PLANNING FOR PERFORMANCE

Ralph H.J.M. Otten(*) and Robert K. Brayton
University of California at Berkeley, California
(*)Also at Delft University of Technology, The Netherlands and Synopsys Inc.

Abstract

A shift is proposed in the design of VLSI circuits. In conventional design, higher levels of synthesis produce a netlist, from which layout synthesis builds a mask specification for manufacturing. Timing analysis is built into a feedback loop to detect timing violations which are then used to update specifications to synthesis. Such iteration is undesirable, and for very high performance designs, infeasible. The problem is likely to become much worse with future generations of technology. To achieve a non-iterative design flow, we propose that early synthesis stages should use "wireplanning" to distribute delays over the functional elements and interconnect, and layout synthesis should use its degrees of freedom to realize those delays. In this paper we attempt to quantify this problem for future technologies and propose some solutions for a "constant delay" methodology.

1 Introduction

In the past three decades, layout synthesis has relied on wire length and area minimization under the constraints of a technology file (design rule set) to generate masks for chips that showed acceptable functionality, yield and performance. Interconnect served merely as the realization of the net list and its influence on performance was negligible. As technology moves deeper into sub-micron feature sizes, and more components are integrated on a single chip, interconnect effects become more problematic.

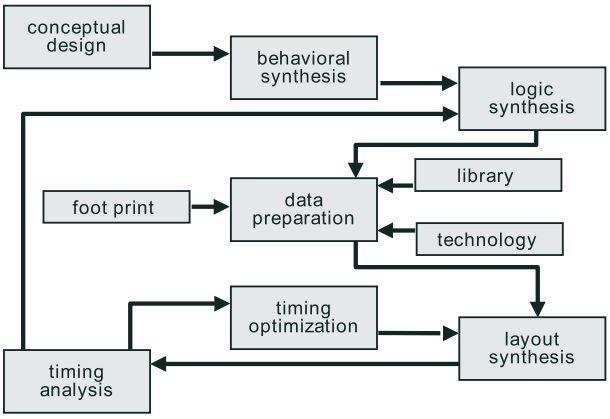

Figure 1: Synthesis with timing constraints

Up to now the effect of wiring on delay was determined by timing analysis tools that detect timing violations and produce either input for timing optimization procedures (such as transistor sizing, buffer insertion and fanout trees) or an updated specification file for higher level synthesis, expecting an improved gate and netlist for layout synthesis. Essentially the back-end of the design process has become a slow iterative scheme (shown in Figure 1), with no guarantee of convergence. Even if the process converges, it is uncertain how the final solution compares with the optimum.

Such iterations can be avoided only if the early design stages are integrated with layout synthesis, or at least are able to incorporate sufficient layout considerations without unnecessary constraints for the back end. This will require a completely new approach, especially for complex designs with very tight performance constraints. The required performance must be guaranteed by construction (and not the arbitrary outcome of indirect optimizations). This affects not only the way layout synthesis should be organized, but also higher levels of synthesis, and logic synthesis in particular.

2 Global wires

In recent years many sophisticated models for interconnect delay have been developed[10]. The complexity of these models and/or the size of the look-up tables used inhibits their use during synthesis, when the geometry of the interconnect is unknown, and when only estimates of length and topology are available. In these early stages only simple models such as Elmore's first moment matching can be used effectively. This model is the basis for almost all methods for reducing delay in point-to-point interconnection with unidirectional signal flow. The most common method is to split the wire into segments buffered by inverters[1]. This method, like any other, has its fundamental limitations which are often reached well before required performance has been achieved. In the next section we will study the limits of repeater insertion.

2.1 Critical lengths and critical delay

We use a first order model for a generic restoring buffer driving a capacitive load through a homogeneous line of length l given in Figure 2. [1].

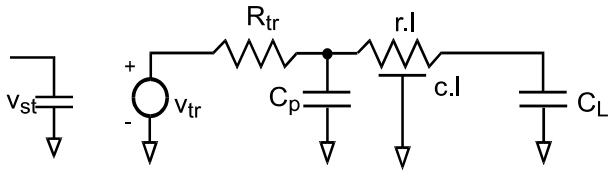

Figure 2: Generic restoring buffer model

The repeater is represented as a voltage source controlled by the voltage v_{st} at the input capacitance. This voltage source switches instantaneously when the fraction denoted by x, $0 \leq x \leq 1$, of the total swing has been reached. The switching at the voltage source is a perfect voltage step. The parasitic capacitance C_p is mainly composed of the drain capacitance of the transistors. The line is assumed uniform. If there is fanout, no resistance is assumed after branching.

Starting from this simple model, a general formula for the delay between the switching of the buffer and completing the

[1] Delay can also be reduced by "tapering" the wire, causing some difficult layout problems. Swing reduction with regenerative reaction to smaller voltage changes at the end of the line is another method that can speed up communication at the cost of increased noise sensitivity.

1-58113-049-x-98/0006/$3.50

x fraction of the swing at the end of the line can be derived, similar to [1]:

$$\tau = b(x)R_{tr}(C_L + C_p) + b(x)(cR_{tr} + rC_L)l + a(x)rcl^2.$$

R_{tr} is the equivalent transistor resistance. The constants a and b depend on the switching model of the repeater, that is on x.[2] This should be chosen so that when we divide the line in n equal parts by inverters the delays of the sections can be added. The size of the inverters is s (in multiples of the minimum size inverter), which makes $R_{tr} = r_o/s$, $C_L = s.c_o$ and $C_p = sc_p$. The initial driver of the line is assumed to have the same size, possibly after cascading up from smaller initial drivers for optimum speed. The total delay for n such sections of length l/n is

$$T = n.\tau = n\left[br_o(c_o + c_p) + b(c\frac{r_o}{s} + rc_o s)\frac{l}{n} + arc\frac{l^2}{n^2}\right] =$$

$$= br_o(c_o + c_p)n + b(c\frac{r_o}{s} + rc_o s)l + arc\frac{l^2}{n} \quad (1)$$

Now we can ask for the values of s and n which give the minimum delay. For n we obtain

$$\frac{\partial T}{\partial n} = br_o(c_o + c_p) - arc\frac{l^2}{n^2} = 0$$

or the optimum length of each section is

$$l_{crit} = \frac{l}{n_{opt}} = \sqrt{\frac{\frac{b}{a}r_o c_o\left(1 + \frac{c_p}{c_o}\right)}{rc}} = \frac{P}{\sqrt{rc}}. \quad (2)$$

Accepting that $r_o c_o$ and $r_o c_p$ are process constants makes the optimum distance between inverters only dependent on the rc product per unit length of the wire.

THEOREM 1 *The length of a section in an optimally segmented*[3] *line is inversely proportional to* $\sqrt{rc}$.

P depends on the process and the delay model (x) only. Since r and c differ from layer to layer, these distances also differ from layer to layer.

Substituting n_{opt} in (1) yields

$$T(n_{opt}) = \left(2\sqrt{abrcr_o c_o\left(1 + \frac{c_p}{c_o}\right)} + \frac{bcr_o}{s} + brc_o s\right)l$$

which shows

THEOREM 2 *The delay of a line that is optimally segmented is linear in its length.*

The optimum repeater size is obtained as

$$\frac{\partial T}{\partial s} = b(rc_o - \frac{r_o c}{s^2})l = 0 \Rightarrow s_{opt} = \sqrt{\frac{r_o\, c}{c_o\, r}}$$

which is independent of n, the number of inverters used. By substituting the optimum repeater size and the optimum number of sections into (1) we find the delay of the line is

$$T(l) = 2l\sqrt{rcr_o c_o}\left(b + \sqrt{ab\left(1 + \frac{c_p}{c_o}\right)}\right)$$

which of course is linear in l. More surprisingly, substituting the critical length shows that the delay of a section of critical length does not depend on the line resistance and capacitance:

$$\tau_{crit} = 2br_o c_o\left(1 + \sqrt{\frac{b}{a}\left(1 + \frac{c_p}{c_o}\right)}\right) = 2br_o c_o\left(1 + \frac{P}{\sqrt{r_o c_o}}\right) \quad (3)$$

and therefore only depends on the process (and the model), but not on the wiring layer.

THEOREM 3 *The delay of a section in an optimally buffered line is the same for all layers.*

Note that all derivations were made on a chain of inverters driving an uniform wire. Using this in more general networks, with different fanouts and branch-off geometries is therefore only an approximation, which can be made more accurate if isolation techniques are used to offset fanout effects.

2.2 Model justification

Since we use the results only for point-to-point connections (that is without branching) between restoring circuits, first moment matching is accurate enough. However, some remarks concerning the model parameters are in order.

The via resistance shows up as a resistor in series with the R_{tr} in Figure 2. It is reasonable that this scales with the size of the inverter[4], and hence can be absorbed in r_o and the formulas do not change. Although $r_o c_o$ is no longer a process constant and a layer dependence is introduced in the critical delay, experiments show that the via resistance, even to the top layer, is negligible and has hardly any effect on the wave forms.

The line was assumed to have constant capacitance per unit length. For advanced technologies, this is dominated by capacitance to sections of interconnection, especially neighboring sections in the same layer. Since these may undergo voltage changes, the value is not even constant. The latter effect may cause variations in the effective capacitance by up to a factor of 3. To make use of the derivations in the previous section, before the geometry of the wiring is known, requires the enforcement of a routing style which produces a time-invariant homogeneous line[5]. In addition, its resistance and capacitance should be known a priori.

The remaining problem with the model is the determination of the effective transistor resistance. It is reasonable that such a resistance exists if we only consider one waveform and a fixed x. The most practical way to obtain useful parameters is to simulate a ring of an odd number of buffer sections with large transistors (100 times minimum size) after extracting c very accurately. Then we optimize speed by varying the value of l for each section to obtain l_{crit}. This will give P of equation 2 since r is known quite accurately. With the length of each section fixed at l_{crit} the ring is optimized next for speed once more, now by varying s. This will yield τ_{crit} and by equation 3 therefore $r_o c_o$. Since we can accurately calculate c_o from the transistor geometry[6], we get

$$r_o = \frac{1}{4c_o}\left(\sqrt{P^2 + \frac{2\tau_{crit}}{b}} - P\right)^2.$$

2.3 Numerical data

To quantify what this all means we performed some calculations using a fictitious, but well reviewed technology file based on [11], and an extraction program [2][7] for solving exact 3-dimensional field problems. The critical lengths and the

[2] If $x = 0.9$ (90%-swing) then $a = 1.0, b = 2.3$. Mostly, we use $(x = 0.5)$ yielding $a = 0.4, b = 0.7$.

[3] We call it an optimally segmented line rather than an optimally buffered line because this length is independent of the buffer size s.

[4] The contact resistance (the largest part) will scale if the contact area grows with buffer size, and also the cross section of the via is likely to scale then.

[5] One (possibly drastic) way of achieving this is to shield each signal line with neighboring lines tied to ground.In addition to reliable characterization, this style eliminates most cross-coupling noise problems.

[6] Theoretically, we don't have to do this since we obtained s_{opt} in the second optimization and $c_o = r_o c/(r s^2_{opt})$. However, since τ_{crit} is likely to be insensitive to s at the optimum, the value of s_{opt} is probably not very accurate, even though τ_{crit} is accurate.

[7] For more information, consult
`http://cas.et.tudelft.nl/research/space.html`

critical delay are given in Table 1[8]. Each layer has its own critical length, and the values are pairwise close. Such a pair is called a *tier*. With a bit of process tuning the critical lengths within a tier can be made almost the same where the difference is mainly in the "between layers" capacitance. The higher tiers having longer critical lengths than the lower ones.

critical parameter	feature size	
	0.25 μ	0.10 μ
$l_{crit}(m1)$	10440	6757
$l_{crit}(m2)$	10600	7162
$l_{crit}(m3)$	36000	43446
$l_{crit}(m4)$	38400	45135
$l_{crit}(m5)$	63200	64932
$l_{crit}(m6)$	62000	56892
$l_{crit}(m7)$		97581
$l_{crit}(m8)$		93378
$T(l_{crit})$	205ps	80ps

Table 1: Critical wire lengths measured in feature size units.

Note that the critical length, measured in the feature size, changes much less than proportional with the feature size! This may come as a surprise, but is mainly due to velocity saturation effects, and therefore represents a trend that will affect smaller feature sizes even more. Other recent studies also indicate that even with scaling down in the logic blocks, the gate delay will continue to dominate the performance [4].

Today's synthesis is capable of handling blocks with up to 10000 gates. A square with side lengths of $l_{crit}(m1 - m2)$ can contain on the order of a hundred of these blocks in a 0.1μ technology. So, even with careful extrapolation, this means that fairly complex blocks can still be designed while mainly controlling gate delay.

3 Wire planning

The term *wire planning* was coined more than ten years ago "to describe an approach that first focuses on determining an optimal plan for global wiring" [3]. In its original context its task was mainly to identify groups of nets each connecting to (almost) the same set of modules. The most common example are *buses*: when identified routing complexity can be reduced considerably by handling bus wires as groups inside *data path generators*.

Another, new task for wire planning is indicated by the computations and derivations of Section 2. Although complex modules can still be designed using present day logic synthesis methodologies, controlling mainly the gate delays to achieve performance, at the chip level, future technologies will involve many (hundreds to thousands) of such complex modules, and at this level wire delay begins to dominate. Wire planning should produce a location for these modules with the main concern that timing constraints on input/output paths are met. This requires knowledge of the global interconnection structure and the performance implications of the functionality of the modules. In the early "conceptual" stages of a design there is not much more than an awareness of size-speed trade-offs. This accuracy depends completely on the design experience in the team. In general, all that can be said is that these interpolated delay-area relations appear convex. Delay in synchronous systems is often measured in terms of the number of clock cycles to complete the computation of the module. Obviously, a module that takes more cycles to do its computation never requires more area than one that takes fewer cycles. Although defining speed of modules in general is not possible, the reasoning will always be similar, whether speed in a given technology is obtained by sizing, parallelism, or other means.

Knowing module functionality and interconnection structures implies a decomposition. Initially, such decompositions emerge solely on the basis of functional considerations, with little regard for their impact on both the performance of the product as well as the efficiency of synthesis steps later on. Therefore, while the design evolves into a hierarchical description acceptable for behavioral synthesis, wire planning tools should aid in quick analyses and proposals for function duplication, absorption and decomposition as well as module (re)locations and (partial) pad planning.

Examples of wire planning tasks are establishing the existence of a module placement in which no path from input to output has to make detours, assigning time budgets to modules such that area is minimized, establishing the existence of a valid retiming and producing a valid minimal-area retiming, assigning wire sections to layers so that feasible time budgets are preserved, and encouraging floorplans that lead to efficient optimizations in later stages of the design.

The final result of conceptual design aided by wire planning is a composition of a network of blocks and interconnections along with well established time budgets and delays. Considering the data concerning critical lengths, blocks will be small compared to these critical "units". They can be treated without internal distributed delays, and their wiring is mostly realized in the lower levels of metal. The tools can aid in creating subsets of regular grids with blocks at grid points and predefined wire segments on the grid lines. The latter enables good characterization of these segments, and routing consists of "using the available segment" rather than "placing segments". By the time that synthesis begins to create the gate and net lists, the delay on the "global" wires is quite well established and therefore also the timing budget that remains for the blocks.

3.1 Monotonic wire plans

Consider a high level description of a design described as a *functional network* modeled by as a directed acyclic graphs with primary inputs as sources, primary outputs as sinks and "functions" on the other nodes. There is an arc from one node to another if the "result" of the former is used as an argument in the latter. If a primary output depends on a primary input, there must be a path connecting them, possibly passing through other blocks, and possibly sharing some with other paths. Total delay is the sum of the delay in the blocks and the delay in the wires. If wires are composed entirely out of sections with critical delay, the total wire delay on a path is a multiple of the critical delay, and is invariant with respect to how the functional units are distributed over the restoring sites (*end points of critical sections*). If a functional block is placed at each "grid point" along a path then no repeaters are necessary. A *wire plan* in this context is a position for all the nodes in the functional network and a pin assignment for all primary inputs and outputs. Such a wire plan is called *monotonic* if all interconnections can be made so that the $\mathcal{L}_1$-length ("Manhattan length") of each input/output path is equal to the $\mathcal{L}_1$-distance ("Manhattan distance") between the two associated i/o pins. Under the model this is the fastest possible

[8] Table 1 is by Amit Mehrotra of UC Berkeley with independent corroboration by studies by Lixin Su, Sunil Khatri, and Dennis Sylvester. The associated technology files can be found at `http://www-cad.eecs.berkeley.edu/Respep/Research/nexsis/strawman`

wire plan for a functional network with that pin assignment[9] having its wires in a given tier.

For a given pin assignment a monotonic wire plan may not exist. This existence question has been answered in [5] as follows. The *support* of a node is the set of primary inputs connected to that node by a directed path. The *range* of a node is the set of primary outputs connected to it by a directed path The *inbox* of a node is the smallest iso-rectangle containing its support, and the *outbox* is the smallest iso-rectangle containing its range. A *bridge* of a node is a minimum $\mathcal{L}_2$-length line connecting its inbox with its outbox. Using these ideas and and working out a few special cases leads to Theorem 4:

THEOREM 4 *Every node in a monotonic wire plan must be placed within the smallest iso-rectangle containing its bridge.*

A simple proof by induction then yields:

THEOREM 5 *A functional network has a monotonic wire plan with respect to a given pin assignment if and only if every node has a unique bridge.*

This makes it very easy to find out whether such a wire plan exists: we only have to check on a node by node basis whether each node in the network has a unique bridge. Such a check is extremely simple since

THEOREM 6 *A node has a unique bridge if*

1. *the support or the range contains a single pin, or*
2. *the range is contained in an iso-line while the support is on a single line perpendicular to that, or*
3. *the output box is in the "projection" of the input box, that is the two boxes have disjoint support in both axes, except for at most one point.*

Note that a placement conformant to Theorem 4 is not necessarily a monotonic wire plan. A valid placement, but possibly having nodes at the same position, is assigning each node the point which the output box has in common with the bridge[10].

Of course, certain deviations from strict monotonicity may be necessary or desirable, because of availability of space or for sharing functionality with other paths. However, deviation from monotonicity can only be allowed if the timing requirements are not violated. Note that monotonicity can always be obtained by duplicating functionality, synthesizing faster blocks, and absorbing functions in their fanout. In the extreme, a monotonic wire plan always exists if each output is produced by a single node.

Once the wireplan for a functional network has been determined, which means that the delay on the arcs of this network is known, the remaining time budgets have to be distributed over the function nodes. If the same graph is a suitable model for this task, and the sources and sinks have arrival times and required times assigned to them, a simple (quasi-)convex optimization problem can be used to answer questions such as "what is the smallest network that does not violate any timing constraints?" Size is in this case the sum of the areas assigned to each node according to its area-delay trade-off.

3.2 Valid retiming

At this level, our goal is to assign delays (measured in units of clock periods) to wires and to get an appropriate retiming of the chip. We assume that the dependency graph is acyclic. At each input and output, a number of latches are put in a consistent[11] way to reflect the number of cycles allowed between the input and the computed output.

Given a wire plan, and therefore the distance k_{ij} between two (point) modules, measured in number of critical lengths at the highest tier, we want to see if it can be retimed so that all connections can be made in a single cycle. For example, if a wire needs 2 cycles even if assigned to the highest level of metal, then a solution is to split the wire into two parts separated by a latch. The following problem is formulated. Given an integer assignment on each wire, k_{ij} does there exist a retiming of the circuit such that whenever there is wire between module i and j the number w_{ij} of latches assigned to that wire, satisfies

$$k_{ij} \leq w_{ij}?$$

If the answer is yes, the placement is *valid* and has a *valid retiming*.

A second problem is, given a placement, to find a valid retiming, if it exists, which minimizes the total area. Here the area-delay tradeoff curves with cycles versus area for each module is used. At the start, each module is modeled with two nodes separated by a edge. Now a retiming will move some of peripheral latches to internal edges. After a retiming, the number of latches in a module edge may be changed. This affects the area of the module through its area-delay trade-off curve. We call this the minimum-area valid retiming problem.

Finally, we look for the placement which gives the least area of all the minimum area valid retimings[12].

3.3 Layer assignment

Here we assume that an RTL description of the design is available. Our aim is to assign a delay number δ_{ij} to each wire, $i \rightarrow j$ in such a way that all delays from chip input or latch output to chip output or latch input can be made in the assigned number of clock periods for that connection. We assume that if a path is allowed a delay longer than a single clock period it is divided into an appropriate number of pseudo-nodes. Thus every combinational path must be within a single clock period. Let π be such a path. We assign half the clock period to this path, reserving the other half for the gates on the path. So

$$\sum_{ij \in \pi} \delta_{ij} = \frac{1}{2}$$

where δ_{ij} will be the delay assigned to connection $i \rightarrow j$. We solve the *zero-slack distribution problem* where all paths satisfy the above equations.

Next, from the different levels of metal, we create a number of *wire types*[13]. Each wire type has a critical length l_{crit}. These are sorted in ascending order and denoted as,

$$l_1 < l_2 <, \ldots, < l_m$$

Now given a placement, let the Manhattan length of wire $i \rightarrow j$ be denoted by d_{ij}. Wire $i \rightarrow j$ can be seen to be in a rectangle bounded by (x_i, y_i) and (x_j, y_j). This rectangle is assigned a cost C_{ij}^k which depends on the wire type k

[9] Under a model where interconnections have capacitance but negligible resistance, a monotonic wireplan has the minimum total wire capacitance. This can be useful when power is a major concern and may be relevant for logic synthesis when a pin assignment is given [5].

[10] Also the points that input boxes have in common with the bridge is a feasible set.

[11] A similar notion was used with the so-called "peripheral" retiming.

[12] What is missing in this formulation is how the assignment of wires to layers plays a role. In the above formulation, only the top level wire type is considered. The fact that a wire can be placed on a lower level of wire and still meet its timing obligation is not considered. A possible answer is to modify the total area cost function, to penalize wires that are put on higher layers.

[13] For example, each level of metal k may be divided into a number of *wire types* (k, i) where i denotes the number of wires tied together on level k to construct a "thicker" wire. This gives us a finer grained resolution for having wires that reach different distances in the same time. Another way to create a wire type is to assign a wire to multiple layers.

assigned to the wire. A wire is more costly if its type is on a higher level or if its rectangle is nearer the center of the chip. We minimize

$$\sum C_{ij}^k$$

Here $i \to j$ is assigned to the k^{th} wire type if and only if

$$\frac{d_{ij}}{l_k} \leq \delta_{ij} \leq \frac{d_{ij}}{l_{k+1}}$$

Thus wire $i \to j$ will be assigned to a wire type where the delay on the wire, $\frac{d_{ij}}{l_k} \leq \delta_{ij}$. Therefore for any path π we have

$$\sum_{ij \in \pi} \frac{d_{ij}}{l_k} \leq \frac{1}{2}$$

If a solution exists, then all wire segments can be assigned a fixed wire delay, δ_{ij}[14]. A solution may not exist if there is no placement where all the delays can be met. In that case, we may have to return to the higher level wireplanning problem and even possibly alter the chip latency.

We also note in the derivation of the critical wire length for each kind of wire, that the optimum sized driver was used. For a wire of less than critical length, we have a choice in the size of the driver which gives more flexibility to adjust the drivers to meet the assigned delays.

4 Layout synthesis

To complement an approach based on wire planning, layout synthesis should realize the functional blocks in such a way that the delays in the blocks do not exceed their timing budgets, or rather keep them right on target. Since logic synthesis knows the budgets after wire planning and the range of available gates, it should deliver a gate list with an assigned specified delay for every gate. Layout synthesis should produce a network in which each gate causes exactly that delay. This is called *constant delay synthesis* [6]. Given a fixed delay for a gate, its size becomes a function of the output capacitance.

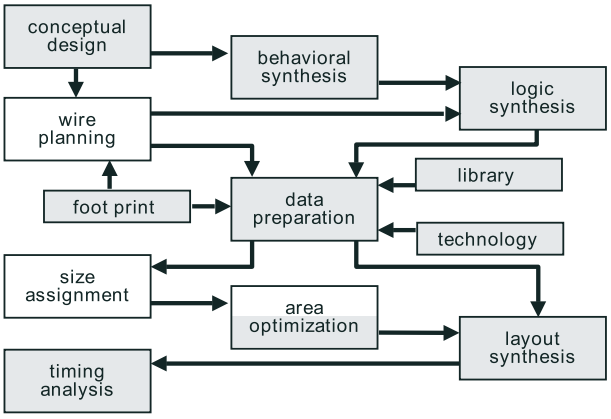

Figure 3: Constant delay flow

This of course is not without consequences for the back-end tools. Sizes now have to be assigned according to the results of logic synthesis, and scale only with the imposed capacitances on the outside. Timing optimization is out of the question: buffer insertion can only serve as an area reduction trick. New cell libraries have to be developed to adequately reflect the demands of delay-based requirements. And finally, the layout generation must be capable of handling a variety of cell sizes, and absorb changes in sizes efficiently.

[14] We have only indirectly accounted for the number of available wires of wire type k by assigning a wire to the least level which gives the required delay. This assumes that wires at the lower levels are the more plentiful. The cost of a wiring rectangle is weighted by its relative overlap with the center of the chip, but the number of wires in the wiring rectangle is not accounted for in this formulation.

4.1 Fixed delays

Again starting from the model of Figure 2, but not including the wire (Figure 4) leads to a delay formula[12] which is the sum of two terms, the *effort delay* and the *parasitic delay*:

$$\tau = bR_{tr}C_L + bR_{tr}C_p = br_o c_o \frac{C_L}{C_{in}} + br_o c_p = \frac{g}{f} + p.$$

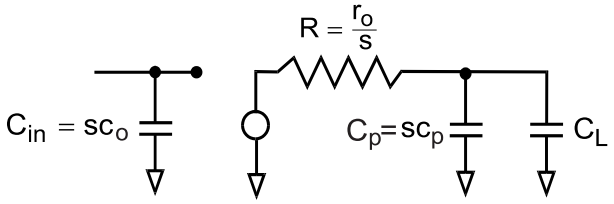

Figure 4: Gate model for obtaining a size independent delay expression

The parasitic delay $p = br_o c_p$ is independent of size. The effort delay g/f is a product of *computing effort* $g = br_o c_o$, and *restoring effort*

$$\frac{1}{f} = \frac{C_L}{C_{in}}$$

The computing effort is also size independent, but in general depends on the function, topology and relative transistor dimensioning of the gate. The important observation is that τ can be kept constant by fixing $f = C_{in}/C_L$. This leads to a new paradigm in synthesis [6, 9]: any delay imposed by synthesis can be realized, provided that the sizes of the gates can be continuously adjusted.In [6] the authors show that the size of a gate varies linearly with the load under constant delay. This enables size assignment after logic synthesis has fixed the *scaling factor* f for all gates and of course the net list.

4.2 Cell generation and shape assignment

Wire plans perform their analyses on point placements or sequences, most likely under the presence of larger blocks that may be pre-placed. This position information along with a possibly partial pin assignment must be preserved during the layout synthesis when the results of size assignment and area optimization become available. This requires efficient and robust floorplan optimization. These qualities heavily depend on the floorplan to be optimized. This can only be achieved by maintaining sliceability throughout the design, from the early wire planning stage down to the determination of the final dissection. This presumed “restriction” is amply offset by the guaranteed optimum in its class.

Cell generation is most likely the big challenge in a constant delay approach. The set of functions can be quite small, but extensive research is necessary to determine which sizes should be made available. Ultimately, a library of cell layout generators seems to be the way to go. In addition, yield is also an issue here.

5 Conclusions

In synthesizing high performance chips using present day design practices, the meeting of timing constraints necessitates an iteration which is not guaranteed to converge. In future technologies, unless these global delays are planned up front, convergence will be even more of a problem and even if convergence is achieved, the answer is likely to be far from optimum. This suggests a shift in design methodology where a global wire plan is put in place beginning at the conceptual stage of the design. We propose an approach in which a wire plan is created before the functionality of the blocks in that plan has been fixed. This allows for better control over the performance of the total design. Inherent to such an approach is that wire delay is accurately known wherever it has impact. This means that "global wires" should be well characterized a priori, which requires a strict layout styles. We have chosen to use a minimum width optimally buffered interconnections with a fairly stable electrically environment. Adherence to this style provides delays linear with distance, and thus invariance over equal length paths. Sharing functional blocks is likely to cause detours in one or more paths, causing additional delay. Creating a wire plan may distribute units all over the chip, thus abandoning the principle of easily recognizable and recoverable blocks, in exchange for exact knowledge of delay on connections and control over delay in the blocks. Enforcing delays in the blocks means that sizes become uncertain, and with uncertainties in size also distances become uncertain. If a block cannot be synthesized with the required delay in the available space, then the wire plan cannot be realized. Thus, reliable predictors in the early stages must be developed to obtain a non-iterative design flow. Of course, existence of solutions can never be guaranteed under too strict timing requirements, but we postulate that this new methodology can find solutions for a broader range of specifications than the current methods.

Acknowledgment: The vision that synthesis should be conducted with guaranteed performance was developed by Lukas van Ginneken. The authors are grateful to the wire planning group, Philip Chong, Wilsin Gosti, Hiroshi Murata,and Mukul Prasad, at Berkeley for stimulating discussions, to the Nexsis group at Berkeley, in particular Amit Mehrotra, Sunil Khatri, Subarna Sinha, and Philip Chong, who made the studies that quantified the critical lengths and to the SRC under grant 324 and the California Micro program, 96-091, with industrial support from Synopsys, Cadence and Fujitsu.

References

[1] H.B. Bakoglu, *Circuits, interconnections, and packaging for vlsi*, Addison-Wesley Pub Co, 1990

[2] F.Beeftink, A.J. van Genderen, N.P. van der Meijs *Accurate and efficient layout-to-circuit extraction for high speed mos and bipolar/bicmos Integrated circuits* ICCD, Oct. 1995

[3] R.K. Brayton, C.-L. Chen, J.A.G. Jess, R.H.J.M. Otten, L.P.P.P. van Ginneken, *Wire planning for stackable designs*, Proceedings 1987 International Symposium on VLSI Technology, Systems and Applications, Taipeh, Taiwan, pp 269-273, May 1987

[4] P.D. Fisher, *Clock cycle estimation for future microprocessor generations*, 1998

[5] W.Gosti, *Wire planning in logic synthesis*, 1998

[6] J. Grodstein, E. Lehman, H. Harkness, B. Grundmann, Y. Watanabe, *A delay model for logic synthesis of continuously-sized networks*, ICCAD, Nov. 1995

[7] Y. Kukimoto, R.K. Brayton, P. Sawkar, *Delay-optimal technology mapping by dag covering*, DAC, June 1998

[8] R.H.J.M. Otten, *Layout compilation*, in *Design systems for vlsi circuits*, edited by G. DeMicheli, A. Sangiovanni-Vincentelli and P.Antognetti, pp.439-472, Martinus Nijhoff Publishers, 1987

[9] R.H.J.M. Otten, L.P.P.P. van Ginneken, N.V. Shenoy, *Speed: new paradigms in design for performance*, ICCAD, Nov. 1996

[10] L. Pileggi, *Delay metrics*, ISPD98

[11] Semiconductor Industry Association, *The national technology roadmap for semiconductors: technology needs*, California, U.S.A., 1997

[12] I. Sutherland, R. Sproull, *The theory of logical effort: designing for speed on the back of an envelope*, in *Advanced Research in VLSI*, UC Santa Cruz, 1991