Design Automation for Deepsubmicron: present and future

Ralph H.J.M. Otten, Raul Camposano, Patrick R. Groeneveld[0]
Eindhoven University of Technology, Eindhoven, The Netherlands;
Synopsys Inc, Mountain View(CA), U.S.A.;
Magma Design Automation, Cupertino(CA), U.S.A.

Abstract

Advancing technology drives design technology and thus design automation (EDA). How to model interconnect, how to handle degradation of signal integrity and increasing power density are changing now, and have led to integrating logic and layout synthesis. Agressive gate sizing to control timing has become part of any modern back-end. From 0.13μ and down, chips will be more susceptive to breakdown during fabrication (antenna effect) or to wear out over time (electromigration) and dealing with these issues will require careful planning.

More integration of fast and accurate analysis with a complete design flow (chip planning, synthesis, placement and routing) will be needed, and still, advancing complexity will affect design and verification. Using hundreds of millions of devices effectively will be possible only by reusing pre-designed intellectual property (IP) effectively and by addressing system-level issues in EDA.

In the long term only more radical changes will keep us on Moore's track, changes that ultimately will have us depart from the two$^+$-dimensional confinement and lead to multiple active layers, and changes that will affect deeply the face of EDA altogether.

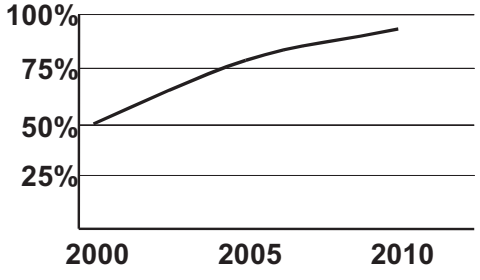

Figure 1. IP reuse as a percentage

[0] otten@ics.ele.tue.nl, raul@synopsis.com, patrick@Magma-DA.com

1. Introduction

The organization of this session is as follows. We will start with an analysis of EDA today from an industrial point of view. It will expose where the industry is now, and where it has to go in the coming years. Then we will discuss in detail how problems of modern technology are tackled in design automation. We will conclude with analyzing the trends in the longer term and propose a number of ways of how to stay "on the road".

Please do understand that this paper is composed of three full size contributions that, due to page number limitations, had to be reduced. Consequently, details are not included, but will be presented at the conference. Issues not touched upon in this paper, such handling cross talk and multiple threshold voltages will be discussed then. With a considerable number of (self)references we have tried to make this paper a useful document to accompany the presentation.

2. Technology Drivers for EDA

Electronic Design Automation (EDA) is one of the key enablers of the semiconductor industry [12]. No chip is designed without EDA. Conversely, semiconductors drive EDA technology [5]. Technology drives EDA in many ways. Synthesis, placement and routing are enabled by multiple technology constraints: a logic library, usually in the form of standard cells of the same height and similar size which simplifies placement and routing, decouples technology from logic and enables synthesis. Furthermore, until recently interconnect was assumed to have negligible delay and digital circuits were modeled with "lumped" components so that field effects such as cross talk could not be analyzed. Power consumption issues were limited to the power grid. All that has changed. Chips today typically are designed using large pre-designed blocks referred to as Intellectual Property (IP) as well as

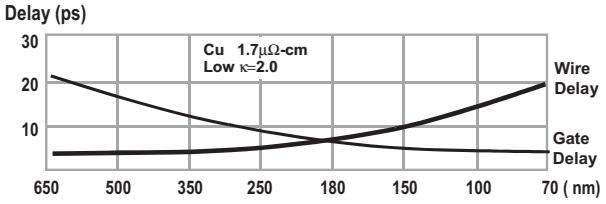

Figure 2. Wire delay versus gate delay

standard cells which may be up to six orders of magnitude smaller [21]. The reuse of IP is the most effective way of addressing complexity in design (figure 1). IP is increasingly becoming one of the main differentiators for system and chip design.

Interconnect can no longer be ignored, since for many designs under $0.25\mu m$ interconnect delay actually becomes larger than circuit delay [6]. Figure 2 shows the delay of a $43\mu m$ long wire in aggressive copper technology compared with the smallest gate delay. As a consequence, wire delays cannot be ignored in logic design any more, meaning that synthesis needs to have a good estimate for wire delays. The initial solution adopted long ago was the so-called "wire load model," which is a statistical model of de lay dependent on the design size. Current solutions do placement and synthesis (or resynthesis) together, or start with wire planning, so that wire lengths can be more accurately estimated from early on in the design process.

Moving forward, cross talk can no longer be ignored in logic design for technology nodes below $0.18\mu m$. Cross talk can be measured as additional delay before a signal stabilizes at the driven value. Worst-case cross talk due to capacitive coupling for a pair of $43\mu m$ long wires is illustrated in figure 3 and compared to interconnect delay in the absence of cross talk. As the technology node approaches $.10\mu m$, inductivity becomes noticeable for long wires at high speed such as busses of several mm of length operating at more than $1 - 2GHz$ [7]. This is already a problem for microprocessor design.

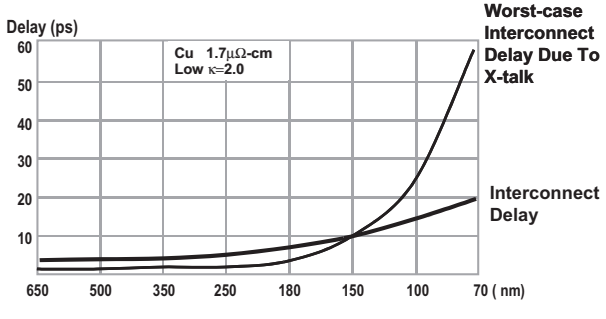

Figure 3. Interconnect delay due to cross talk

Computing cross talk requires knowledge of the exact position of the wires involved. Avoiding cross-talk can be achieved by several mechanisms: signals can be offset so that they switch at different times, wires can be placed further apart, driver strengths can be increased, additional buffers can be inserted, etc. Hence, we expect to see a much tighter integration of synthesis, placement and routing. Power dissipation of CMOS circuits so far has been well approximated by exclusively modeling the dynamic power. As we approach $0.10\mu m$, leakage power will become significant. Optimizing overall power consumption may be the foremost goal of the design. Figure 4 shows the power densities assuming typical technology parameters and an increase in switched capacitance from $600pF/mm^2$ to $2000pF/mm^2$

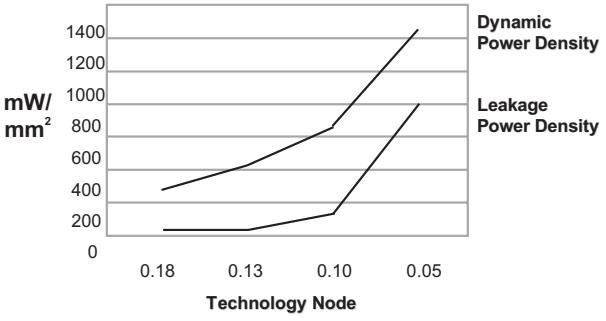

Figure 4. Power densities

Again, EDA needs to incorporate models for leakage and then minimize it by, for example, using multiple threshold-voltage libraries [2].

The effect of complexity on EDA can be illustrated by logic simulation. The simulator has to keep up not only with the increasing size of the designs to simulate but also with an exploding numbers of vectors to simulate. Simulation speed has increased by a factor of fifty compared with the thousand-fold increase of simulation need every 6 years (figure 5). In addition processor speeds have increased approximately by a factor of 10-12 during the same period.

Thus, functional simulation speed has increased by roughly 10*50=500 times over the last 6 years, approximately half the rate of the thousand-fold need. Further speedup can be achieved by hardware acceleration or emulation.

Furthermore, formal techniques have emerged as a powerful aids to simulation. As chips become more complex, they increasingly encompass larger and larger subsystems or complete electronic systems. The electronic design of these "Systems-on-Chip" (SoC) requires system knowledge. Conversely, designing systems becomes increasingly designing chips. System design has been automated much less than electronic design and tends to

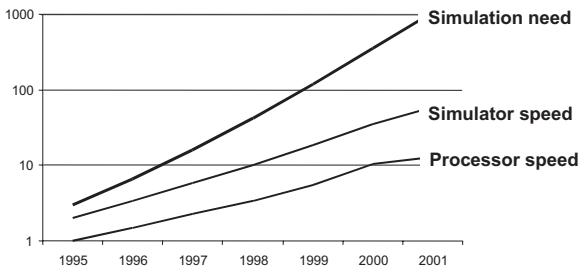

Figure 5. Normalized (w.r.t. 1995) simulation speeds

be much more domain specific than EDA. System design today is mostly IP-based. Processors enable the implementation of large parts of functionality in SW. The con cept of IP is further extended by defining "platforms" [10], which are basic architectures geared towards specific families of applications.

EDA continues to change to meet the needs of electronic design responding to technology and complexity complexity. The advent of Systems-on-Chip is introducing system design aspects into EDA. IP based design is eveolving towards "platforms". We expect these trends to continue as long as integrated circuit manufacturing keeps moving to smaller technology nodes.

3. State-of-the-art layout design

3.1. Aggressive Gate sizing

With reducing feature sizes, the electric properties of wires became more prominent, so prominent even, that a feasible layout implementation was not possible without extensive manual tweaking to fix timing problems. The longer wires are dictating the system timing. The actual length of a wire between gates is, however, only known after the circuit has been put through a number of physical design stages. In order to get to these stages a feasible gate-level net list must be available. A proper gate net list depends on the expected parasitics. This chicken-and-egg loop was generally fixed by iteration or manual intervention.

The source of this timing closure problem lies in the fact that a fixed netlist was presented to the placement and routing tools. Finding the physical locations for each of the gates inherently makes some wires longer, while others become shorter. The long wires will have a dominating effect on the timing, since the worst-case wire path on the entire chip determines the timing. Timing driven placers attempted to address this problem by giving timing critical nets a high priority, and with that, forcing a shorter wire length. Unfortunately, this priorization leads to unstable convergence: making some wires shorter will inevitably come at the expense of making others longer, and possibly some other unexpected net starting to dominate timing.

The first 'line of defence' in maintaining timing correctness is appropriate gate sizing. Ivan Sutherland [22] introduced the concept of effort that elegantly captures the first-order relationship between gate size, speed and delay. Larger gates have bigger transistors that can charge the capacitance of a wire quicker, typically making the path faster. During the placement process the wire length between a gate output and the inputs it drives can be estimated. This translates into an estimated wire capacitance that, on its turn, corresponds to a certain gate size. Adapting the gate size to the parasitic load can keep the path delay constant over a certain range. Meanwhile, gates that are are less timing-critical are sized down to save area and power consumtion.

There is a catch, however. Gates with a larger drive strength will also impose a larger parasitic load on their input pins. The larger internal transistor will have a larger active gate area. In conventional static CMOS circuits this relationship is approximately linear: doubling the drive strength of a gate will double the parasitic load of its input pins. This means that this gate will have to be driven by a gate with a larger drive strength. Sizing down a gate will have the opposite effect: a smaller drive strength will be needed to drive its input at a given speed.

The 'constant delay' physical synthesis technology built on that concept[1] has been inplemented in certain modern physical synthesis systems, e.g. [1]. The guiding concept in constant delay physical synthesis is to pick delays beforehand, and keep them fixed throughout the physical design steps. The latter is performed through aggressive gate sizing, buffer insertion and logic restructuring. The netlist at the input of such system is quite different from the one that actually ends up in copper wires and silicon gates on the chip.

Modern physical synthesis tools will size each gate up or down to match the speed requirements with the parasitic load of the wire. As a result, all gates are sized just 'right': not too large (which would impact power consumtion and the load upstream) and not too small (which would cause timing to fail). The theoretical basis and its elaboration can be found in [15]. The typical flow is summarized in figure 6. It is a combination of sizing the gates to satisfy timing requirements and to insert buffers

[1] Other names are *fixed delays, gain-based synthesis, and timing guarantee.* It was already presented in 1996 [19] following clues from the effort theory, and the observation that gate delay is constant when gate size scales with its capacitive load.

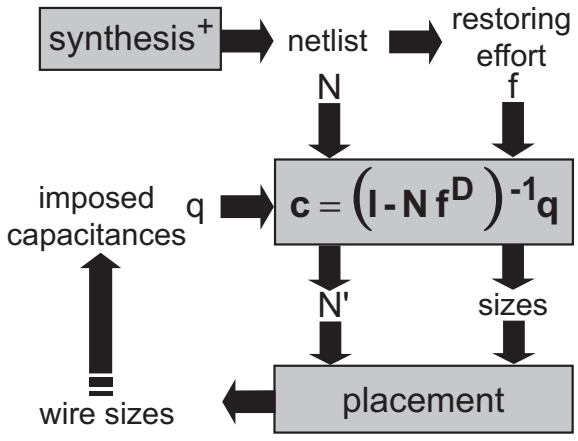

Figure 6. Sizing flow for timing closure. N is the netlist as incidence matrix, f is the vector of reciprocal efforts (c_{in}/c_{out}), q is the vector of wire capacitances, and N' is the modified netlist.

to recover area wherever it does not affect overall timing.

Agressive gate sizing results in a wider spread of gates that can be found on a chip: very small drive strength for the short local wires, and quite large drive strength to enable a signal to cross the chip with maximum speed. Having variety of drive strength on the chip, combined with the peculiarities of sub-0.13μ process technologies pose a number of new challenges for an automatic layout design flow. Challenges that did not exist in the conventional automatic placement and routing tools. The dominant ones will be reviewed in this paper.

3.2. Electromigration

The high-drive strength gates will drive significant currents into a wire. While conducting such current through the metal interconnect, the electrons will interact with the lattice imperfections. The electric current can be seen as an 'electron wind' that slowly moves the atoms in a particular direction.

The copper or aluminium on most chips are polycrystalline, that is, made up of small 'grains' of lattice. Atoms can be transported at the boundaries between these grains as a result of the electron wind. In the direction of the electron flow metal atoms will be deposited over time ('hillocks'), while in the opposite direction voids will grow between some grain boundaries. In time the voids will reduce the conductivity of the wire, which eventually could cause the wire to stop conducting altogether. The hillocks can introduce mechanical stress or start shorting with a neighboring wires.

If not properly dealt with during the physical design of a circuit, this electromigration phenomenon will cause the premature failure of the circuit. Most chips must have a MTBF (Mean Time Before Failure) of 10 years. Failure due to electromigration of a single wire is generally modelled through Black's equation[3]:

$$MTBF = \frac{A}{j^2 * exp(\frac{E_a}{kT})}$$

where A is a material constant, j the current density, E_a is the activation energy and T is the temperature. The first wire out of the millions of on-chip interconnects that fails due to electromigration will typically cause the entire chip to fail.

What parameters influence electromigration so that we can harness it during physical design? The first thing that should be noted is that the temperature appears in the exponent of the equation, and will therefore have a very strong effect on the MTBF. The temperature depends on the wire self-heat due to its resistance, the temperature of the surrouding wires and transistors and the heat conductivity of the packaging. At design time, these parameters are hard to predict and therefore we have to assume a worst case.

Good heat conductivity is critical in allowing high current densities. Conventional copper home wiring (e.g. $110V$) is designed for current densities in the order of $10^4 A/cm^2$. The wire would melt if the current density were much higher. The current density in on-chip copper wiring can reach $10^{10} A/cm^2$ without meltdown due to the much better heat conductivity of the silicon substrate. Unfortunately, the new low-K dielectrics that are used to reduce the parasitic wire capacitance are also poorer heat conductors than the dielectrics used in older process generations. This makes the effect of the Ohmic wire self-heat on electromigration more prominent.

Controlling DC electromigration by widening wires

Since we cannot control the temperature very well, controlling current density is the major parameter to harness electromigration during layout design. The wire height is fixed in most process technologies. Therefore using wider wires is the only way to significantly improve the MTBF. Wide wires are used in the power infrastructure of the chip. The maximum current through a (tungsten) via is generally less than the current through a metal wire of the same width. Routers are tuned to insert more than one via at a layer change for high-current wires.

Placing the additional via and widening the wires will reduce the maximum wiring density that can be achieved. A lower density typically increases the size of the chip and with that the unit cost of the chip. Therefore a trade-off between long-term reliability and short-term profits is an important engineering decision.

The metal line length also affects electromigration. Blech [4] showed experimentally that electromigration is only a problem if a wire is longer than a certain critical length (typically in the order of $50\mu m$). The mechanical stress in wires shorter than this critical length prevents electromigration breakdown.

Metal slotting

The wire width cannot be increased arbitrarily. The metal grain microstructure of the metal influences its susceptibility for electromigration failure. During processing of the chip, the interconnect metal wires are annealed such that the narrow wires have a 'bamboo' style grain micro structure. Such bamboo structure has a much better MTBF than general grain structures that resemble a jigsaw puzzle. Widening the wire will enventually result in a non-bamboo microstructure. To keep the structure bamboo style, most processes specify a maximum width rule for all metal wires. It is roughly in the order of 4 times the minimum width, which is not nearly wide enough for supply power wires.

To comply with this design rule, wide wires are 'slotted' by carving rectangular holes in the wires. Typically this is done for the power infrastructure on the chip. For best results, the slotting of larger wires should not occur as a postprocessing step. Designing a proper power structure with proper slotting is far from trivial. To avoid such design problems, many VLSI design teams revert to a fine-grain power mesh that is omnipresent.

Density rules: Metal Filling

Slotting wires effectively reduces the metal density on a particular layer. Controlling the local density in a particular layer is quite critical during IC processing. Chemical vapor deposition and chemical-mechanical polishing (CMP) produce results that are dependent on the local layout features. To make these effects uniform and predictable, the local density of the metal layout features should be uniform as well. These processing constrains are generally formulated as a minimum and maximum density of metal features (wires) within any arbitrary window on the chip. Typically the maximum metal density is 50% and the minimum density is 20% within any arbitrary 100 μm by 100 μm window on the chip.

In areas of dense routing this rule is typically complied with. Sparser areas will need to be filled with dummy features. Typically these are small pieces of loose wire. Placing these features poses no significant problem from a technical or an algorithmic point of view. An overview on techniques can be found in [9]. The dummy features are placed in the vacant spots of each metal layer. Maintaining a certain distance to the existing wires can ensure that the parasitic fringe capacitance with the new features does not significantly affect circuit timing. Practically, however, filling is postponed until the last moment since it dramatically increases the size of the mask data and makes any ECO almost impossible.

Other mask design rules

Many DSM manufacturing rules are encoded in the mask design rules (one might even say, hidden). Most vias have assymmetric overhangs that are often called 'end of line' rules. The additional metal overhang and the end of the line allows for some amount electromigration stress. It serves as a 'reservoir' near the sensitive tungsten-metal interface.

The feature sizes on the chip are smaller than the wavelength of the light that is used in production. As a result, the actual pattern on the chip deviates significantly from the orginial GDSII mask data file. To correct for this distortion during production, specific mask features are added. This process is called OPC (optical proximity correction), an automatic step that occurs by dedicated software in the mask shop. For VLSI design tools use the slightly higher lever of the mask design rules. Phase shifting mask is another technique to push the lithographical barrier. Currently the technique is used to reduce the transistor gate widths. The implications can be largely hidden in the internals of the standard cell. More information on OPC and PSM can be found in [20, 8].

AC electromigration

The previous model of electromigration applies to DC currents where the metal is subject to an electron wind from a constant direction. In a VLSI implementation, only the power supply wires carry DC currents. All signal wires are charged exacty the same amount of times as are they are discharged. In such case the reverse current has a 'healing effect'. As a result, the signal wire AC electromigration issue is negliable. Signal wires can still break down, but the exact cause of this failure mechanism has not been studied nearly as much as the DC electromigration. The phenomenon is called thermomigration.

Signal wire breakdown always occurs around the vias (contacts) as a result of thermal stress. The thermal stress is the result of the wire self heat due to the RMS current. The stress is reduced somewhat by requiring an 'end-of-line' overhang at the via as part of the mask design rules. The thermal self-heat can be controlled by increasing the width of the wire and adding additional vias.

3.3. Buffer insertion

As stated in the introduction, modern physical synthesis tools will instantiate a wide range of drive strengths to meet timing constaints. To cross $10mm$ on a 0.18μ chip, the optimial buffer distance is approximately $2mm$ and

the optimal driver size is in the order of 80x. A deviation from these optimal parameters will cause the signal to arrive later. Conventional standard cell libraries have gates with 1, 2, 4, 8 and at most 16x drive strength. This means that it will not be possible to cross the chip at maximum speed with such library. Having the larger drive strength cells available does significantly improve chip timing.

Such large drive strength cells will drive larger currents into the wires. As a result, wire self-heat (AC thermomigration) could affect the reliability of the chip. This can be handled through estimating the RMS current at the output of the high-drive strength cell. In case signal activity data is available, the effective duty cycle on the wire can be known. In other cases it can be assumed that the wire switches as the clock frequency. This results in a table of wire widths for the wires that are driven by such cells. In each layer the wire width can be different due to the different material properties. Also the amount and size of the vias are encoded is such 'wire code'. The first nets that are effected on the chip are typically the clock wires.

3.4. Antenna rules handling

Chips are usually processed 'from the bulk up', each time adding an additional layer of interconnect. Current processes have up to 8 layers of interconnect. While the metal interconnect chip is being assembled, the interconnect of a net will consist of a number of disconnected pieces of metal wire. Some of them connect to gate inputs, and one to a gate output.

The very small silicon gate length of a 0.13μ transistor gate makes it extremely sensitive to breakdown as a result of static charge. The half-assembled (and unconnected) wires at an intermediate stage of the metal processing will act as 'antennas' that pick up electrical charge. The longer this wire, the more charge can built up. Eventually the transistor gate input will break down. A diffusion contact such as a gate output or a special purpose diode can prevent the build-up of charge.

Antenna design rules are formulated in the design rule deck to address this issue. A typical antenna rule puts a maximum on the ratio between the metal wire length in a layer and the transistor gate area. For a typical 0.18μ process, the maximun length of an 'antenna' wire is in the order of $500\mu m$ (50 gate sizes). In a typical 0.13μ process the antenna length is dramatically smaller, sometimes even down to $20\mu m$ (a hand full of gate sizes). Without special precautions in the routing flow, many more wires will violate the antenna design rule.

The fact that advanced physical synthesis tools aggressively downsize gates makes the antenna problem even worse. Many gates are very small. The smaller the gate size, the smaller the transistor gate length, and the shorter the maximum antenna length.

The simplest way to deal with antenna problems is to protect each gate input by a special diode (a diffusion contact). The obvious drawback is the additional layout area that is necessary for the diodes. A typical design with 1000000 cells will need area for 30000000 diodes. Also, the diode adds an additional parasitic load to the gate input.

The second approach is to ensure that the router 'jumps' to the highest layer of the net near each gate input. This would avoid the size penalty of the diode approach. Typically the antenna problem was addressed as a postprocessing step on an existing routing. This worked for older 0.25μ and 0.18μ process technologies. In 0.13μ, the short antenna length results in a dramatic increase in the number of 'jumpers' that need to be added to the wire. Also, the additional interconnect layers make the inpact of the diode jumpers on the routing congestion more severe. The jumper insertion step is more likely to fail.

This fairly complex matter must be addressed in a similar way as the timing closure problem: in a number of steps that initially try to avoid the problem from occuring in the first place. At an early step in the design flow, diodes can be inserted at gate inputs that are known to be critical. Also, diodes should be inserted at all hierarchical module inputs. Inserting these diodes early avoids surprises such as additional path delay or a sudden area increase with congestion later on the design flow.

The next step is to avoid antenna problems from occuring through global routing. In the global routing step, longer nets will typically have multiple equivalent implementation options, each with a different path and layer assignment. Picking the most 'antenna friendly' one will avoid the majority of the future antenna problems. The global router is encouraged to generate routes that change layer up to a higher layer near gate inputs. In a later step, after power planning, again critical gate inputs are checked for their likelyhood of having a tough antenna problem. In case a problem is likely (such as when the cell is below a higher level power wire that would avoid the jumper) again a diode is inserted. Adding diodes at this stage is much easier than at the point where the entire placement is fixed. As a next step, a normal routing is performed with built-in jumper insertion for antenna avoidance. The few antenna violations that remain are again solved through ECO-style diode insertion.

4. Trends in chip design

Much of recent vlsi-theory assumes predictability. For example, the theory of global wires and optimal buffering is based on uniform wire capacitance and resistance. The

variation of environment over the length of a "long" wire, especially in the distance to the closest other wire segment in the same layer, causes reality to be far from such homogeneity. Already when these theories were derived [14] a drastic solution was proposed: put every signal wire between two wires on fixed voltages. It has been argued that this makes characteristics such as delay predictable, but also at their worst case value. This is not true! Due to miller-like effects the observed capacitance can be up to three times its static value. Besides the wires with fixed voltage, which constitute a mesh, do not have to have *design rule width* because of the high reliability of meshes in performing such function. Another argument against it was the fact that fifty percent of the wiring resources were wasted. This also is less severe if the fixed voltages are the useful power and ground voltages [11] which have to be brought almost everywhere on the chip. Well-known designs come very close to using fifty percent of the resource! In any case, cross talk has been effectively countered by the approach.

Another problem is the resistivity in wires. In fact, it defeats the *constant delay concept* of section 3.1. Of course one can decide to "live" with it [1] by buffering all "long" wires, and use the capacitance in the sizing algorithm. Another approach is to introduce a hierarchy in wires, separating them at the so called *critical length*. It is important to note that the delay of a section of that length, with buffers optimal for speed, is a process constant. Originally [14], global wires were defined as those of which the delay can be decreased by buffering; all others were considered local. Recently [7], local wires were defined to be bounded by approximately a quarter of that *critical length*. It has no impact on the main conclusion: modules of up to the order of 100,000 gates can be adequately treated by constant delay approaches. They can be handled by today's logic synthesis tools, and do not benefit from buffer insertion as far as speed is concerned. Their internal wiring is *local*.

Global wires are assumed to be optimally buffered, which implies wire delay linear in length. This led to *wire planning* [16]: the idea to fix wiring before the positioning of the modules. The delay due to global wires does not depend on module positions when this delay is linear in the length, provided that no detours are introduced. Only the pin positions determine the wire delay. Given the timing requirements the timing budget left for the modules can be determined. With delay-area trade-offs the division of those budgets over the individual modules can be determined under area minimization. Non-feasibility can be concluded when total area exceeds the available space.

Theoretically this approach also has problems, one of which is due to the fact that, optimally buffered lines have fixed input and output capacitances. However, the

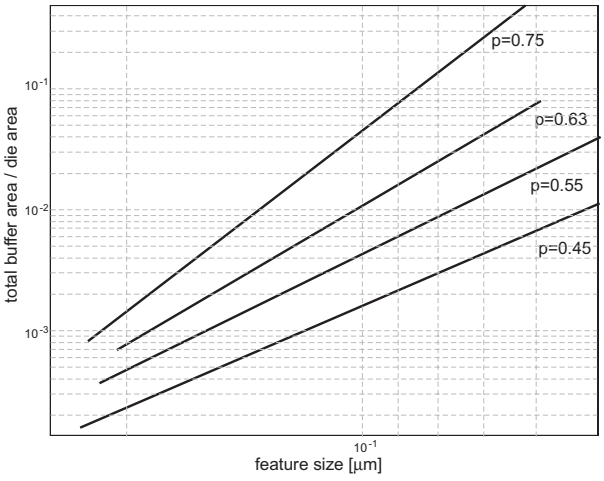

Figure 7. Buffer area as a function of feature size for several rent exponents

sensitivity to size and position is low close to optimum, and the "slack" area after budgetting for minimum area can be used for adaptations. There are also practical problems when every global wire (that is, wires longer than the critical length) are optimally buffered. Two of them get quickly worse when the feature size gets smaller: one is the total area taken by these buffers and the other is caused by variations in the threshold voltage. The former is illustrated in figure 7, which is obtained by deriving the wire length distribution from rent's rule, and integrating the area of all global wires (length $> \quad l_{crit}$), including their buffers [17].

The problems are greatly aleviated by an extra active layer on top of the wiring structure [18] (or full vertical integration or 3D chips). It not only provides for the extra area, but it allows for more robust V_t and avoids much of the routing blockage when these buffers have to be contacted in the bulk layer. In addition, more flexible adaptation to a footprint is enabled, and even clock skew can be reduced to negligible values by including optical receivers in that layer [13].

All these "fixes" do not address, in part or not at all, points such as the explosive growth in mask cost, the gigantic engineering effort in designing future systems on a chip, and the consequences for programming the hardware into a product. Reconfigurability and plain *master image* ideas force themselves into the foreground, but continueing the line of reasoning that produced much of the results of this section, lead us to the concept of homogeneous processing [18]. Using the observed and derived laws of chip industry (e.g. moore's law, rent's rule, feature size development) and the rules of well-balanced

processing systems[2], show that, beside all the phenomenae discussed above, memory will dominate future high performance computing systems to the extent that the computing resources will become negligible. So why not including all thinkable functionality in mask-identical tiles with seas of memory, and select the required specific functionality when finalizing the product. Although not feasible in the short term, we may well find that eight to ten years from now, information technology will have to resort to such styles in order to provide the demands in processing in an economically feasible way. Will that be the end of EDA?

5. Concluding remarks

Observing chip industry and EDA's response to its development, we analyzed its present state, both from an industry's point of view and from the tool builder's perspective. It shows that EDA's evolution over the last decennia has to accelerate dramatically to provide the required tools for the near future, and prepare itself for revolutionary changes in the long term, still less than a decade though!

References

[1] *Blast Fusion white paper*. Magma Design Automation, May 1999.

[2] L. Benini and G. DeMicheli. *Dynamic Power Management, Design Techniques for CAD Tools*. Kluwer Academic Publishers, Boston, Ma, U.S.A., 1998.

[3] J. Black. Mass transport of aluminium by momentum exchange with conducting electrons. *Proceedings 6th annual Reliability Physics Symposium*, pages 148–153, 1967.

[4] I. Blech. Electromigration in thin aluminum films on titanium nitride. *Journal of Applied Physics*, 19:1203, 1976.

[5] R. Camposano and D. MacMillen. Design technology for systems on a chip. *Proceedings VLSI-SoC, Montpellier, France*, pages 3–7, December 2001.

[6] J. Cong. An interconnect-centric design flow for nanometer technologies. *Proceedings of the IEEE*, 89(4):505–528, April 2001.

[7] A. Deutsch, P. Coteus, G. Kopcsay, H. Smith, C. Surovic, B. Krauter, D. Edelstein, and P. Restle. On-chip wiring design challenges for gigahertz operation. *Proceedings of the IEEE*, 89(4):529–555, April 2001.

[8] A. B. Kahng and Y. Pati. Subwavelength lithography and its potential impact on design and eda. *Proceedings 36th Design Automation Conference, New Orleans(LA), U.S.A.*, pages 799–804, 1999.

[9] A. B. Kahng, G. Robins, A. Singh, H. Wang, and Zelikovsky. Filling and slotting: Analysis and algorithms. *Proceedings International Symposium on Physical Design, Monterey(CA), U.S.A.*, pages 95–102, 1998.

[10] K. Keutzer, S. Malik, A. Newton, J. M. Rabaey, and A. Sangiovanni-Vincentelli. System-level design: Orthogonalization of concerns and platform-based design. *IEEE Transactions on ICCAD*, 19(21):1523–1543, December 2000.

[11] S. Khatri, A. Mehrotra, R. K. Brayton, A. Sangiovanni-Vincentelli, and R. H. Otten. A novel vlsi layout fabric for deep sub-micron applications. *Proceedings 36th Design Automation Conference, New Orleans(LA), U.S.A.*, pages 491–496, June 1999.

[12] D. MacMillen, M. Butts, R. Camposano, D. Hill, and T. Williams. An industrial view of electronic design automation. *IEEE Transactions on ICCAD*, 19(21):1428–1448, December 2000.

[13] R. H. Otten. Complexity and diversity in ic layout design. *Proceedings International Conference on Circuits and Computers, New York(NY), U.S.A.*, pages 764–767, October 1980.

[14] R. H. Otten. Global wires harmful? *Proceedings 1998 International Symposium on Physical Design, Monterey(CA), U.S.A.*, pages 104–109, April 1998.

[15] R. H. Otten. Timing closure: the solution and its problems. *Proceedings ASP-DAC2000, Yokohama, Japan*, pages 359–364, January 2000.

[16] R. H. Otten and R. K. Brayton. Planning for performance. *Proceedings 35th Design Automation Conference, San Francisco(CA), U.S.A.*, pages 122–127, June 1998.

[17] R. H. Otten and G. S. Garcea. Are wires plannable? *2001 International Workshop on System-Level Interconnect Prediction, Sonoma(CA), U.S.A.*, pages 59–65, April 2001.

[18] R. H. Otten and P. Stravers. Challenges in physical chip design. *Proceedings International Conference on Computer Aided Design, San José(CA), U.S.A.*, pages 84–91, November 2000.

[19] R. H. Otten, L. P. van Ginneken, and N. V. Shenoy. Speed: new paradigms in design for performance. *Proceedings International Conference on Computer Aided Design, San José(CA), U.S.A.*, page 700, November 1996.

[20] M. L. Rieger, J. Mayhew, and S. Panchapakesan. Layout design metholdologies for sub-wavelength manufacturing. *Proceedings 38th Design Automation Conference*, pages 85–88, 2001.

[21] W. Savage, J. Chilton, and R. Camposano. I.p. reuse in the system on a chip era. *Proceedings of the 13th ISSS, Madrid, Spain*, 99:2–7, September 20-22 2000.

[22] I. Sutherland, B. Sproull, and D. Harris. *Logical Effort: designing fast CMOS circuits*. Morgan Kaufmann Publishers, San Francisco, CA, U.S.A., 1999.

[2] Notably the observation of Dick Case in the early sixties that balanced computing system require memory size proportional to processing speed