

Interconnect Scaling Implications for CAD

Ron Ho, Ken Mai, Hema Kapadia, Mark Horowitz

Stanford University, Computer Systems Laboratory, Stanford, CA 94305

{ronho, demon, hema, horowitz}@vlsi.stanford.edu

1.0 Abstract

Interconnect scaling to deep submicron processes presents many challenges to today's CAD flows. A recent analysis by Sylvester and Keutzer examined the behavior of average length wires under scaling, and controversially concluded that current CAD tools are adequate for future module-level designs. In our work, we show that average length wire scaling is sensitive to the technology assumptions, although the change in their behavior is small under all reasonable scaling assumptions. However, examining only average length wires is optimistic, since long wires are the ones that primarily cause CAD tool exceptions. In a module of fixed complexity, under both optimistic and pessimistic scaling assumptions, the number of long wires will increase slowly with scaling. More importantly, as the overall die capacity grows exponentially, the number of modules and thus the total number of wires in a design, will also increase exponentially. Thus, if the design team size and per-designer workload is to remain relatively constant, future CAD tools will need to handle long wires much better than current tools to reduce the percentage of wires that require designer intervention.

2.0 Introduction

With process technologies capable of fabricating a billion transistor chip on the horizon, the CAD community faces many new challenges. Notably, interconnect scaling in deep submicron processes may force a fundamental change in current ASIC design methodologies. If interconnect delay becomes a significant fraction of total delay, timing convergence for standard-cell design blocks will become difficult or impossible to achieve with current, crude, fanout-based wire load models. Designers will need new tools and methods to synthesize large blocks in deep submicron technologies.

In a 1998 ICCAD tutorial, Sylvester and Keutzer carried out a detailed analysis of interconnect scaling and its potential effects on CAD methodologies [1][2]. By examining the scaling of average length wires, they concluded that CAD tools are adequate for future module-level designs. We examine the sensitivity of their analysis to a range of possible technology scaling assumptions by modeling their simulations with an RC tree delay model.

Since design speeds and timing convergence in synthesis flows are typically constrained by long wires, not average-length ones, we extend the analysis to long wires. We show that for a fixed complexity design, the number of long wires

grows slowly with scaling. If chip complexity remained constant, this increase in long wires could be handled by small improvements in today's CAD design flow. Unfortunately exponentially increasing die capacity exacerbates the increasing number of long wires per module by driving up the number of modules. Thus, with constant design team size, the number of gates per designer will grow exponentially. To prevent the per-designer workload from also growing exponentially the percentage of wires that need manual intervention must fall exponentially. This implies that future tools must handle a greater percentage of long wires without designer intervention. This is the key challenge that the CAD community must face.

3.0 Underlying problem in synthesis

Today's CAD methodology does not guarantee convergence to a final solution that meets timing constraints due to the discrepancy between wire-load estimates used by synthesis during logic optimizations and the actual wire-loads after layout. The initial synthesis step in this methodology decides the overall logic structure of the netlist using fanout-based wire-load models supplied by the standard-cell library vendor. This wire-load model is derived from statistical analyses of past designs in the library and represents the median of the wire-load distribution for each fanout. However, the post-layout wire capacitance has a Poisson distribution with a narrow peak around the statistical wire-load length and a long tail to the right. Figure 1 shows the discrepancy between post-layout and statistical wire-loads for a small design, where the nets are sorted by fanout and post-layout capacitance.

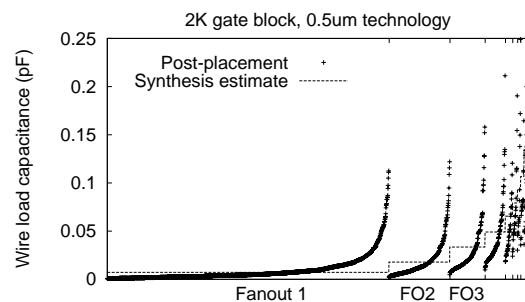


Fig. 1. Estimated and actual wire loads ([3])

Thus, even though synthesis estimates the wire-load of short and medium-length nets with reasonable accuracy, it highly underestimates the longer nets. For these long nets, synthesis would pick the wrong logic structure to drive such a

net and would not account for the intrinsic wire delay. These nets cause the appearance of new critical paths to be fixed by incremental optimizations using back-annotated wire-loads, but currently these optimizations are limited to gate sizing, buffer insertion, and critical path re-synthesis. Any changes need to be merged into the existing layout without perturbing the unchanged logic. If the layout is changed too much, the back-annotation used by incremental synthesis optimizations will be significantly different from actual wire-loads, possibly resulting in a new set of critical paths. Finally, long wires have intrinsic delay (from their self-resistance) so it might not be possible to meet critical path timing using this placement. Hence synthesis cannot guarantee that incremental optimizations will be able to meet timing constraints.

Arguably, CAD methodologies need only enable the common case of average wires, and the designer will manually handle the exceptions. This CAD model is tenable (and marketable) only if the number of exceptions remains at a reasonable level. To gain further insight into the future need for module-level CAD tool improvements, Sylvester and Keutzer examined the behavior of average length wires under scaling. In the next section, we review and extend their analysis.

4.0 Average length wires

Sylvester and Keutzer were motivated by popular forecasts claiming that as technologies scale, interconnect delays will increase and eventually dominate module-level timing [4][5]. At what size design, they asked, would traditional design flows using crude interconnect models be unable to converge on a design that would meet timing constraints?

To answer this question, they constructed strawman technologies for generations from $L=0.25\mu\text{m}$ to $L=0.05\mu\text{m}$ that were generally more conservative than those described in the SIA roadmap [4]. Using these technologies, they simulated a ring oscillator with and without average length wires between the gates to get intrinsic gate delay and interconnect delay. When adding wires, the gates were upsized until the incremental upsizing speed benefit was under 2%. Since their simulations showed that under scaling, interconnect delay fell relative to total delay, they concluded that with proper upsizing, interconnect scaling will not prevent current CAD flows from working for future 50K gate designs.

4.1 Reformulation of the analysis

Here we reformulate their simulation study (sections 7.1.2 and 7.1.3 of [1]) in an equivalent analytical approach. We model transistors by an effective resistance and measure delay by calculating the RC time constants [6]. Consider a fanout=2 ring oscillator of inverting gates. Without wires between the stages, the “intrinsic gate delay” of each stage is independent of device size and is shown in EQ 1. Next we add an average-length wire between stages and upsize the gates. The iterative upsizing algorithm used in [1] arrived at the same device sizes as well-known circuit design techniques that aim for effective fanouts to be around 4 for optimal performance [7]. Since the

line resistance of average length wires ($230\mu\text{m}$ long in a $0.25\mu\text{m}$ technology) is much less than the driver resistance, we will follow Sylvester and Keutzer’s lead and ignore the wire resistance. The delay with wires is shown in EQ 2¹.

$$D_{nowire} = R_{drv}(2C_{gate} + C_{diff}) \quad (\text{EQ 1})$$

$$D_{withwire} = R_{drv}(C_{wire} + 2C_{gate} + C_{diff}) \quad (\text{EQ 2})$$

Since the D_{nowire} (intrinsic gate delay) term is not dependent on device size, we assume that the devices are sized the same in both cases, so that the R_{drv} terms are equal. Then the ratio of interconnect delay to total delay can be written as

$$\text{Ratio} = \frac{C_{wire}}{C_{wire} + 2C_{gate} + C_{diff}} \quad (\text{EQ 3})$$

As we scale to future processes, we can apply their scaling heuristics [2], where S is the process scaling factor ($S=0.7$ per generation).

Using these scaling factors, the numerator of the delay ratio decreases faster than the denominator, and thus, the ratio of wire delay to total delay decreases. If we substitute appropriate numbers for the capacitances, we get ratios close to their simulation results (39% and 26% from simulation versus 38% and 25% from the model, for $0.25\mu\text{m}$ and $0.10\mu\text{m}$ technologies respectively). Since this RC model matches the simulation-based analysis of [1], we can use it to examine the sensitivity of the analysis to variations in scaling assumptions.

4.2 Revisiting the underlying assumptions

For the interconnect delay to total delay ratio to fall, wire capacitances must scale down faster than gate and diffusion capacitances. However, how wires, gates, and diffusions will actually scale is unclear. Sylvester and Keutzer scaled wire capacitance per unit length by 0.85 per generation for low- κ dielectrics, which is slower than the SIA roadmap but more optimistic than other projections [5]. They scaled gate capacitances down faster than S, since they projected that T_{ox} would decrease only slowly. Yet the use of high- κ oxide dielectrics would allow aggressive scaling of an effective T_{ox} [8][9][10]. Finally, although diffusion capacitances were scaled down slower than S, shallow trench isolations (STIs) or simply legged devices can cause C_{diff} to scale faster than S [11]. Silicon-on-insulator (SOI) devices could also reduce the impact of diffusion capacitance drastically.

Some scaling scenarios (*e.g.*, effective low- κ dielectrics, aggressive T_{ox} scaling, and no diffusion capacitance reduction) imply that wire capacitance falls faster than gate and diffusion capacitance and that the average interconnect to total delay ratio will decrease with scaling. However, other scenarios (*e.g.*, interconnect low- κ scaling at 0.625 -- just 5% slower than previously assumed, T_{ox} scaling as previously assumed, but STIs

1. The R_{drv} term includes a factor (close to $\ln 2$) that accounts for the fact that we are interested in the delay to the switching point of the next gate, which is taken to be $V_{dd}/2$.

and legged devices causing diffusion capacitance to scale faster at $S^2 * S^{-0.5} = 0.61$) imply that wire capacitance actually grows compared to gate and diffusion capacitance. In such cases, the interconnect to total delay ratio will *increase* with scaling. Thus the analysis is sensitive to the scaling assumptions used and can lead to opposite conclusions given reasonable but slightly different scaling assumptions.

The critical point to notice is that in all these scenarios, regardless of what scaling assumptions are used, the average interconnect to total delay ratio does not change that much, up or down, between successive technology generations. This result confirms Sylvester and Keutzer conclusion: current CAD tools will be able to handle average wires in future technologies. However, restricting the analysis to these short average wires can be optimistic, since timing problems can arise due to long wires and their appreciable resistances.

4.3 Average wire length determination

Sylvester and Keutzer derived average wire lengths using empirically fitted Rent parameters and Donath statistics [12]. One problem with using average wire lengths is that they are a very weak function of the total gate count. Using the Donath formulation for average wire length [12], Figure 2 shows average wire length as a function of gate count for various values of Rent's exponent. Sylvester and Keutzer examined a number of designs and determined that the average Rent's exponent was approximately $p=0.7$. The corresponding curve below shows that as gate count grows from 5K to 50K, a ten-fold increase, average wire length only increases by a factor of 1.66.

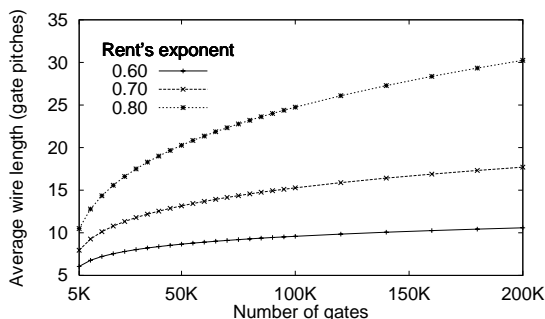


Fig. 2. Average wire length versus gate count

This weak relationship between average wire length and gate count arises since computation and communication in VLSI designs tends to be spatially local. Designers often construct large gate count blocks by accreting several smaller blocks together, thus keeping the average wire length fairly short. However, synthesis convergence is most vulnerable when unpredicted wire loads and delays add to critical paths, which occurs with long wires, not with average length ones.

5.0 Long wires

A “long” wire can be defined simply as one whose intrinsic delay is some fraction of a gate delay in a given technology. The gate delay we use is a fanout-of-four inverter (FO4) delay. A wire is considered long if its length exceeds L_{crit} :

$$L_{crit} = \sqrt{\frac{\alpha D_{FO4}}{R_w C_w}} \quad (\text{EQ 4})$$

R_w and C_w are per-unit length wire resistance and capacitance. In this paper, we will choose the fraction α as half of a FO4 delay, although the exact value is not central to the analysis. For a typical 0.25 μm process, a FO4 delay is about 90pS, and a minimum width copper M1/M2 wire has a capacitance of about 0.15fF/ μm and a resistance of about 150m Ω / μm . Thus, the L_{crit} for this technology is about 1.3mm, which is certainly longer than the average length wire, but still a realizable length in modestly sized modules. For example, even with an optimistic area utilization of 100% and a cell pitch of 10 μm , a 50K gate block still has a semi-perimeter of more than 4mm. In Section 5.1, we will derive estimates of the number of wires that exceed L_{crit} for a given design size by examining the wire length distribution.

Long wires present two problems for today's CAD methodologies. First, long wires have large parasitic capacitances, so that driving gates can be undersized. Simple upsizing will match the driver to the wire, but at a cost of ramping up in previous gates and across other fanout paths, leading to timing divergence as described in Section 3.0. Second, no matter how large we size the driver, the intrinsic wire RC delay, unpredicted during synthesis, remains constant. Various techniques, such as widening wires or relayout, mitigate the effects of long wires but are not part of today's standard CAD flow.

Under scaling, the D_{FO4} scales by S , R_w scales by approximately $1/S^2$ (assuming nearly constant aspect ratio), and C_w scales by between 1 and S (depending on the use of low- κ dielectrics). Thus, L_{crit} scales by between S and $S^{1.5}$, meaning that with scaling, more and more wires in a constant complexity design will exceed L_{crit} . The scaling of L_{crit} in gate pitches is shown below for both SIA and Sylvester/Keutzer projections, assuming that gate pitches scale with process (starting at 10 μm in the 0.25 μm technology). Note that L_{crit} scales down slowly, only changing by 1.6 between 5 technology generations.

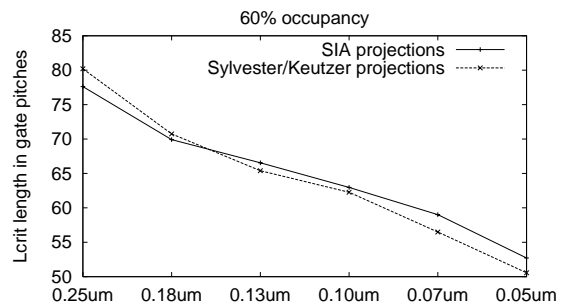


Fig. 3. L_{crit} under scaling

Using different scaling factors can lead to different slopes of the above curve, but all reasonable scaling assumptions lead to some decrease in L_{crit} with scaling. Thus, the number of wires that exceed L_{crit} will increase with scaling. To give us an idea of the number of nets that will require manual intervention by the designer(s), the next section examines the tail of the

wire length distribution to determine the percentage of wires exceeding L_{crit} ("long" wires) in a design of a given gate count.

5.1 Wire length distributions

Davis *et al.* constructed a model for wire length distributions, given Rent parameters and gate count [13], and showed it was a good fit for many designs. We applied their model to four specifically synthesized blocks: three units from the M-Machine, a fine-grained multicomputer designed at MIT and Stanford [14], and the global placement of Magic, a synthesized controller chip from Stanford's Flash multiprocessor [15] (minus the artificially long hand-routed MiscBus). From Figures 4 and 5 (we only show one M-Machine plot for brevity), we see that the model is a good fit for the wire length distributions of these designs, which span a wide range of gate count. The outliers in the M-Machine data are from long buses.

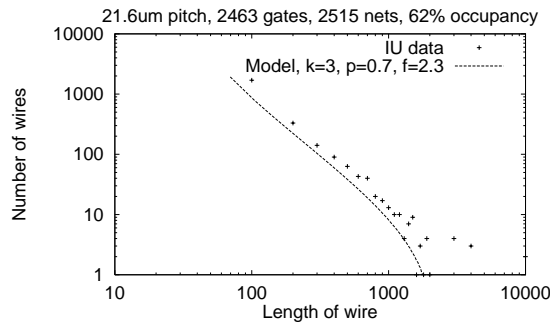


Fig. 4. M-Machine IUctrl unit

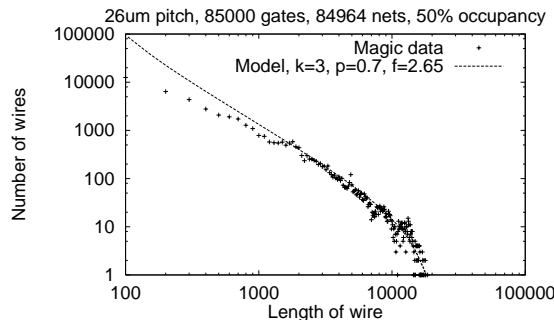


Fig. 5. Flash MAGIC node controller

The percentage of wires that exceed L_{crit} for a variety of gate design sizes and process technologies is shown below.

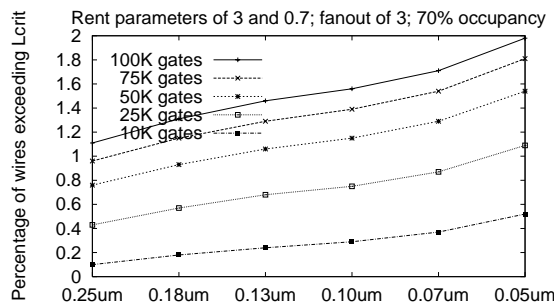


Fig. 6. Percentage of wires that exceed L_{crit}

From the above analysis we can conclude that we are not heading for a technology node in which there will be a catastrophic failure in module-level CAD tools such that the timing of most nodes in the system will deviate significantly from the pre-synthesis estimate. However, simply because module-level tools will not perform significantly worse in the future does not mean that there will be no need for improvement. Specifically the exponentially increasing design complexity will force a need for an improvement in the module level tools.

Scaling has two effects on the design -- it increases the percentage of wires that exceed L_{crit} , and it increases the total number of wires in the design. Scaling from $0.25\mu\text{m}$ to $0.05\mu\text{m}$ doubles the percentage of wires greater than L_{crit} in each 50K block, but also increases the number of blocks by 25. To keep the design time reasonably constant, the percentage of exceptions must fall by at least 25. Given these numbers, arguing about exactly how the number of wires that exceed L_{crit} scales for a fixed sized block misses the real problem. The exponentially increasing design complexity causes an exponential growth in the number of blocks. So even if the number of exceptional wires in a block is constant under scaling, the number of exceptional wires in the total design is growing rapidly. The tools must be much better at solving wire problems to keep designer productivity growing.

5.2 CAD Implications

As the design is scaled to the next technology generation, the capacity of a same-sized die doubles, and thus the complexity and gate count of the design has grown. If the designer now owns twice as many of these 50K gate blocks, then either she has to manually intervene for twice as many wires, or the CAD flow must now be able to handle wires longer than L_{crit} . The cumulative wire distribution function from Davis [13] gives the longest wire (in gate pitches) as the percentage of wires varies, as shown below. Here, area utilization is assumed to be 60%.

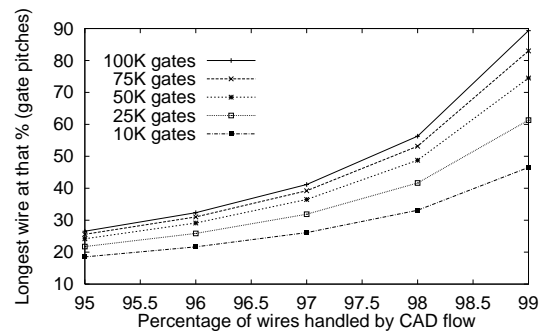


Fig. 7. Wire distribution with percentage of wires

The slope of this function tells us what extra wire length the CAD flow must handle to reduce the designer's workload appropriately. For the example in Figure 7, with a 50K gate design for which 0.8% of the wires are manually handled, the additional wire length the CAD flow must handle for a 0.4% gain in percentage of wires is about 32 gate pitches. In a $0.25\mu\text{m}$ technology with a $10\mu\text{m}$ gate pitch and a 60% area uti-

lization, this additional length is around 510 μm (added onto the L_{crit} of 1.3mm), an increase of nearly 40% in length and a factor of 2 in wire delay.

From these results we can see that as processes scale and designs grow in complexity, designer workloads for manually fixing long nets will increase. If the CAD tools do not improve, the design time and cost will scale up rapidly. If the tools do improve to handle even a slightly larger percentage of nets, the wire length that they must handle grows rapidly, and the currently ignored wire R will have to be taken into account.

This need for fewer exceptional wires is the core reason that researchers are exploring new methodologies that use variable width routing, metal promotion, synthesis-driven layout techniques [16], layout-driven synthesis techniques [3], post-layout resynthesis optimizations [17][18], and combining layout and synthesis [19].

6.0 Conclusions

From the above analysis, we can see that the performance of today's CAD tools on module-level blocks will not be significantly worse in the future. However, as technology scales, this level of performance will no longer be good enough. Assuming that the design team size remains constant, the number of modules that an individual designer is responsible for grows exponentially. Given that the number of CAD tool exceptions that a designer can manually handle is fixed, the number of exceptions per module must decrease exponentially. Thus, the CAD tools will have to improve to handle longer and longer wires, in order to decrease the percentage of exceptions, so that the number of exceptions per designer remains constant. Fundamentally, it is not that the CAD tools will perform much worse in the future, but rather that the pressure of ever improving process technology will require them to perform much better.

7.0 Acknowledgments

We gratefully acknowledge M-Machine data from Andrew Chang and constructive discussions with Dennis Sylvester. This work was supported by DARPA contract number MDA904-98-C-A933.

8.0 References

- [1] D. Sylvester and K. Keutzer, "Getting to the Bottom of Deep Submicron," ICCAD 1998.
- [2] D. Sylvester and K. Keutzer, "Getting to the Bottom of Deep Submicron (presentation slides)," ICCAD 1998.
- [3] H. Kapadia and M. Horowitz, "Using Partitioning to Help Convergence in the Standard-cell Design Automation Methodology," DAC 1999.
- [4] SIA, "National Technology Roadmap for Semiconductors," 1997.
- [5] M. Bohr, "Interconnect Scaling - The Real Limiter to High Performance ULSI," IEDM 1995, pp. 241-4.
- [6] M. Horowitz, "Timing models for MOS circuits," Ph.D. Thesis, Stanford University, 1984.
- [7] I. Sutherland *et al.*, "Logical Effort: Designing Fast CMOS Circuits," Morgan Kaufmann, January 1999.
- [8] T. Sorsch, *et al.*, "Ultra-thin, 1.0-3.0nm, Gate Oxides for High Performance sub-100nm Technology," 1998 Symp. on VLSI Tech., pp. 215-6
- [9] I.C. Kizilyalli *et al.*, "Stacked Gate Dielectrics with TaO for Future CMOS Technologies," 1998 Symp. on VLSI Tech., pp. 216-7.
- [10] M. Khare *et al.*, "Highly Robust Ultra-Thin Gate Dielectric for Giga Scale Technology," 1998 Symp. VLSI Tech., pp. 218-9.
- [11] T. Lin, *et al.*, "A Fully Planarized 6-Level-Metal CMOS Technology for 0.25-0.18 Micron Foundry Manufacturing," IEDM 1997, pp. 851-4.
- [12] W. Donath, "Placement and average interconnections lengths of computer logic," IEEE Trans. Cir. Syst., CAS-26, April 1979, pp. 272-277.
- [13] J. Davis *et al.*, "A Stochastic Wire-Length Distribution for Gigascale Integration (GSI) -- Part I: Derivation and Validation," IEEE Trans. Electron Devices, vol. 45, No. 3, March 1998, pp. 580-9.
- [14] S. Keckler *et al.*, "The MIT Multi-ALU Processor," HotChipsIX Digest, August 1997, pp. 1-7.
- [15] J. Kuskin *et al.*, "The Stanford FLASH Multiprocessor," Proc. 21st Intl. Symp. on Computer Architecture, April 1994, pp. 302-13.
- [16] W. Gosti, *et al.*, "Wireplanning in logic synthesis," ICCAD 1998, pp. 26.
- [17] G. Stenz *et al.*, "Timing driven placement in interaction with netlist transformations," ISPD 1997, pp. 36-41.
- [18] M. Lee *et al.*, "Incremental timing optimization for physical design by interacting logic restructuring and layout," Proc. ACM/IEEE International Workshop on Logic Synthesis, May 1998, pp. 508-13
- [19] A. Salek *et al.*, "A DSM design flow: putting floorplanning, technology mapping, and gate placement together," DAC 1998, pp. 287-90.