The Scaling of Interconnect Buffer Needs

Prashant Saxena

Advanced Technology Group
Synopsys Inc.
Hillsboro, OR, USA

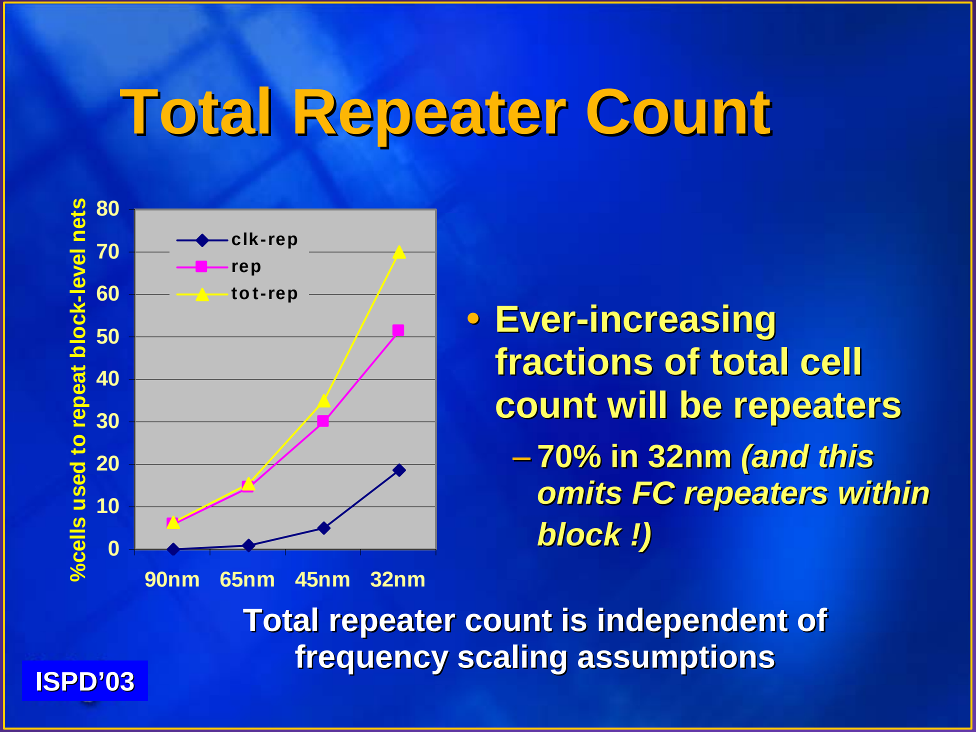
International Workshop on
System Level Interconnect Prediction

Munich, Germany

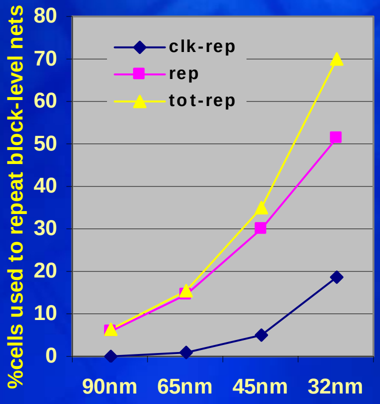
March 5, 2006

The Sky is Falling!

- Cong'97
  - 800K buffers at 50 nm
- Saxena'03
  - 70% buffers at 32 nm



Total Repeater Count



- Ever-increasing fractions of total cell count will be repeaters
  - 70% in 32nm *(and this omits FC repeaters within block !)*

Total repeater count is independent of frequency scaling assumptions

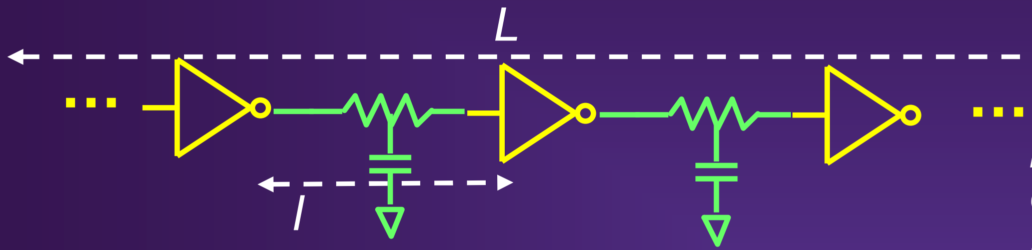
ISPD'03

… or, is it?

- ISPD'03 data point projected historical trends
- But are those trends sustainable? Necessary?
- What about alternative scaling scenarios?
    - Relax assumptions underlying data point

Optimal inter-buffer length

- First order (lumped parasitic, Elmore delay) analysis



***R_d* – On resistance of inverter**
***C_g* – Gate input capacitance**
***r,c* – Wire res., cap. per μ**
***h* – Gate width**

- **Assume *N* identical buffers with equal inter-buffer length *l* (*L = Nl*)**

$$T = N\left[R_d\left(C_g + cl\right) + rl\left(C_g + cl\right)\right]$$

$$= L\left[rcl + \left(rC_g + R_d c\right) + \frac{1}{l}\left(R_d C_g\right)\right]$$

- **For minimum delay,**

$$\frac{dT}{dl} = 0 \quad \Rightarrow \quad L\left[rc - \frac{R_d C_g}{l_{opt}^2}\right] = 0 \quad \Rightarrow \quad \boxed{l_{opt} = \sqrt{\frac{R_d C_g}{rc}}}$$

SYNOPSYS®

Optimal interconnect delay

- For optimally sized buffers (using *dT/dh = 0*),

$$\boxed{rC_g = R_d c}$$

- Substituting l_{opt} back into the interconnect delay expression:

$$T_{opt} = L\left[\underbrace{\left(\sqrt{R_d C_g rc} + R_d c\right)}_{\text{Device delay}} + \underbrace{\left(\sqrt{R_d C_g rc} + rC_g\right)}_{\text{Wire delay}}\right]$$

Delay grows linearly with *L* (instead of quadratically)

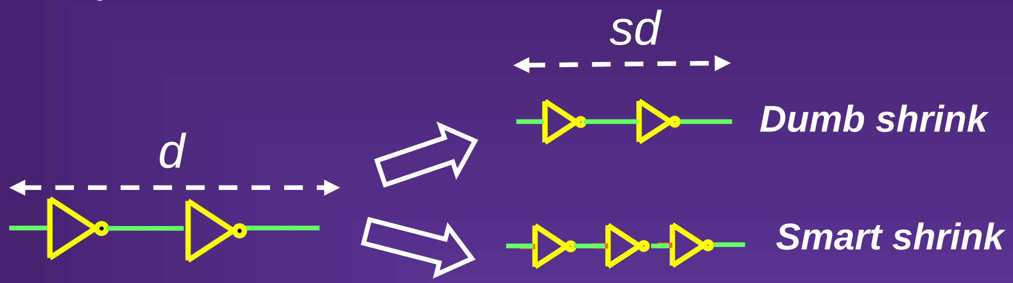
- For optimal sized buffers,

Equal delay in wire and device

(Constant ratio even under more accurate delay models)

Inter-buffer length scaling

- With scaling, devices speed up but wires don't
  - Scaling upsets the delay balance between buffers and wires
  - To restore balance, add more buffers



- Optimal inter-buffer length scales as:

$$l_{opt_{new}} = \sqrt{\frac{(R_d C_g)s}{\left(\frac{r}{s^2}\right)c}} = l_{opt}\,(s\sqrt{s})$$

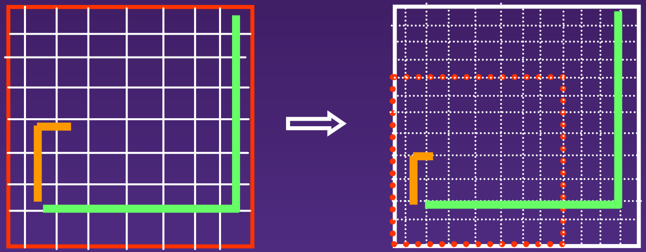
- **Optimal inter-buffer length scales by *s*$^{1.5}$ (not *s*)** *(s=0.7)*

Experimental Methodology for ISPD’03 Data Point

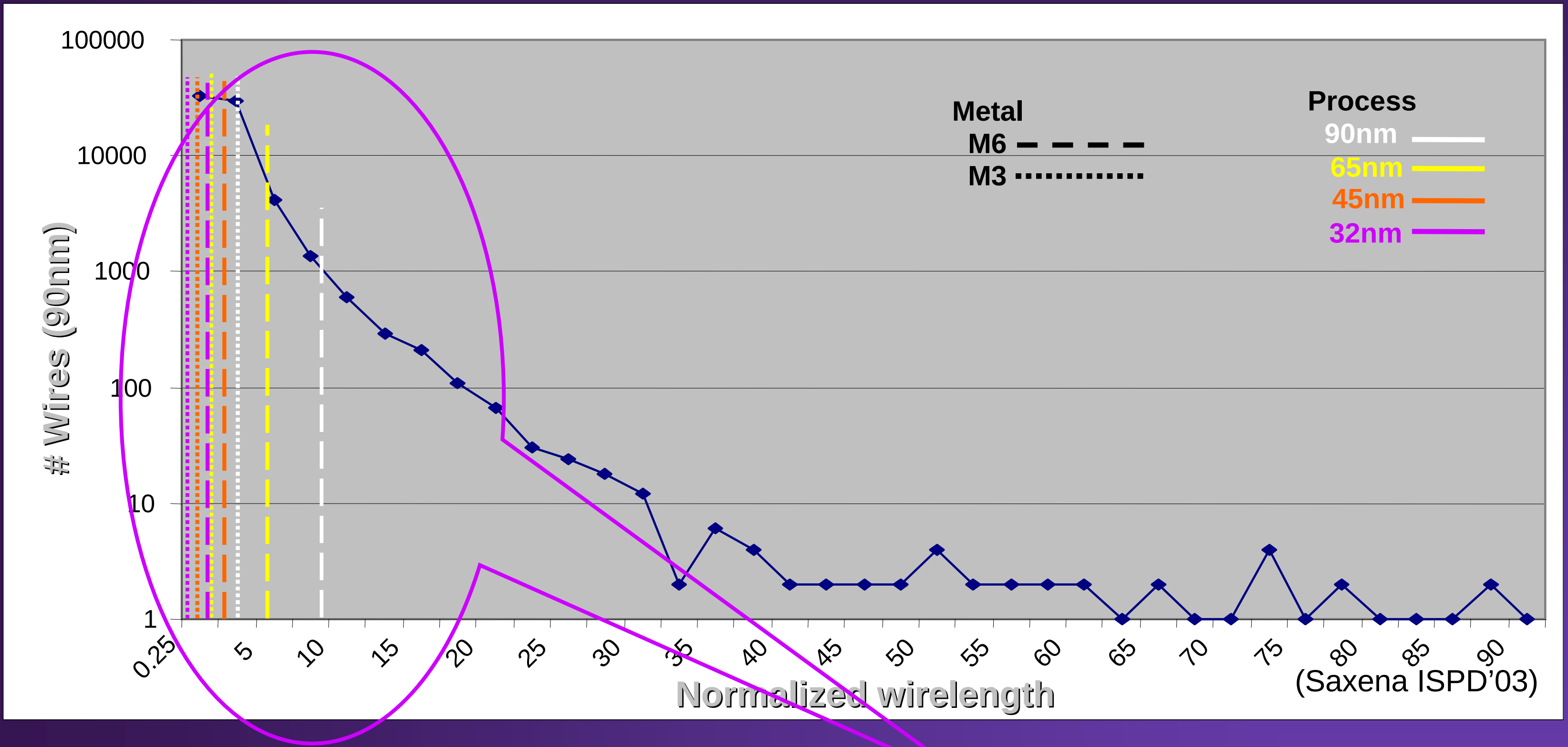
- Spice simulation of an “infinite” uniformly buffered line
  - Device scaling calibrated against existing process technologies
  - Devices ~30% faster per node
  - Geometric shrink of wires
- Determination of optimal inter-buffer separation
  - for different process nodes
  - for different metal layers at each node

Experimental Methodology - II

- Extract wiring distribution of a synthesized block
  - Pre-buffering histogram
  - ~80K cells at 90 nm
- Scale wiring distribution to future process nodes
  - First order design scaling assumptions
    - Block area invariant ➔ #nets doubles
    - Wirelength distribution unchanged
- Determine #buffers for each net
  - using corresponding inter-buffer separation

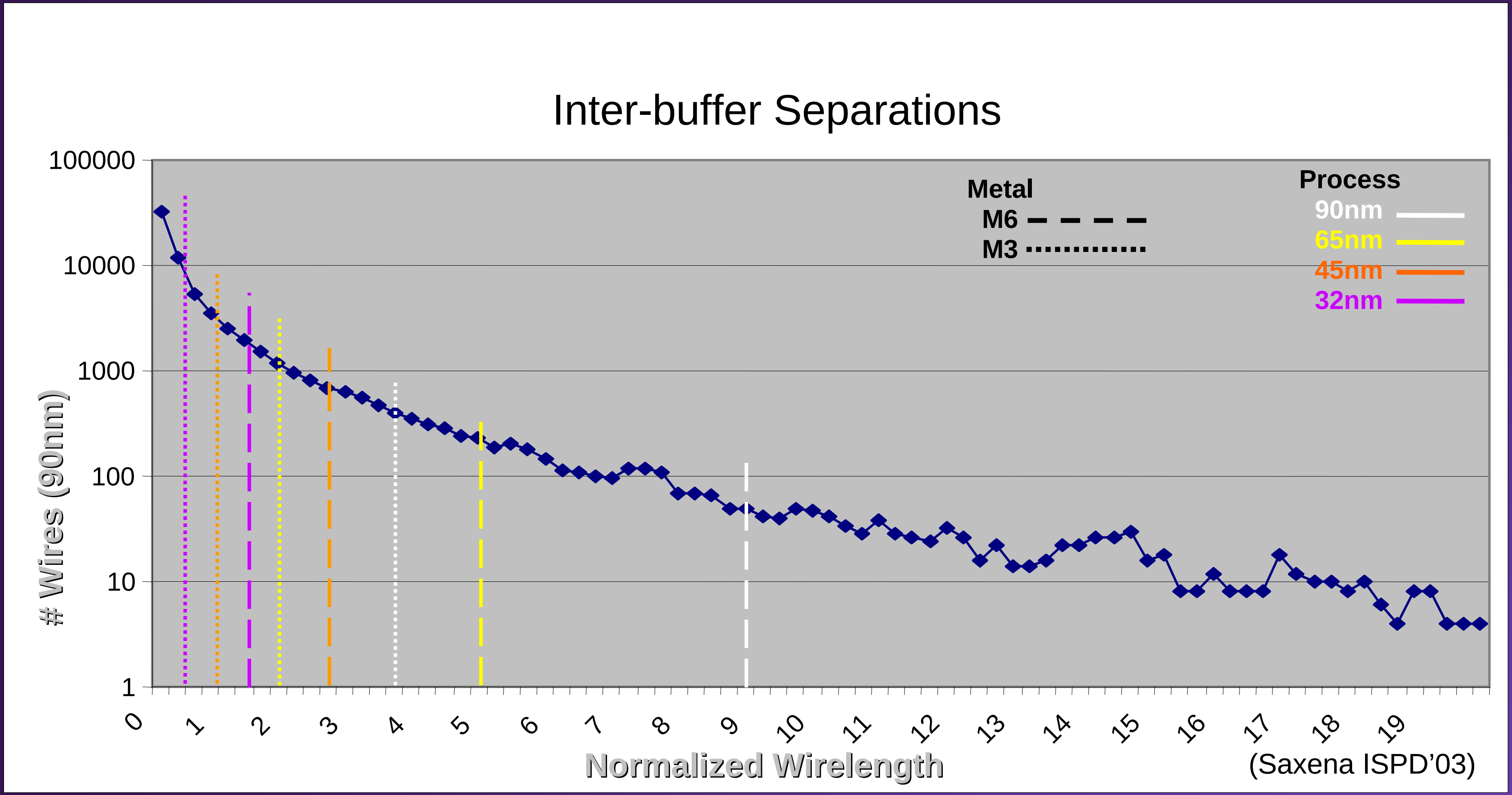


Block Wiring Histogram and Inter-buffer Separation



Optimal separations moving rapidly to the left… *(zoomed view coming up)*

Block Wiring Histogram: *Zoomed View*

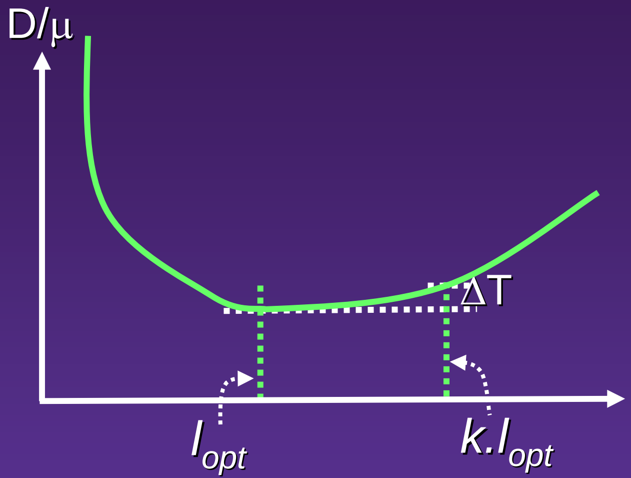


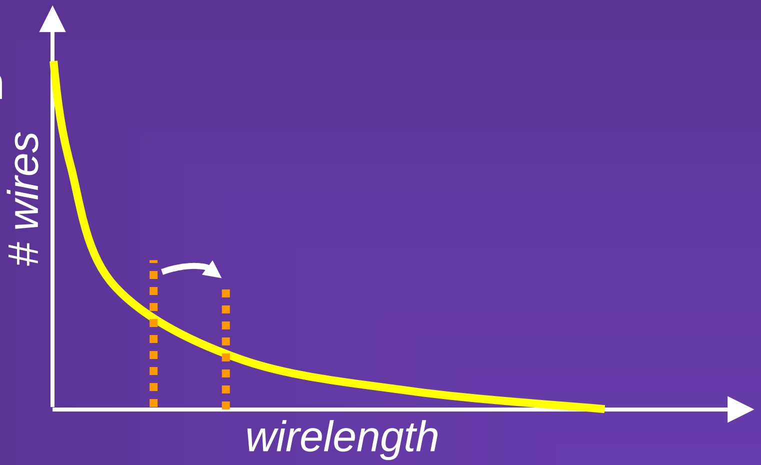
Scaling Assumptions

- Optimal inter-buffer separation
- Invariant block area
- Ideally shrunk wires
- Proportional layer assignment
- Ideal device speedup
- Invariant wiring distribution shape

Increased Inter-buffer Separation

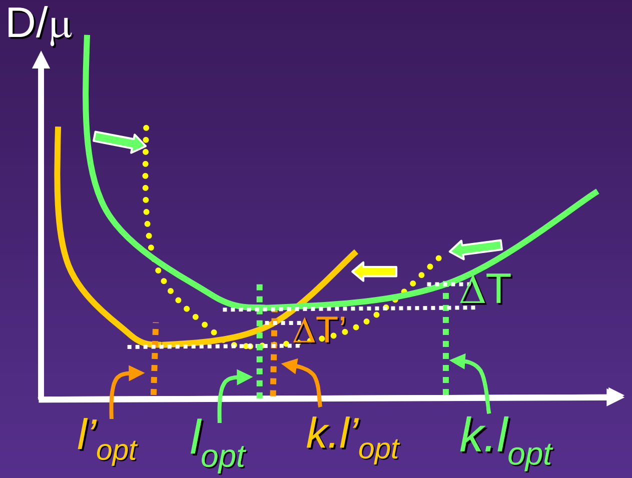
- **Signal speed vs. inter-buffer separation curve is quite flat around optimum**
- **With increased separation, significantly fewer nets require buffers**
  - **Histogram gets steeper to the left** (even on semi-log plot)
  - **Previous generation's buffer fraction for ~70% back-off**
  - **Delay and noise degradation**





Increased Inter-buffer Separation

- **Normalized delay degradation** $[T(k.l_{opt}) - T(l_{opt})] / T(l_{opt})$ **should not worsen with scaling**
  - **Even** $T(l_{opt})$ **scales at** $s^{0.5}$ **(not** s**)**
- **For this,** k **cannot grow!**



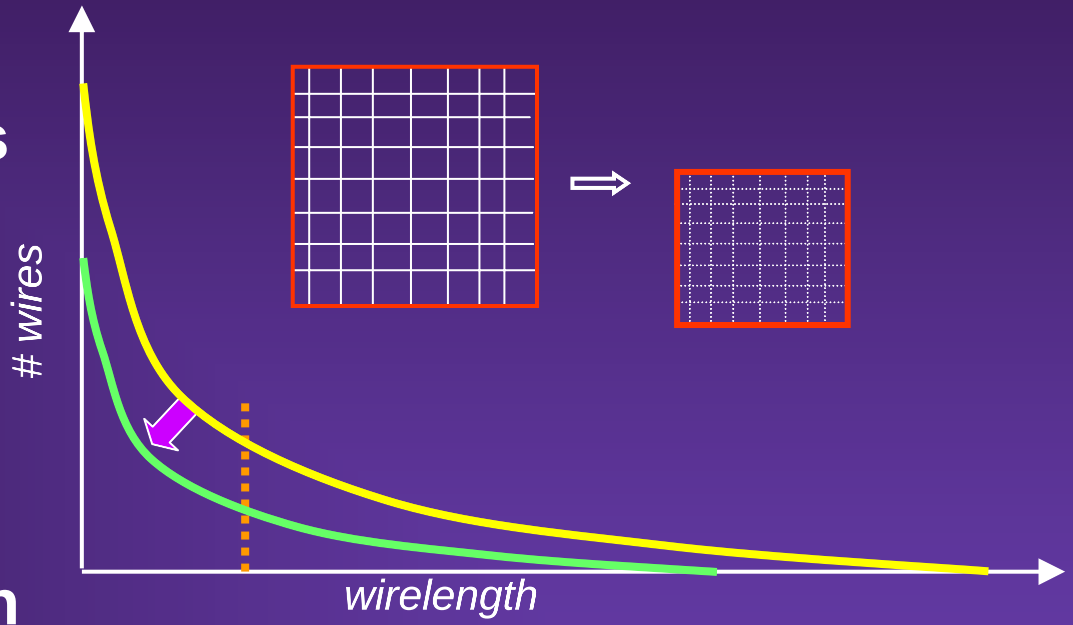
- **Ditto for normalized peak noise degradation**

- **Back-off length also scales at** $s^{1.5}$ **(not** s**)**
- **One time back-off … already taken**

Block Area Reduction

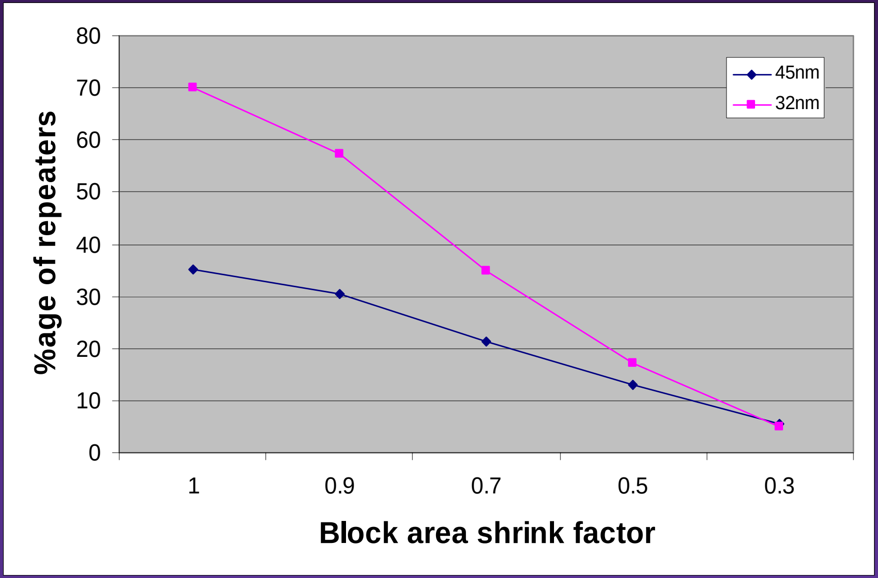
- Fewer long wires that require buffers

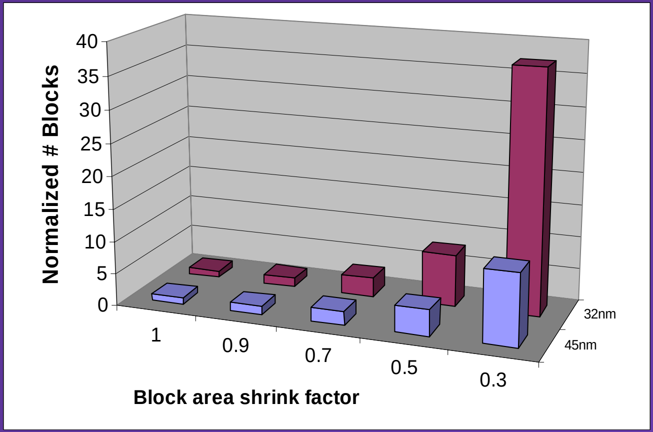
- Larger block count
  OR
  reduced integration



Block Area Reduction

- Reduced chip logic area => reduced functionality
  - Goes against history
- Smaller blocks require fewer buffers
- … but # blocks grows rapidly
  - Flat buffer %age requires block area to shrink to ~33%
  - 3x blocks per process node (for same chip logic area)





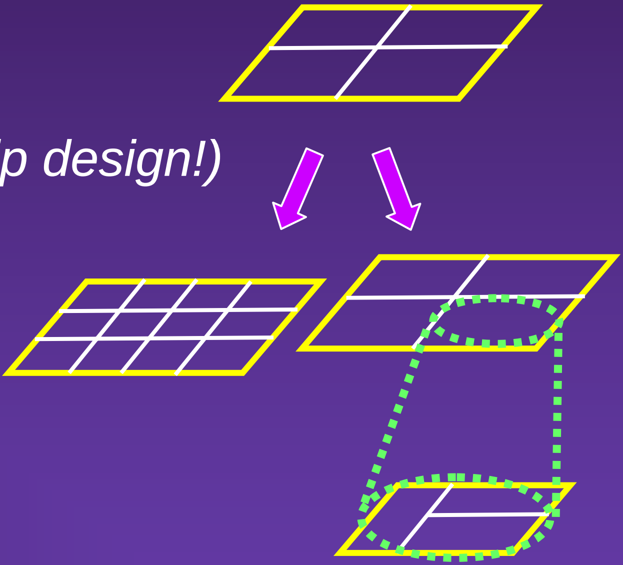
Full-chip Assembly with Shrunk Blocks

Shrunk blocks control block level buffer growth

BUT

# blocks triples!

(and block assembly is the hardest part of chip design!)



- **Flat assembly**

  (Fragmentation of paths across blocks)

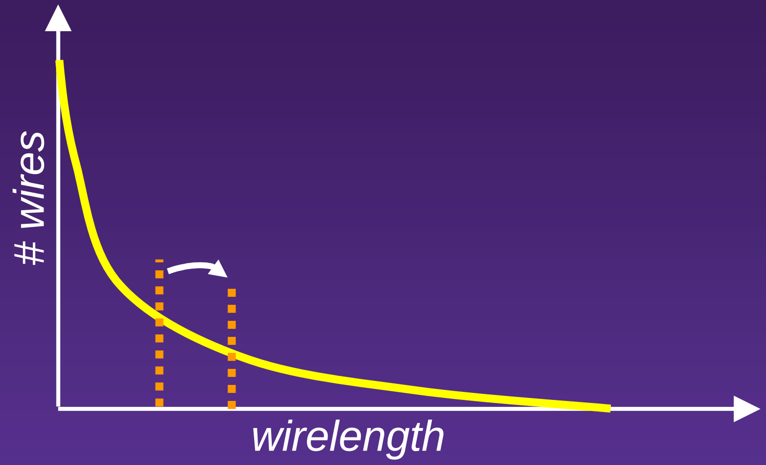
  OR

- **Increased hierarchy**

  (Lack of visibility across hierarchy levels)

Fat Wires

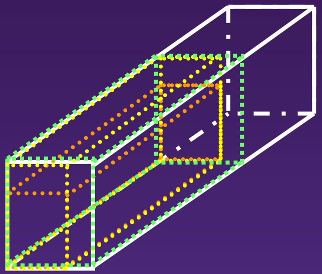
- **Recall** $l_{opt} = \sqrt{\frac{R_d C_g}{rc}}$
- **Fat wires: smaller *r* increase**
  - *rc* vs. *rc/s*^{2} **per** μ **in extreme case**
- **Increased inter-buffer separation**
  - **Balancing device and wire delay**

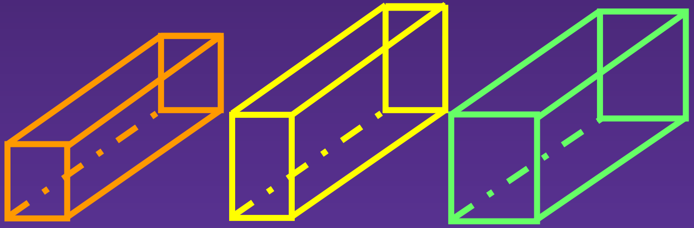


Fat Wires

- Tall wires : hard to manufacture, noise-prone
- Tall and wide wires

- Designs increasingly wire dominated
  - Wide wires increase block area
    - Area translates to cost, yield
  - Spread out cells cause longer wires
    - Increased delay

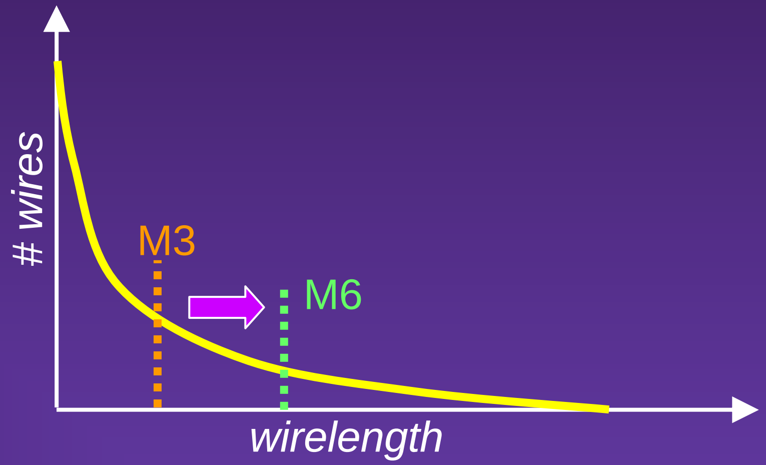




Up-layering

- Inter-buffer separation much larger on upper metal layers

- High demand for upper layers
  - Global clock, power grid
    - Voltage droop degradation
  - Critical global wires
- Significant routing congestion
  - Lower layers: via stacks
  - Added layers: diminishing benefits
- Power hungry
  - Increased wire capacitance, larger drivers



Slower Transistors

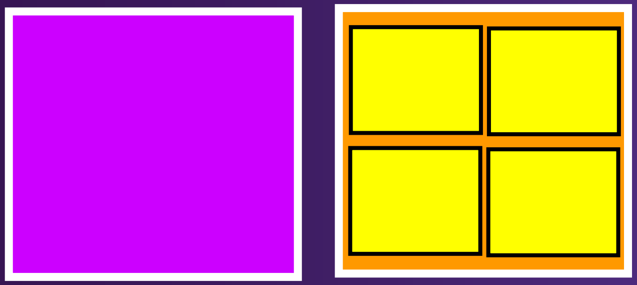
- Balancing device and wire delay
  - Slower driver => longer (slower) wire segment
- Designs often power limited
  - Elevated *Vt* : lower leakage
  - Lowered *Vdd* : lower dynamic power
- Reduced price premium for raw *GHz* in many high-end (*μP*) designs
  - Functionality, concurrency, power

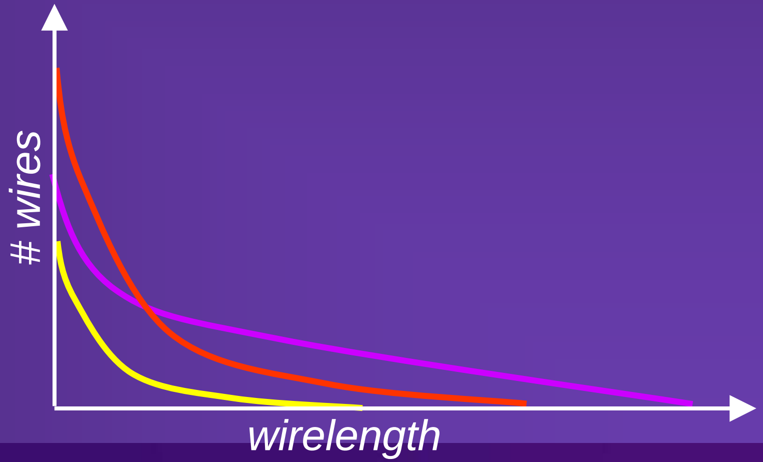
Slower Transistors

- Faster designs: still a competitive advantage
- Buffer reduction: little gain for high cost
  - Recall $l_{opt} = \sqrt{\frac{R_d C_g}{rc}}$
  - Tyranny of square root: $R_d C_g$ (device τ)
  - e.g., 10% slowdown => 4.8% inter-buffer length increase

Architectural Options

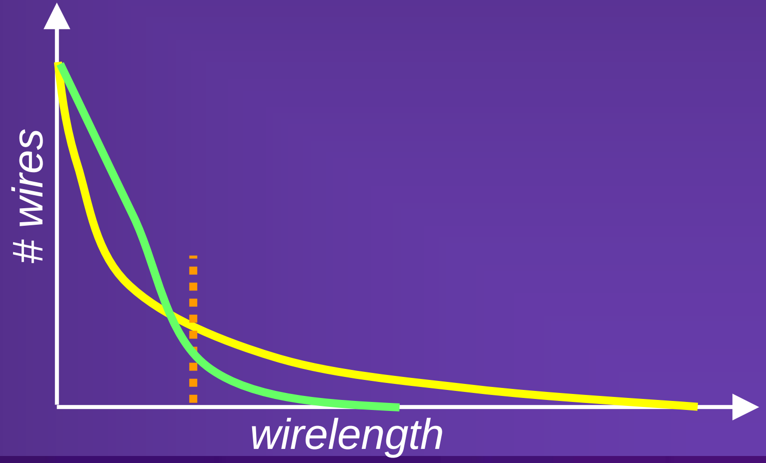
- Alternate scaling scenarios also face interconnect tyranny (albeit to differing degrees)
- Most promising approach: simplify interconnection complexity architecturally
  - Modify wiring histogram shape (i.e. Rent's parameters)
- An example: multi-core microprocessors
  - Goes against traditional approach of increased integration through block size scaling
  - Some performance and throughput overhead for increased concurrency





Integration Technology Options

- **3-D integration** (Banerjee'01, Deng'01, Das'03, Black'04)
  - **Multiple layers of active devices**
- **Changes wiring distribution shape by eliminating many long wires**
- **Manufacturing technology promising but immature**
  - **Thermal Dissipation**
  - **Manufacturability**
  - **Parameterized Yield**
  - **Testability**



Summing Up

- Straightforward projections of historical trends yield an infeasible design point
- … but alternative scaling scenarios are not encouraging either

- Architectural approaches are most promising
  - Modify the wiring distribution shape
- 3-D integration is another promising option

CAD Implications

- *Discussed extensively in ISPD'03 paper*

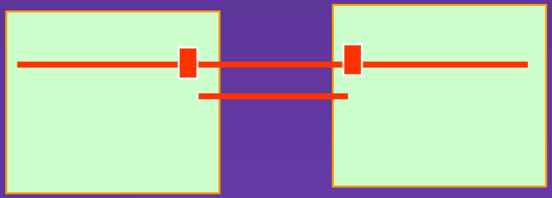
- Sequential Optimization
  - Post-RTL latency optimization
  - Optimization across sequential boundaries
- Synthesis
  - Misleading fanout metrics (along with literal/gate count and logic depth metrics)
  - Dense encodings and logic replication
- Layout
  - Buffer prediction and allocation during placement
  - Route-dependent on-the-fly buffer handling

Wire Pipelining Mis-prediction

- Cycle latency of nets in μarch. spec. depends on floorplan
  - Downstream implementation must guarantee specified interconnect cycle latencies
- Hard to change interconnect latency downstream
  - Arch. perf. simulations, formal verification proofs, validation test vectors are invalidated
  - So, μarchitects often pad cycle latency estimates
    - Hard to predict bus cycle latency since blocks not yet implemented

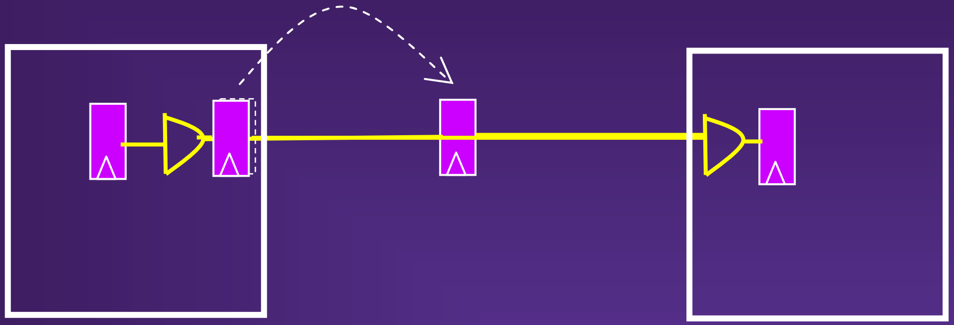
      => pin positions unknown, block areas can grow

With frequency roadmap slowdown, mispredicted interconnect latency problem not as urgent

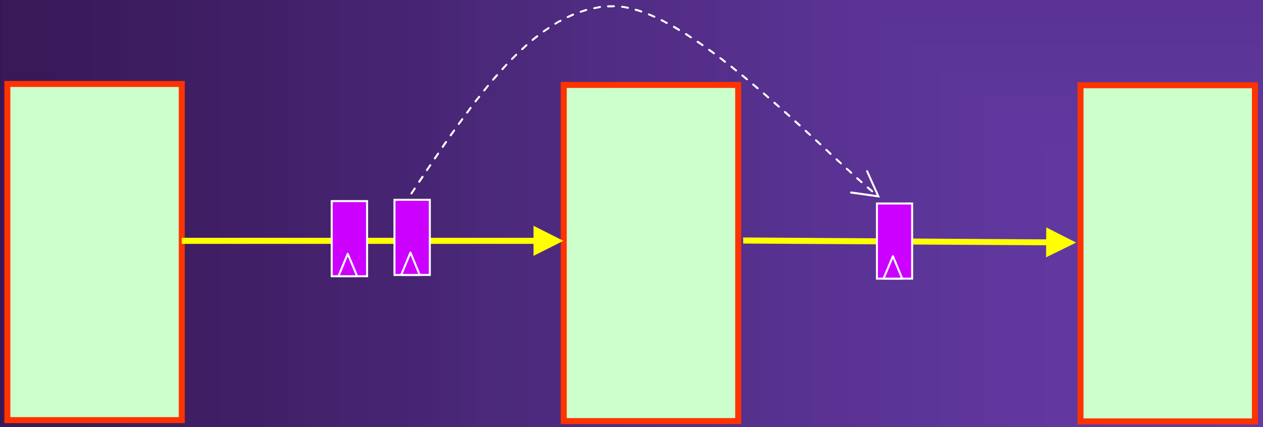


Retiming with wire pipelining

- Move flops out from logic blocks onto wires

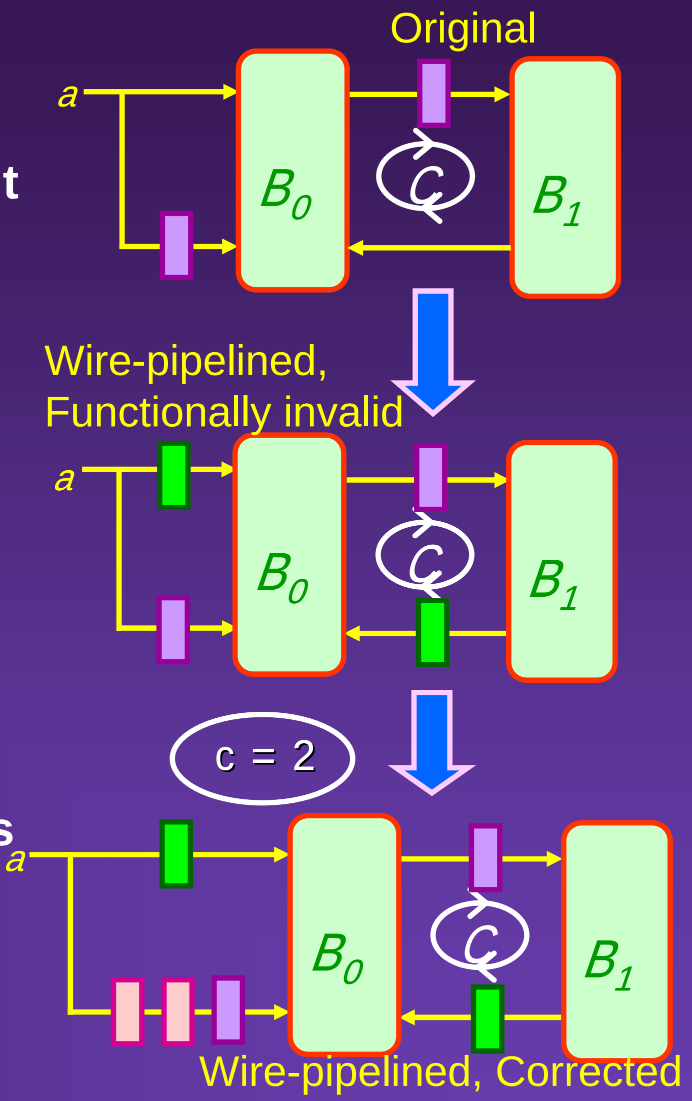


- Move clocked repeaters across blocks



Beyond Retiming

- Retiming: not always possible or sufficient
  - Logic blocks may not have enough sequentials with sufficient slack
  - Blocks may be black-boxes (e.g. 3rd party IP cores)
- Non-retimed wire pipelining
  - Functionality needs to be restored
  - System throughput decreases
  - *c-slow transform*: valid data at PIs/POs once every *c* cycles
- Behavioral equivalence (instead of c-slow cycle-equivalence) – open problem!

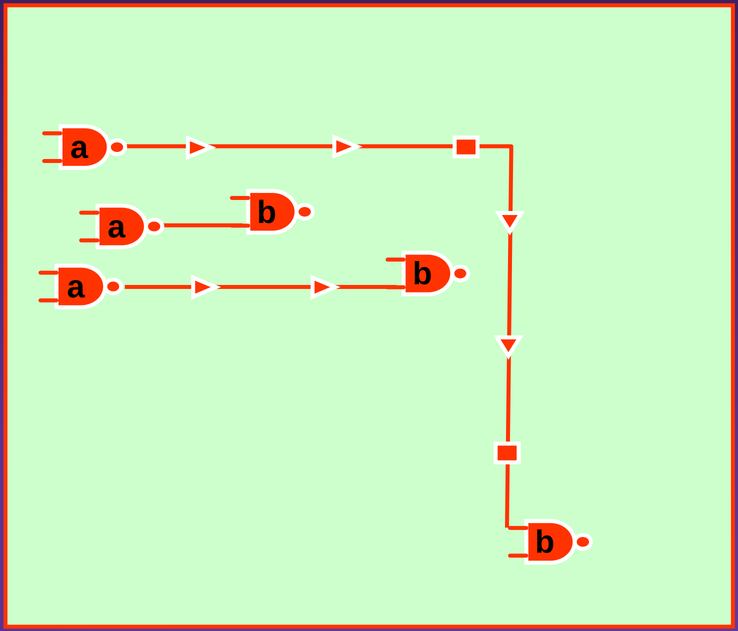


Source: Nookala, DAC'04

Buffering and Placement

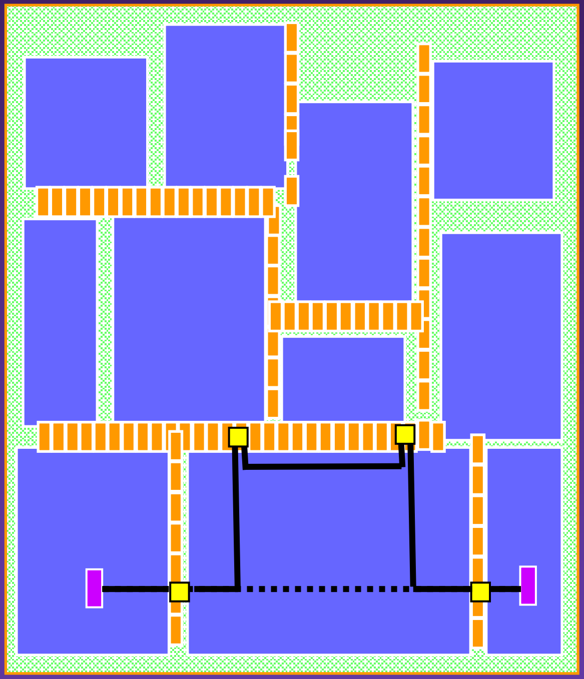
- **Buffering needs of a net : route-dependent**
- **Plan for expected buffers during placement**

- **Buffer blocks in channels** (Cong'99, Sarkar'00)
- **Fine-grained buffer allocation**
  - **White space management** (Brenner'03, Yang'03, Li'04)
  - **Explicit buffer modeling within placer** (Saxena'04)



Buffer Banks

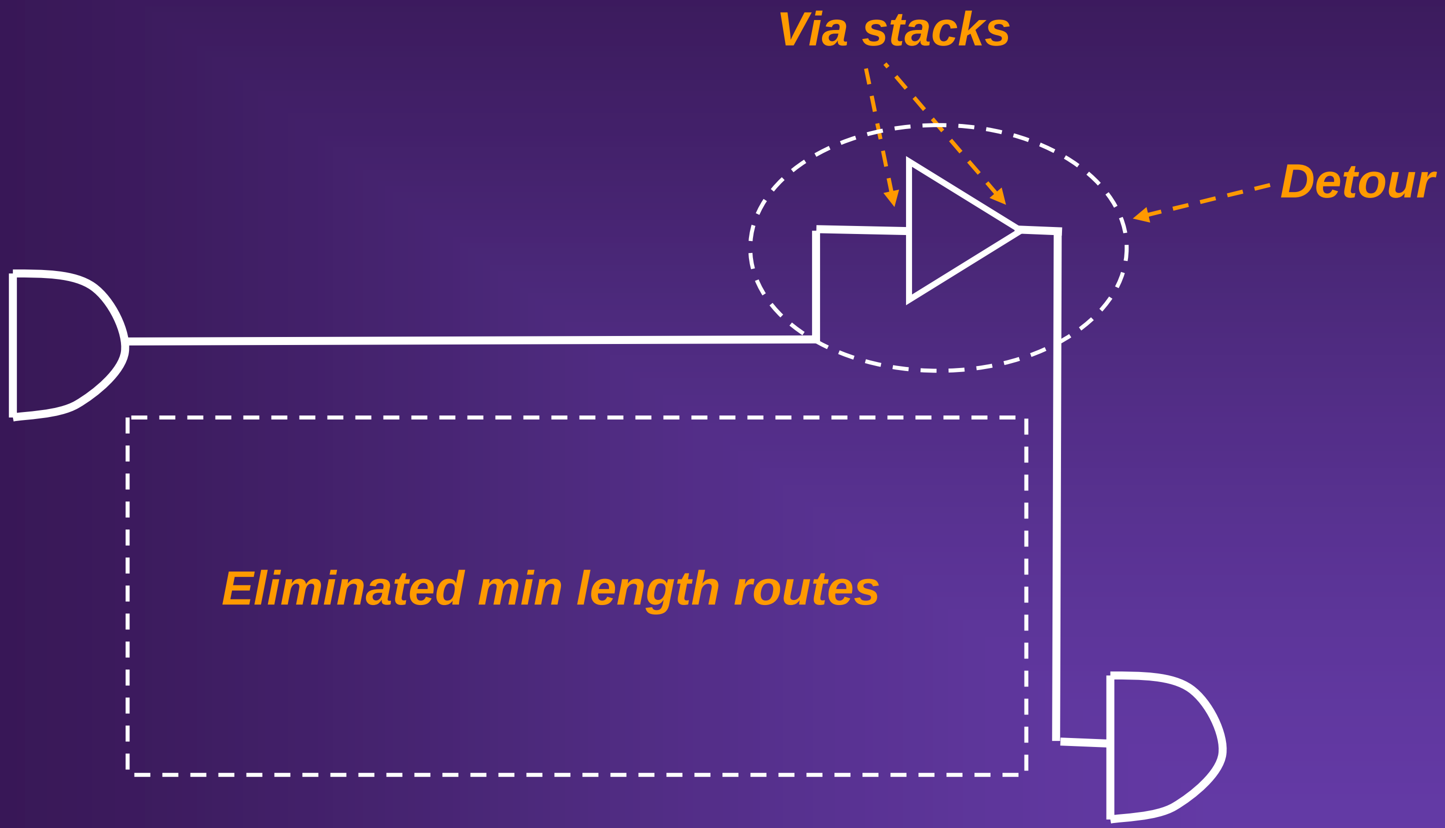
- As inter-buffer separations shrink, the detour to reach a bank can become significant
- Often become thermal and IR-drop hot spots
- Net fragmentation by buffers causes poor layout
- *Sometimes unavoidable* (black boxes, IP cores, etc)



Fine-grained Buffer Allocation

- ***Congestion-aware white space management***
  - **Introduce white space in cell-congested areas**
  - **Creates space implicitly for buffers**
  - **May have problems with dense designs**
    - **Implicitly proxies buffer density by routing congestion**
- ***Explicit buffer modeling within placer***
  - **Reserve space for buffer close to its expected location**
    - **Force model that captures buffer semantics**
  - **Dynamically create and delete “virtual” buffers**
  - **Promising approach for dense designs**

Routing Congestion due to Buffers



Routing Congestion due to Buffers

- **Interconnect synthesis that comprehends congestion** (Alpert'04)
- **Router that comprehends (simplistic) buffer insertion**
  - **Where is a buffer needed on a net? When is it redundant?**
  - **Where can it be placed?**
  - **Net ripup-and-reroute that can move buffers also**
- **Post-routing cleanup of poorly buffered nets** (Lembach'05)
  - **Insertion, deletion and/or relocation of buffers**

Buffered Interconnect Synthesis

- ***Environmentally-aware interconnect synthesis*** (Alpert'04)
- **Environmental cost for routing and buffer congestion**
- **Basic framework:**
  - **Fast, congestion-oblivious performance-driven buffered Steiner tree heuristic**
  - **Congestion-aware relocation of Steiner points**
  - **Resource-constrained van Ginneken variant for re-buffering (and sizing) of final topology**
- **Generalized cost for all nets**
  - **Non-critical nets: environmental cost**
  - **Critical nets: delay**
  - **All nets: max slew constraints**

Delay Modeling of Buffered Nets

- **Net delay models to capture effect of buffering** (Alpert'04)
- **Predict the eventual optimized delay of a net**
  - **Linearized delay for critical nets**
  - **Quadratic delay for non-critical nets and over large macros**
  - **Fitted linear delays for medium sized macros**

- **Applicable at early stages of design planning and implementation**

Concluding Remarks

- Rapid buffer growth leads to infeasible design in most scaling scenarios
- Most promising approach: change wiring distribution shape
  - Architectural choices
  - 3-D integration
- Methodological and algorithmic impact at each stage of design
- Several recent works hold promise