

It shows that predicting the future of wires is truly a dodgy business

- So we'll hedge our bets
  - Look at both aggressive and conservative projections

Be realistic about wires

For wires, only consider RC delays

- But include as many non-idealities as possible
  - Copper diffusion barrier (ALD vs non-conformal)
  - Surface scattering effects
  - Low-k dielectrics within and/or between metal layers
  - Thinning from polishing
- Get data from literature, ITRS, fabs, and magic 8-balls

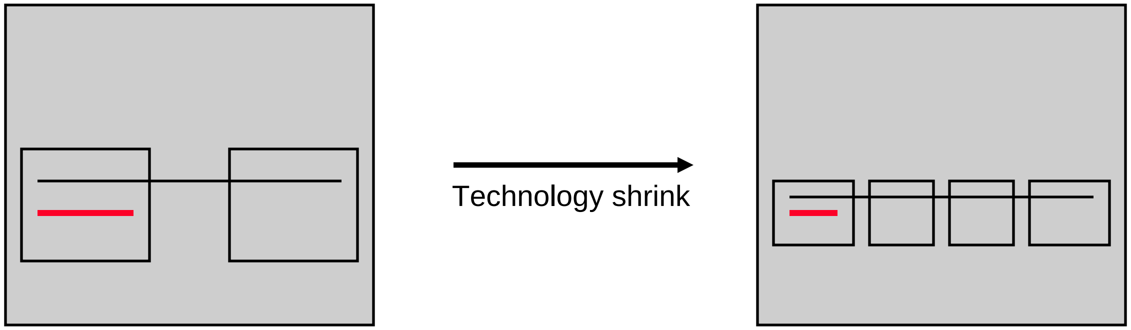
Keep in mind there are two kinds of wires (Sylvester, ICCAD ’98)

- Those that scale in length (“local” wires)
- Those that do not (“global” wires)

Scaled length wires

Wires of constant logical span get shorter each technology shrink

- For example, module-level wires

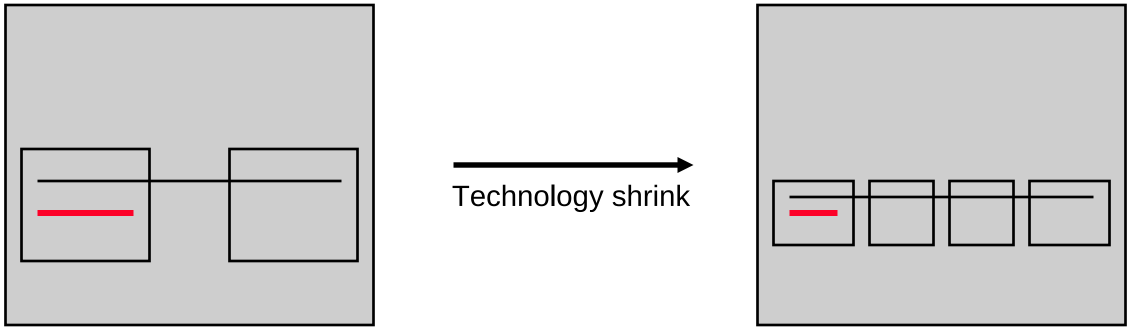


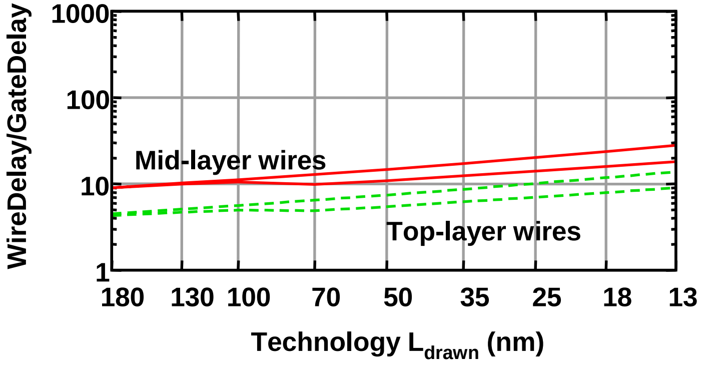
Technology shrink

Scaled length wires

Wires of constant logical span get shorter each technology shrink

- For example, module-level wires

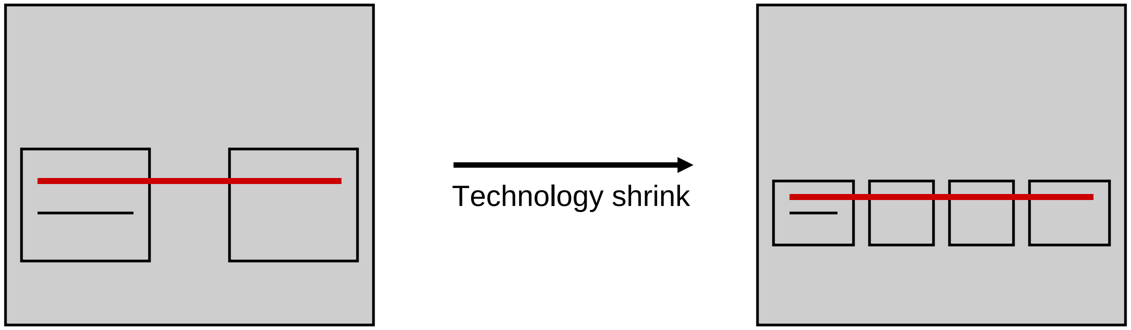




Repeated wire delays stay “constant-ish” with gate delay

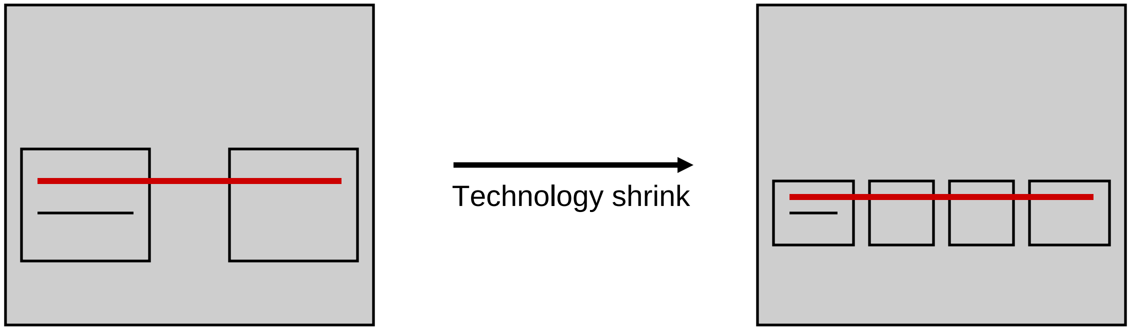
Global wires

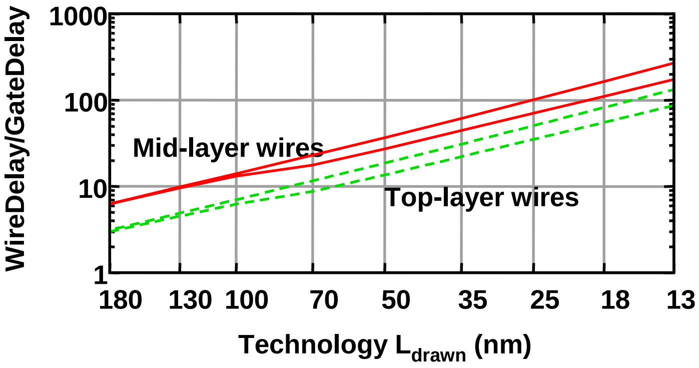
Global wires do not scale down in length



Global wires

Global wires do not scale down in length





Delay gets dramatically large relative to gates

And these are optimally repeated; bare wires are even worse!

What about bandwidth?

We’ve looked at the delay of short and long wires

- Important because it gives us the logical span of wires
- But what about the bandwidth of wires?

Note that bandwidth is per-cross-section. For a given wire length,

- I can make each wire wider
- Or I can have more wires

Repeated VLSI wires have enormous BW

- Cross-sectional bandwidth of one metal layer ~10 Tb/s
- (Okay, that ignores power, clocking, repeater costs. But still…)
- Wires can be thought of as slow but providing high bandwidth

What about bandwidth?

We've looked at the delay of short and long wires

- Important because it gives us the logical span of wires
- But what about the bandwidth of wires?

Note that bandwidth is per-cross-section. For a given wire length,

- I can make each wire wider
- Or I can have more wires ← usually the better choice

Repeated VLSI wires have enormous BW

- Cross-sectional bandwidth of one metal layer ~10 Tb/s
- (Okay, that ignores power, clocking, repeater costs. But still…)
- Wires can be thought of as slow but providing high bandwidth

What about coupled noise?

Wires are going to get skinnier and skinnier

- Aspect ratios up to 3, maybe 3.5, depending on whom you ask

We’re just going to have to live with some levels of noise

- Coupled noise on wires approximately $\frac{C_c}{C_c+C_{wire}} \cdot \frac{1}{1+\frac{\tau_a}{\tau_v}}$

There are some interesting solutions in the circuit space

- Noise cancellation, staggered repeaters (Naffziger, ISSCC ’01)

What about coupled noise?

Wires are going to get skinnier and skinnier

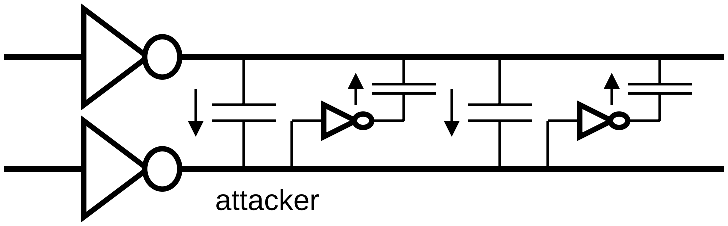
- Aspect ratios up to 3, maybe 3.5, depending on whom you ask

We’re just going to have to live with some levels of noise

- Coupled noise on wires approximately $\frac{C_c}{C_c + C_{wire}} \cdot \frac{1}{1 + \frac{\tau_a}{\tau_v}}$

There are some interesting solutions in the circuit space

- Noise cancellation, staggered repeaters (Naffziger, ISSCC ’01)



What about coupled noise?

Wires are going to get skinnier and skinnier

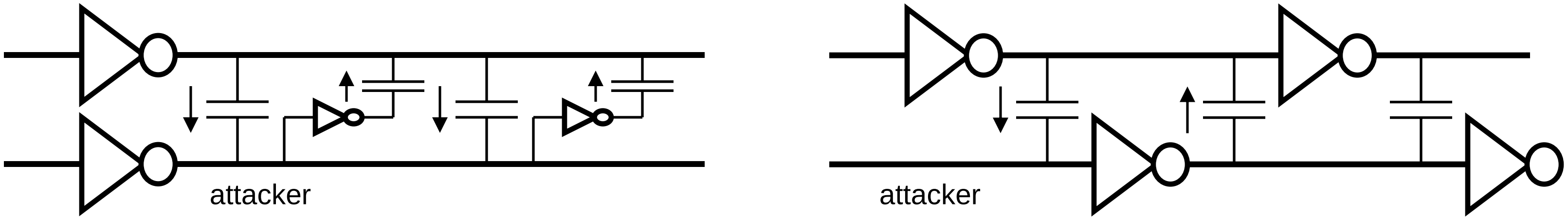
- Aspect ratios up to 3, maybe 3.5, depending on whom you ask

We’re just going to have to live with some levels of noise

- Coupled noise on wires approximately $\frac{C_c}{C_c + C_{wire}} \cdot \frac{1}{1 + \frac{\tau_a}{\tau_v}}$

There are some interesting solutions in the circuit space

- Noise cancellation, staggered repeaters (Naffziger, ISSCC ’01)



Solving noise

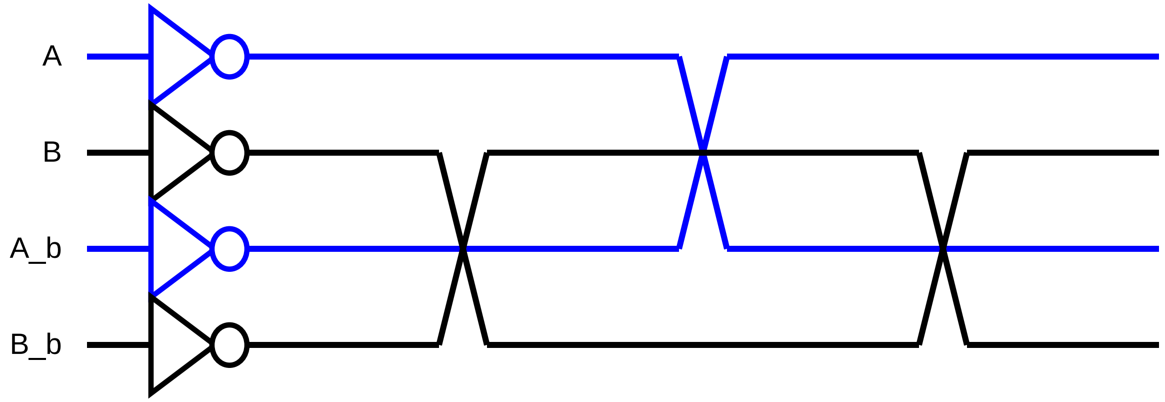
Always trade off a cheap resource (BW) for a dear one (noise)

- Differential signaling w/ twisting largely eliminates noise
  - Cost: differential receiver, minor via congestion, and 2x wires

Solving noise

Always trade off a cheap resource (BW) for a dear one (noise)

- Differential signaling w/ twisting largely eliminates noise
  - Cost: differential receiver, minor via congestion, and 2x wires



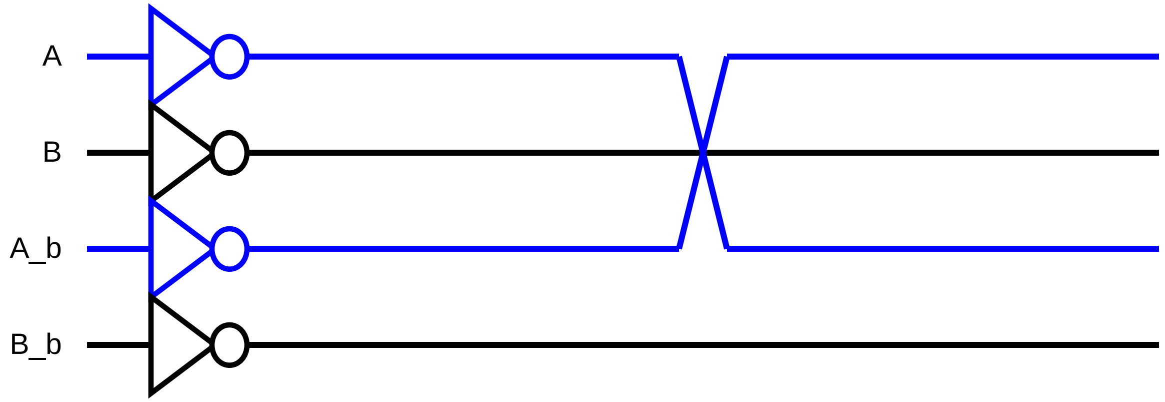
Full noise rejection

No Miller cap

Solving noise

Always trade off a cheap resource (BW) for a dear one (noise)

- Differential signaling w/ twisting largely eliminates noise
  - Cost: differential receiver, minor via congestion, and 2x wires



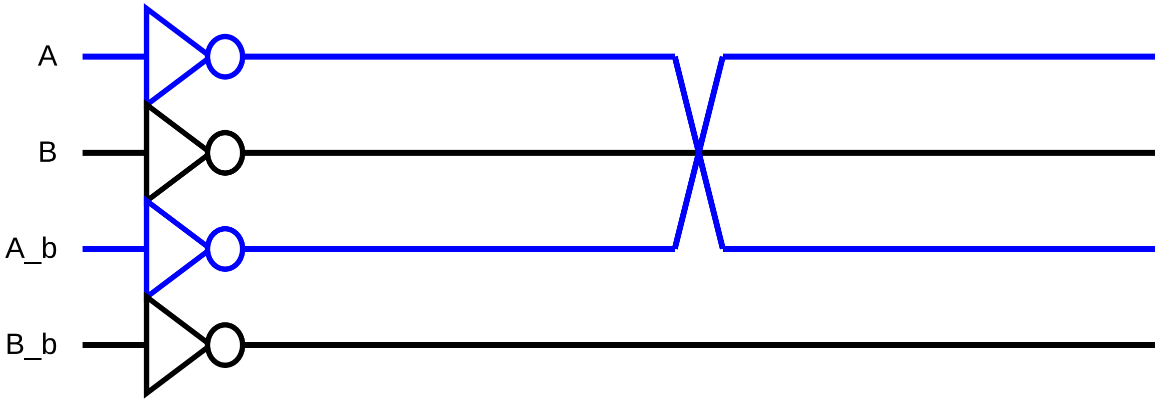
Differential noise rejection only

No Miller cap

Solving noise

Always trade off a cheap resource (BW) for a dear one (noise)

- Differential signaling w/ twisting largely eliminates noise
  - Cost: differential receiver, minor via congestion, and 2x wires



Differential noise rejection only

No Miller cap

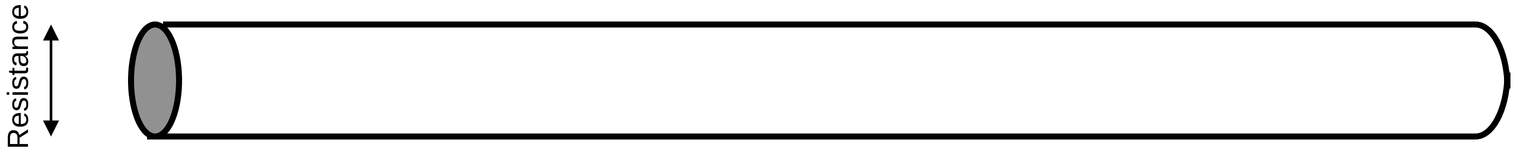
Caveats

- Not truly costless, and perhaps not yet a point-solution
- But a way to go in the long run, as wires get cheaper
  - Global router just runs differential wires and ignores noise

What about inductance?

We traditionally ignore L (even though it has a water analogy)

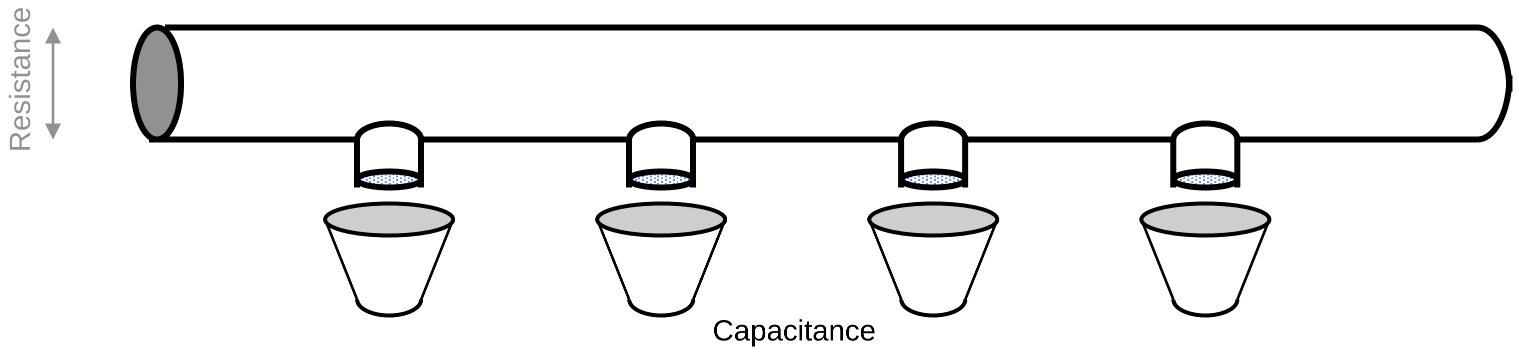
- Resistance (R) is related to the diameter of a water hose
- Capacitance (C) is related to
- Inductance (L) is related to



What about inductance?

We traditionally ignore L (even though it has a water analogy)

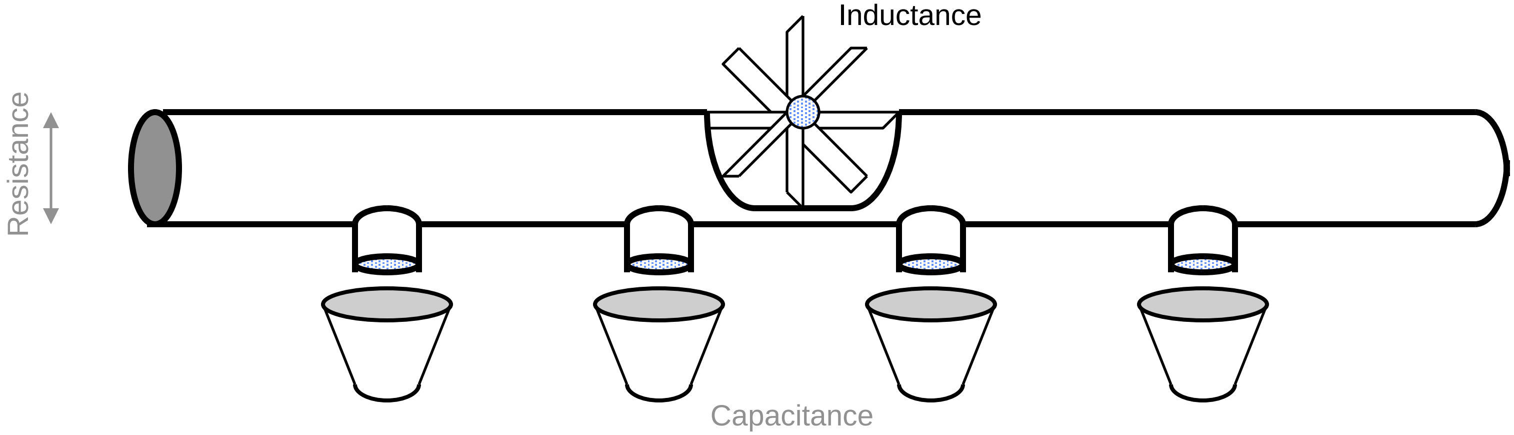
- Resistance (R) is related to the diameter of a water hose
- Capacitance (C) is related to mini-bathtubs along the hose
- Inductance (L) is related to



What about inductance?

We traditionally ignore L (even though it has a water analogy)

- Resistance (R) is related to the diameter of a water hose
- Capacitance (C) is related to mini-bathtubs along the hose
- Inductance (L) is related to… water wheels along the hose



But inductance has gotten lots of press recently

Can we ignore inductance?

The truth is, inductance can be exploited if you want to use it

- Sharpen edges of signals, cross-chip I/O, distributed amps

But for wires of typical dimensions, you can safely ignore it

- Realistic wiring systems have L=0.2-0.5nH/mm (FastHenry)
- Short wires don't care about inductance
  - When driver impedance > line impedance: $R_t > 2Z_0$
- Long wires don't care about inductance
  - When attenuation factor > 1: $0.5 * R_{wire} > Z_0$

It is worth considering for clocks and power grids, but not signals

- Especially for differential signals!

Can we ignore inductance?

The truth is, inductance can be exploited if you want to use it

- Sharpen edges of signals, cross-chip I/O, distributed amps

But for wires of typical dimensions, you can safely ignore it

- Realistic wiring systems have L=0.2-0.5nH/mm (FastHenry)
- Short wires don't care about inductance < 2mm for 180nm
  - When driver impedance > line impedance: $R_t > 2Z_0$
- Long wires don't care about inductance
  - When attenuation factor > 1: $0.5{*}R_{wire} > Z_0$

It is worth considering for clocks and power grids, but not signals

- Especially for differential signals!

Can we ignore inductance?

The truth is, inductance can be exploited if you want to use it

- Sharpen edges of signals, t-lines, distributed amps

But for wires of typical dimensions, you can safely ignore it

- Realistic wiring systems have L=0.2-0.5nH/mm (FastHenry)
- Short wires don't care about inductance < 2mm for 180nm
  - When driver impedance > line impedance: $R_t > 2Z_0$
- Long wires don't care about inductance > 2mm for 180nm
  - When attenuation factor > 1: $0.5{*}R_{wire} > Z_0$

It is worth considering for clocks and power grids, but not signals

- Especially for differential signals!

What does wire scaling tell us?

- CAD tools dealing with wires need to improve
  - Even for scaled-length wires → chip complexity growing

- We need to design wire-centric circuits
  - Contain noise effects, avoid sensitivity to variations

- We need to build wire-aware architectures
  - Accept that long wires are slow, and design around them
  - Avoid “70% of our cells are repeaters” (Saxena, ISPD ’02)

- We ought to move towards modular machines
  - Computation relies on local, not long-range, communication
  - Use huge available bandwidth of the long-range wires

Outline

First, review basic wire scaling trends

- Pay attention to hot-button topics like noise and inductance

Consider high-performance CPU trends

- Focus on frequency, power and cost

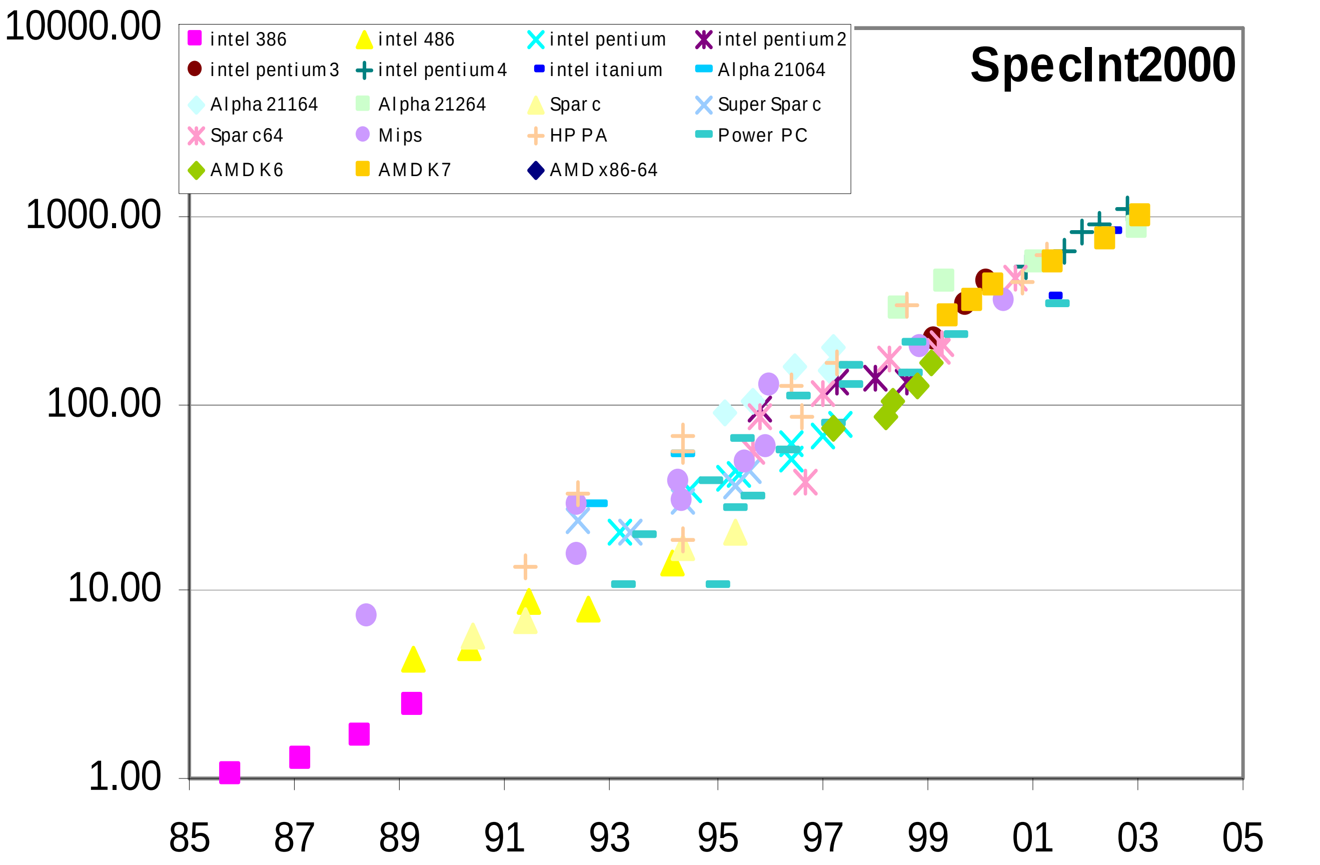
Look at where all of these trends are taking us

- Examine the kinds of machines VLSI *wants* us to build

Ask: what's wrong with this picture?

- Break the wire scaling paradigm with Proximity Communication

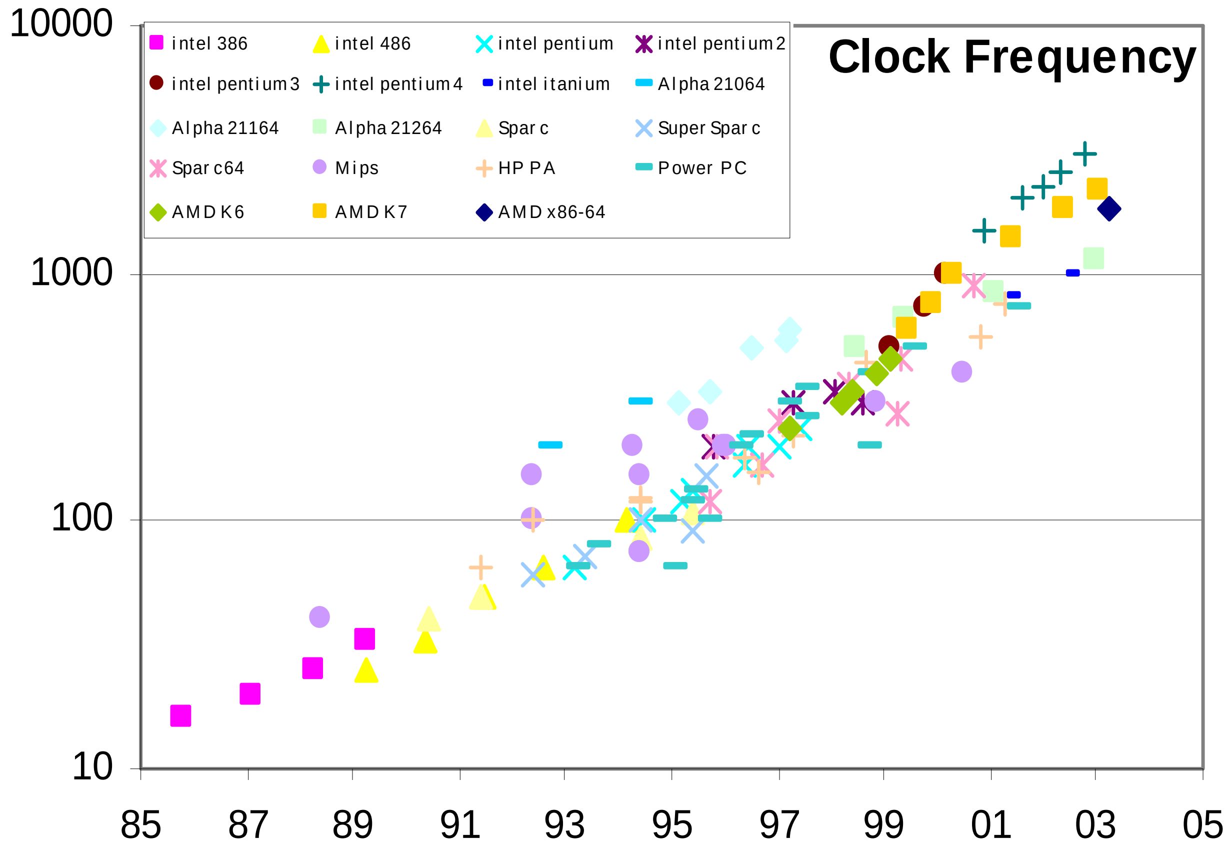
Can we maintain high-performance trends?



Source: Horowitz, 2005

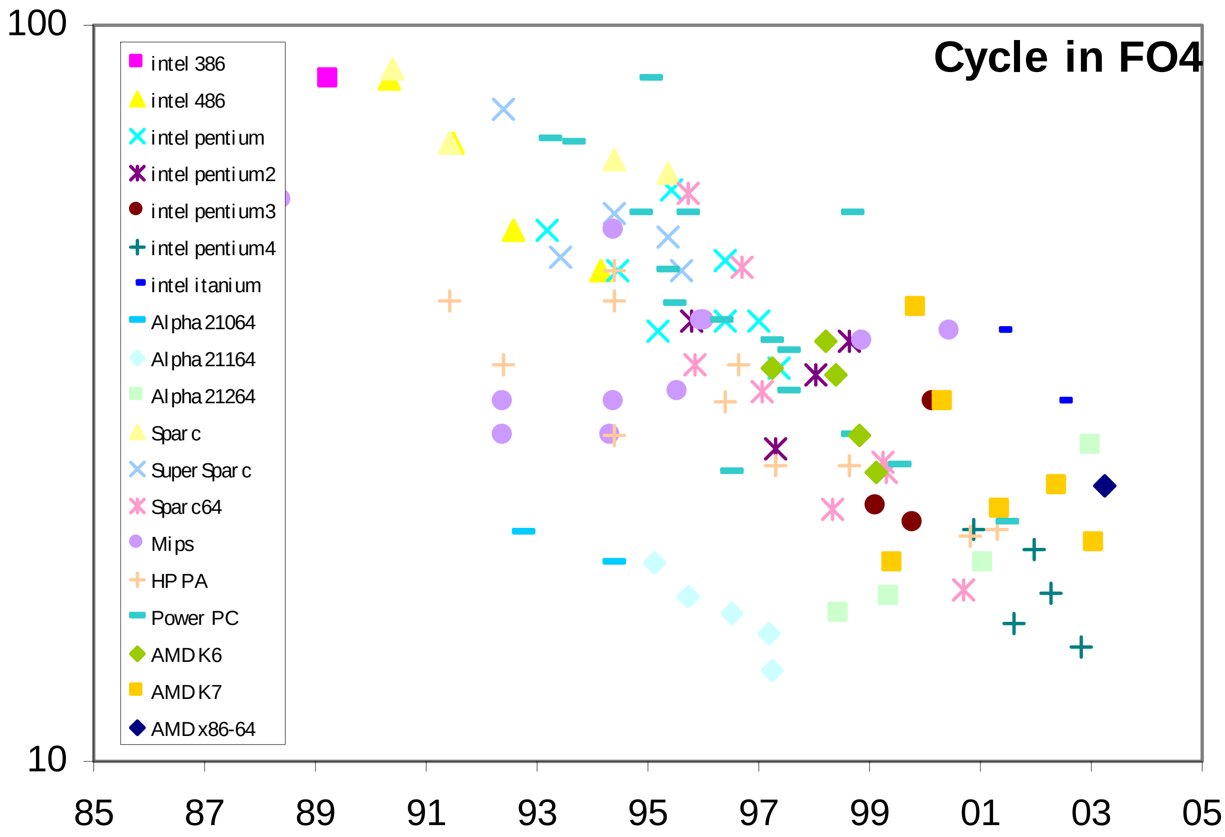
Gains come from architecture (CPI) and circuits (frequency)

We’ve pushed pretty hard on frequency



This trend is faster than underlying technology improvements!

Normalize frequency to process



Circuits have indeed gotten faster, relative to underlying technology

- Wait…doesn’t this curve eventually hit 0?

Cycle times in FO4 will level out

FO4 cycle times have fallen pretty far

- 16 FO4 per cycle in Intel short-tick machines (Pentium4)

They probably won’t fall much further

- Fast clocking becomes too onerous in both power and timing
- A timing signal faster than 8FO4 is practically a sine wave

There is some disagreement on the actual limits

- 6-8 FO4s optimal for performance (Hrishikesh, ISCA ’02)
- 18 FO4s optimal if you include overhead (Srinivasan, ISM ’02)

Frequency scaling will throttle back to “what the process gives us”

So push on CPI for performance scaling?

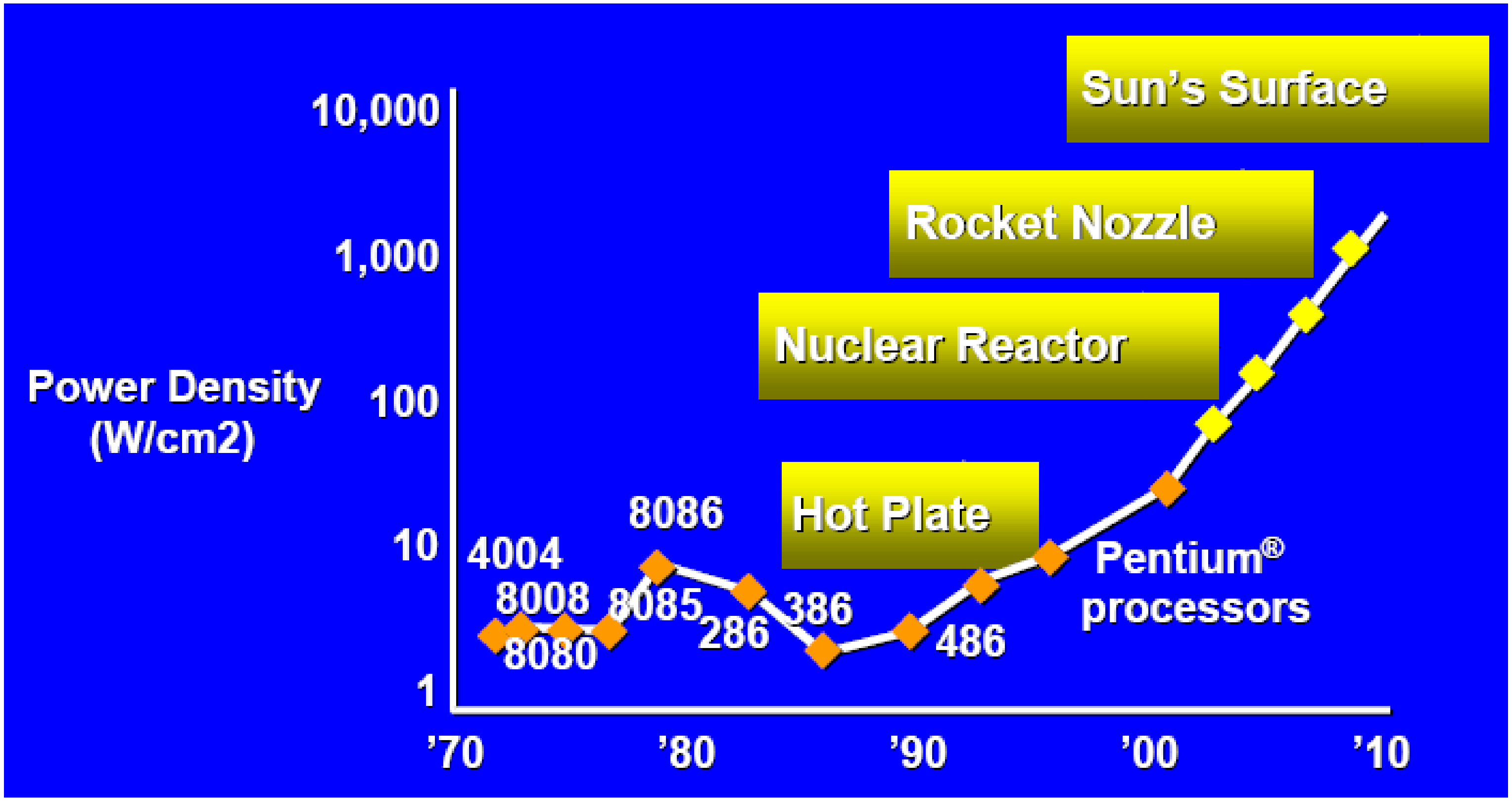
I’m not an architect, nor do I play one on TV

- But I can see that not much more ILP is available
  - Even exploiting the ILP that we can see is expensive
  - Cost of finding more is prohibitive
- And creating parallelism through speculation is costly
  - Very power inefficient to do work that we’ll throw away

Instead, architectures are moving to thread-level parallelism (TLP)

- Multiple compute cores with loosely coupled control
- Global packets get scheduled/routed between cores
  - High latency, yes, but also high bandwidth
- The networks-on-a-chip model

Power trends of high-performance machines



Source: Gelsinger, Intel, ISSCC2001

Slide is long in the tooth and overused, but it gets the point across

- Avoiding high power density has become a design priority

More on power trends

Today's CPUs are already power constrained

- More performance is available in our chips
  - If we could afford to dissipate the heat
  - If we could carry (lug) the energy around in batteries

Supply voltage has stopped scaling with channel length

- Due to V_{dd} versus V_{th} concerns, performance and leakage
- Power = CV^2f
- We've gotten off of the "constant power density" track

Managing power

The old VLSI question: “What do you do with a billion transistors?”

- The new VLSI answer: “Don’t run them all at once!”

We should build modular machines with lots of compute cores

- Don’t power them all on, all at once
  - Just the ones specific to your application
  - A high-bandwidth global interconnect moves data around
- An opportunity for specialization in the modular cores
  - Different applications use different mixes of functional blocks
  - Exploit this reconfiguration in the architecture

What’s the cost of money?

Complexity drives design costs (Gartner/Dataquest 2003)

- 180nm ASIC averages $4M in design costs
- 130nm ASIC averages $10M in design costs
- 90nm ASIC averages $25M in design costs

Complexity drives design team size (Sematech 2002)

- Transistors/die growing at a 68% CAGR
- Productivity (transistors/person-month) growing at 21% CAGR

And don’t forget the mask sets: $0.75M in 90nm and getting worse

Who can afford to build a big chip today? Tomorrow?

Managing costs

Blind extrapolation says

- Only Exxon and Wal-Mart will be able to afford to build CPUs

The solution? Build universal computing chips

- More flexible than today's CPUs
  - Integer code, streaming code, vector code, microkernels...
- Fast like a dedicated ASIC – or at least within a factor of 2-3x
  - Certainly faster than an FPGA

Build this machine with… (you guessed it) reconfigurable modules

- Local fast computation within each module
- Slow but high bandwidth global communication

Outline

First, review basic wire scaling trends

- Pay attention to hot-button topics like noise and inductance

Consider high-performance CPU trends

- Focus on frequency, power and cost

Look at where all of these trends are taking us

- Examine the kinds of machines VLSI wants us to build

Ask: what’s wrong with this picture?

- Break the wire scaling paradigm with Proximity Communication

Lots of trends point in the same direction

- Wire scaling pushes us towards *modular machines*
  - Rely on local wires for computation
  - Explicit high latency and high bandwidth of global communication
- Architecture scaling pushes us towards *modular machines*
  - ILP is largely mined out in big monolithic machines
  - Modularity exploits thread-level parallelism
- Power scaling pushes us towards reconfigurable *modular machines*
  - We can’t power our transistors all at once, anyway
  - Modularity enables an application-specific functional mix
- Cost scaling pushes us towards reconfigurable *modular machines*
  - Reconfigurability allows for universal computing systems
  - Modularity allows for greater range of reconfigurability

Have I mentioned “modular machines”?

What do I mean by a reconfigurable modular machine, anyway?

- Subdivide die into individually enabled functional blocks

- Blocks are not identical and can be reconfigured
  - Memories that can look like a cache, a RAM, or a buffer
  - Adders that do 64b, and four saturating 16b, and so on…
  - Multipliers that do multiply, XOR-multiply, and MACs

- Communication between blocks has architecturally explicit delay
  - Latency visible to the programming model
  - Lots of bandwidth available to be used
  - Network-on-a-chip with static or dynamic routing

People *are* building what VLSI wants

Lots of work in academia

- MIT Raw project: a scalable microprocessor with 16 tiles
- Stanford Smart Memories: 64 compute tiles w/mesh network
- Stanford/MIT Imagine chip: 8 compute clusters, stream registers
- UT-Austin TRIPS: Systolic-like compute array in “malleable” grid

And movement in this direction in industry

- Multi-core processors are explicitly modular
  - Sun, IBM Power5, Intel Montecito and P4D, Fujitsu and AMD
- Chip-multi-threading is the next step in this chain
  - Sun’s Niagara

Outline

Review of wire scaling trends

- Paying attention to hot-button topics: noise, inductance

Overview of high-performance trends

- What is driving our fast CPUs?

Look at where these trends are taking us

- Building machines that VLSI *wants* us to build

What’s wrong with this picture?

- Breaking the wire scaling paradigm w/ Proximity Communication

So far, so good?

VLSI wants us to build *big* modular chips

- To overcome the disparity between on-chip and off-chip BW
- Raw = 330mm^2, Montecito > 500mm^2, Niagara ¼ 340mm^2

If you split functionality between chips your performance suffers

- Global on-chip wires may be slow, but they have big bandwidth
- Chip-to-chip wires are both slow and low bandwidth
  - Can’t have that many high-speed serial links

Big chips maximize the available functionality

- With more blocks, the reconfigurability can be more universal
- With more blocks, the overall performance can be higher

The drawback of size

But big chips put us on the wrong side of economics

- Die size and yield are not compatible
- Big chips are awfully expensive

Take the modular machine notion to its natural extreme

- Wafer-scale integration (uh-oh)
- Nobody has made the economics of *0 yield* work yet

The underlying problem

- "Impedance mismatch" between on-chip and off-chip bandwidth
- Gets worse with interconnect scaling
- Driven by high-performance VLSI trends

Break the interconnect scaling paradigm

What if we could match off-chip bandwidth to on-chip bandwidth?

- Make your modular machine out of smaller discrete chips
  - Retain high bandwidth long-range communication
  - Some additional global latency acceptable (it’s slow already)

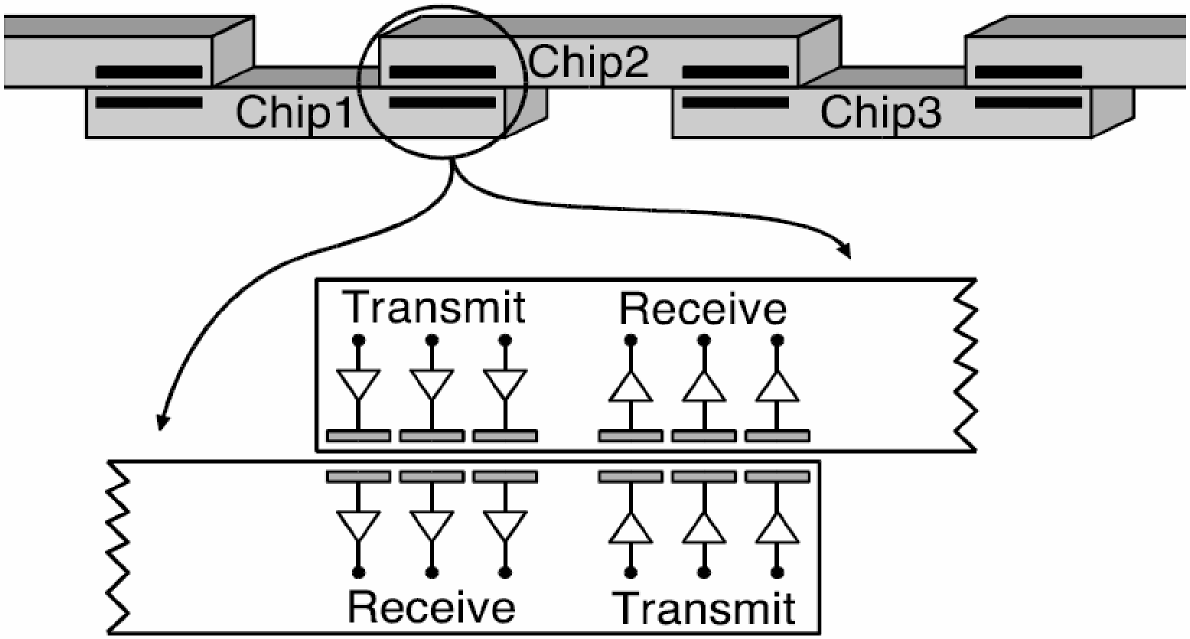
Benefits?

- Modularity still fits with scaled wire delay and TLP
  - Keep the same overall performance
- Reconfigurability turns into application-specific assembly
  - With much less overhead
- Individual die costs are far lower
  - Not all of the chips need to be at the same technology

Proximity Communication

A chip-to-chip signaling scheme from Sun Labs

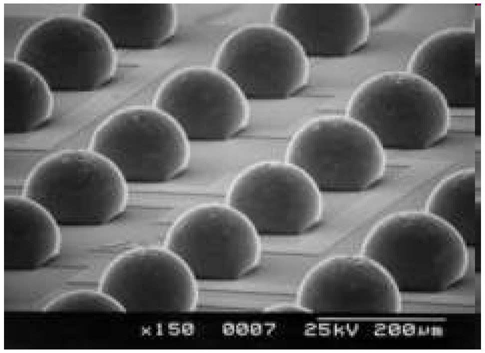
- Make capacitors out of two metal pads on two adjacent chips
- Place them face-to-face and send data across the capacitor

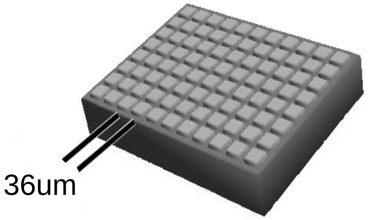


Size of Proximity Communication structures

Pads are small and can be packed with very high density

- Under scaling, they keep a density advantage over area balls

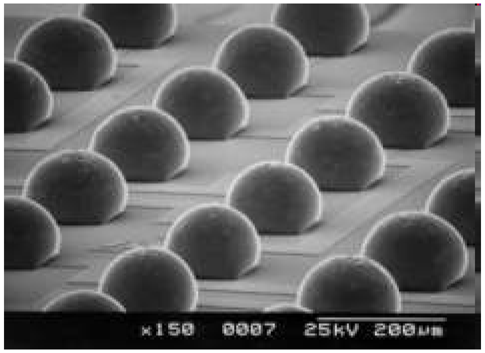


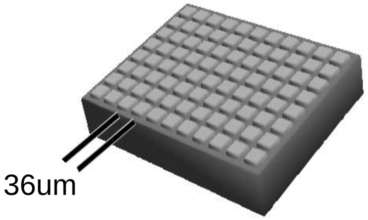


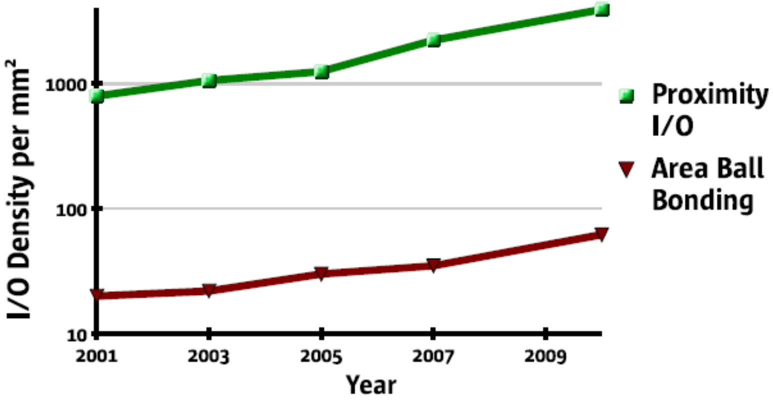
Size of Proximity Communication structures

Pads are small and can be packed with very high density

- Under scaling, they keep a density advantage over area balls







Benefits of Proximity Communication

Provides high-bandwidth off-chip communication

- Metal pads are made of on-chip structures that scale
- Off-chip bandwidth tracks on-chip bandwidth
- Allows modular machines to be built from separate (small) die

Avoids conductive/permanent attachment between chips

- Much lower power because we can safely omit ESD structures
- Die can be easily tested, assembled, removed and replaced
- Application-specific die configurations are possible

Offers a physical substrate for modular reconfigurable machines

Challenges of Proximity Communication

Mechanical misalignment degrades communication channels

- Static (assembly) and dynamic (thermal or vibrational) important
- Physical alignment structures become critical to the architecture

Capacitors block transmission of DC levels

- Need biasing at the receiver

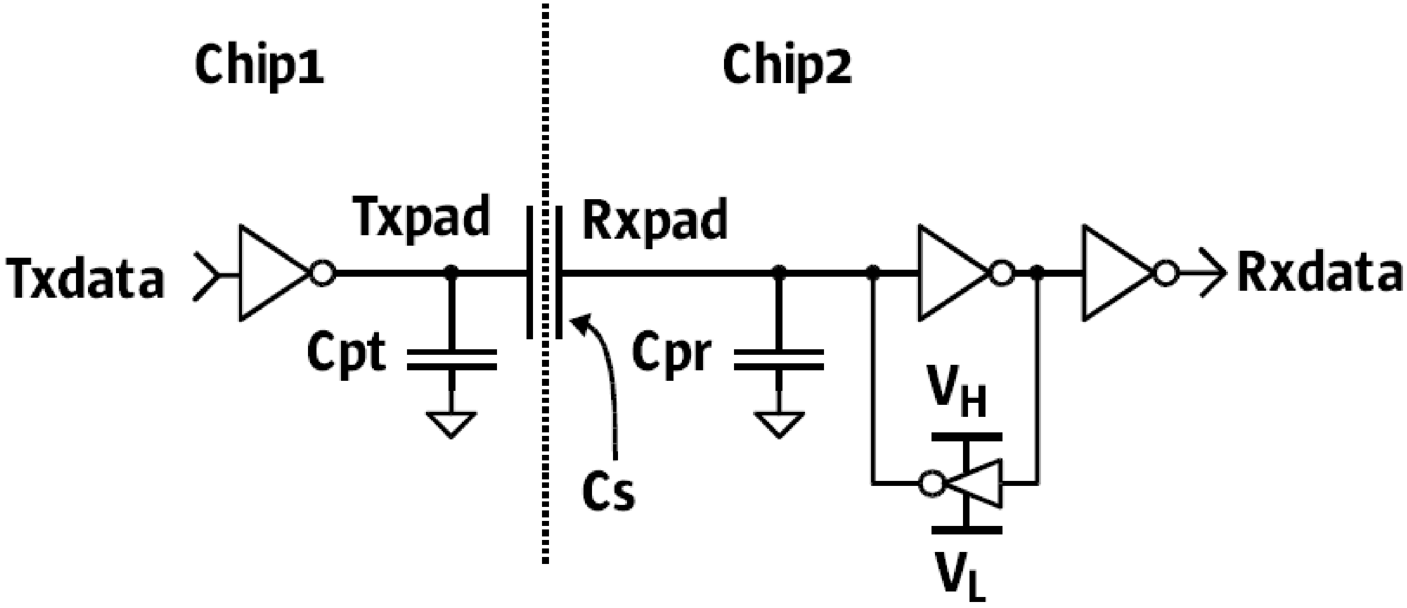
Channel requires amplification of small-swing signals

- Capacitive divider gives signals ¼ 10% of supply
- Swing also dependent on Z-axis misalignment (separation)

A sample Proximity circuit

Receivers use weak feedback to bias the input around V_{trip}

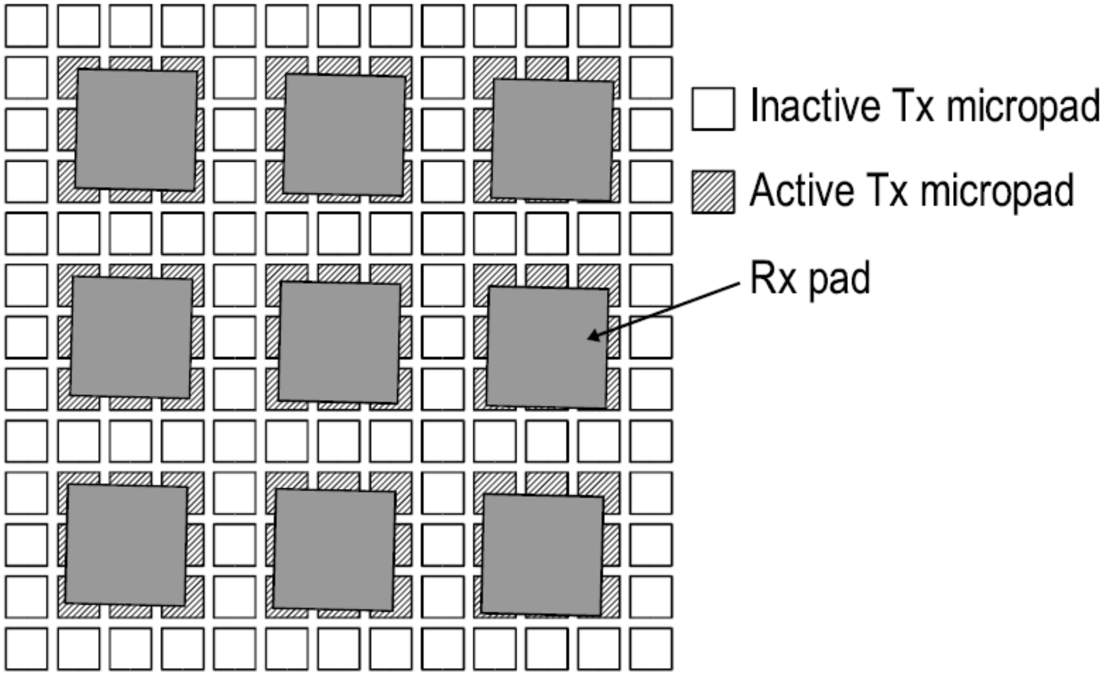
- This circuit example uses an unclocked receiver
- Clocked amplifiers provide more sensitivity at complexity cost



Dealing with misalignment

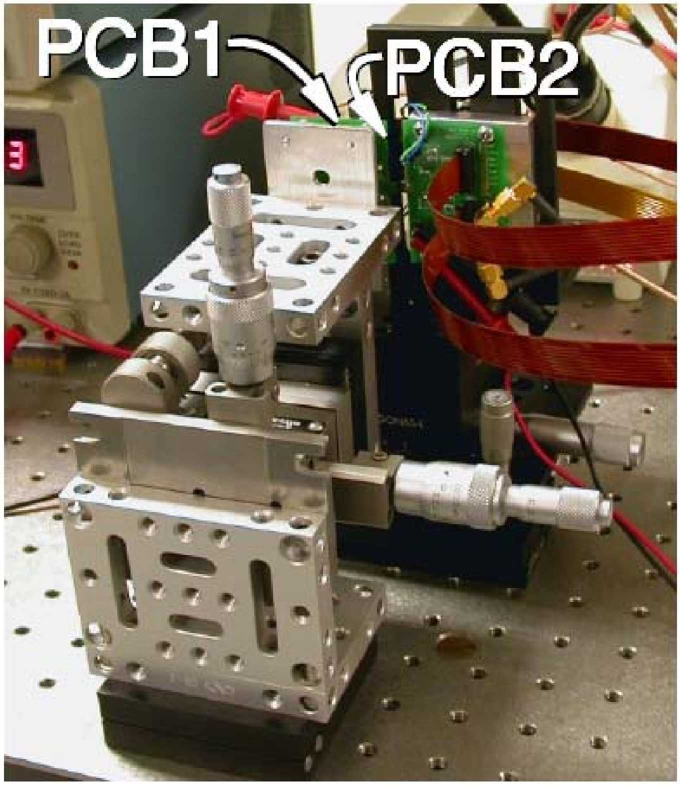
Data-steering circuits dynamically fix in-plane misalignment

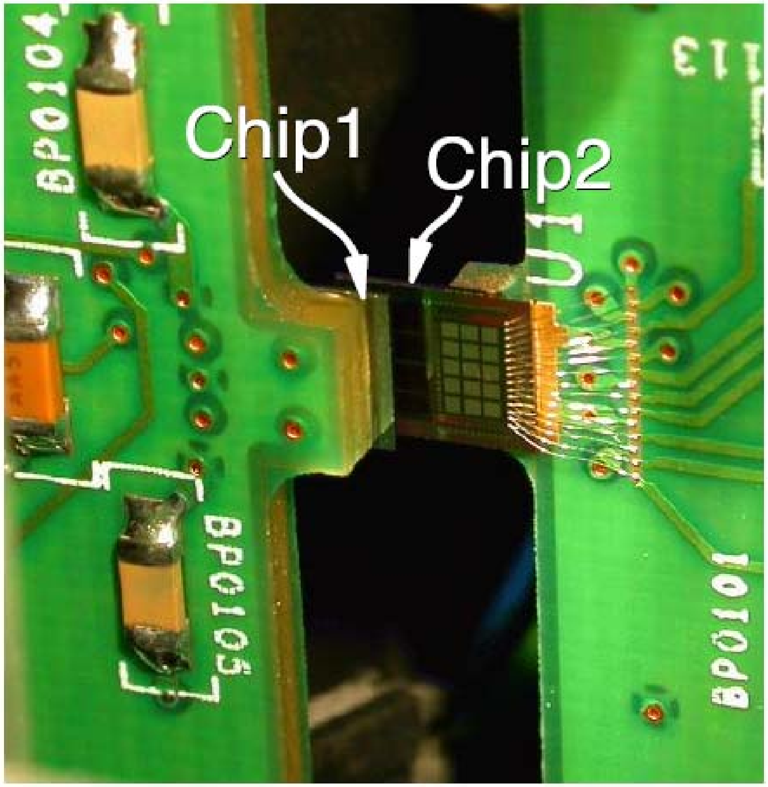
- Transmit pads are more numerous and finer than receive pads
- Misalignment verniers (not shown) detect required steering



Testing Proximity Communication chips

Place chips face-to-face with six-axis positioning system





Proximity Communication status

Working silicon shows channels communicating

- Bit periods of 6 FO4
- Transmission power on par with on-chip wire power dissipation
- Pad sizes around 300λ on edge
- BER < 10^{-12} (currently limited by test instrumentation)

Currently working towards:

- 100s or 1000s of pads at densities > 2000 pads/mm^2
  - Off-chip bandwidth that matches on-chip wire bandwidth
- Asynchronous handshaking control for robustness
  - Off-chip protocol that matches on-chip async networks

What Proximity Communication gives us

Offers us off-chip bandwidth that matches on-chip bandwidth

- Breaks today's interconnect scaling paradigm (in a good way)

Allows us to consider dividing large die into smaller die

- With no large power, delay, or bandwidth penalties

Enables systems made of many such small die facing each other

- Chips can be from different technology generations
- Chips can be easily tested, inserted, removed and replaced

Let us ask interesting questions

- What kind of system should I build with this kind of off-chip BW?

How this helps our scaling story

We can still build what technology trends push us to build

- Wire delays argue for modularity (scaled- vs. fixed-length wires)
- Architectures argue for modularity (TLP beats out ILP)
- Power argues for modularity (only enable what you need)
- Cost argues for modularity (build reconfigurable systems)

With Proximity Comm. modular systems aren’t huge and costly

- They’re made up of numerous small chips
- Chips replaceable per application or per reliability requirements

All we need to do

- Finish the research on Proximity Communications ☺

Summary

Examined wire scaling and also architecture, power, cost curves

- All pushing us towards modular reconfigurable machines

The problem is, top-performance modular ULSI chips will be huge

Fundamental issue is a mismatch between on-chip and off-chip BW

- Driven by trends of high-performance ULSI

Proximity Communication offers to match on-chip and off-chip BW

- Promising area of research currently studied at Sun Labs
- May enable cost-effective modular, reconfigurable systems

http://research.sun.com/async