Power-Aware Slack Distribution for Hierarchical VLSI Design

Hyung-Ock Kim
Korea Advanced Institute of Science and Technology
Daejeon 305-701, Korea

Youngsoo Shin
Korea Advanced Institute of Science and Technology
Daejeon 305-701, Korea

Abstract— **Hierarchical design plays an important role in microprocessor and ASIC domains where design complexity limits design productivity and tool capacity. Slack distribution, which assigns arrival times and required arrival times at hierarchical boundaries, is a key component in resolving timing issues. In this paper, we present a new slack distribution methodology targeting power minimization. The approach is formulated as a nonlinear optimization problem, which can be solved very efficiently. Experiments with example designs show that up to 14% power can be saved with the proposed methodology.**

I. Introduction

The enormous growth in VLSI complexity has led to unprecedented problems in design methodologies in microprocessor and ASIC domains. The limitation of tool capacity and design productivity requires a design to be processed as a hierarchical fashion. In hierarchical design, as opposed to flat design, a design is partitioned into hierarchical blocks. Hierarchy can constrain optimization process of the physical designs such as placement and routing. However, it is very efficient for design architectures with natural functional partitions and for a design with different design teams working independently on their own partitions at the same time. Microprocessor design heavily relies on hierarchical design methodology [1], where a unit such as an instruction decode unit represents a top-level design hierarchy. Hierarchical design is also prevalent in ASIC domain, such as in System-on-a-Chip (SoC) where reusable cores, either hard- or soft-, constitute a hierarchical boundary.

Hierarchical design requires additional design steps such as pin assignment, wiring resource assignment, and resolving timing issues at hierarchical boundaries [1]. Timing issues are especially painful to handle and frequently require iterations of feeding timing assertions and timing abstractions down and up the hierarchies respectively. Timing assertions of each partition consist of arrival times (ATs) with phase tags and slew at the inputs and required arrival times (RATs) with phase tags and output capacitance loads at the outputs. A simple and intuitive way to assign ATs and RATs is to rely on the length of the longest path from partition inputs to latch inputs and from latch outputs to partition outputs respectively. This may be a reasonable way in view of timing, but not necessarily in view of power consumption. If there is a partition with long timing path but with less power consumption, instead of assigning a large time budget we may try to reduce the timing path by exploiting a parallelism or by exploiting dual threshold voltages, for example, which potentially can lead to increased time budget for partitions that are connected to it, which in turn can be used to reduce power consumption of those partitions.

In this paper, we present a new technique for timing assertion generation, which we call power-aware slack distribution (PASD). Timing assertions are generated in such a way that the total power consumption is minimized yet all timing paths remain in positive slacks. We exploit multiple V_{DD} technique, such as voltage island [2], thus assuming that each partition can be powered by independent voltage source. We show that PASD can be formulated and solved under nonlinear optimization framework.

The remainder of the paper is organized as follows. In the next section, we explain the motivation of our approach and discuss related work. In Section III, we address our technique of PASD and in Section IV we show experimental results on example designs. Finally, a conclusion follows in Section V.

II. Motivation and Related Work

In hierarchical design flow, a design is partitioned into hierarchical blocks with timing assertions imposed at the boundary of a chip, namely ATs at chip inputs and RATs at chip outputs. Since each block is to be processed independently, these chip-level timing assertions need to be translated into block-level timing assertions (ATs at block inputs and RATs at block outputs). This process is called *slack distribution*. Slack distribution affects the overall performance of the design, since it handles timing paths crossing block boundaries, which may dictate the operating frequency of the design. Since the supply voltage is determined by the worst timing path, slack distribution affects the power consumption of the design as well.

Fig. 1 shows an example of a hierarchical design, where $C_{ij}, i = 1,2,3,4$ denotes a j-th combinational sub-block in a `block` i. C_{0j} represents a top-level logic, which consists of glue logic, test logic, and so on. Suppose in the process of slack distribution we want to assign late RATs at the outputs of C_{14} (thus large time slack for C_{14}). This implies late ATs at the inputs of C_{31} and C_{02}[1], which in turn implies late ATs for C_{21}. The overall effect can be observed in terms of the number of

[1]Note that ATs for C_{31} and C_{02} are different from RATs for C_{14}, since there is interconnect delay.

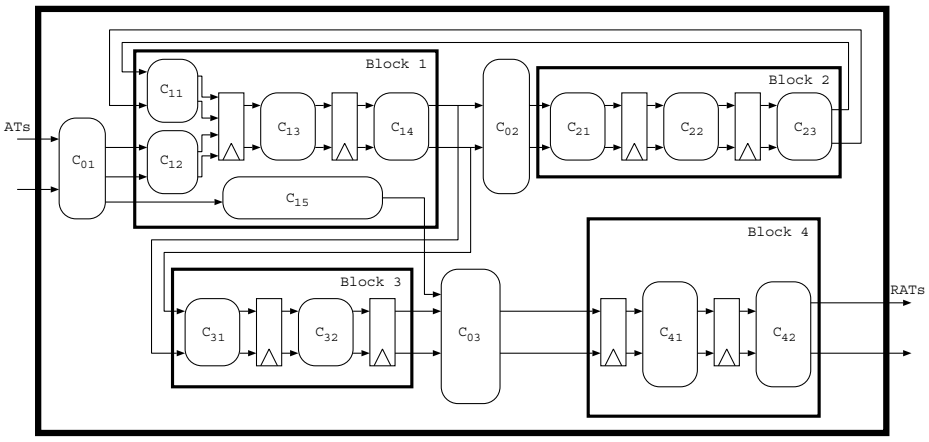

Fig. 1. An example of a hierarchical design for slack distribution.

logic to implement each sub-block. There is a high probability that the number of logic to implement C_{14} is reduced, because logic synthesis can target area optimization with *loose* timing constraints. The opposite is true for C_{31}, C_{02}, and C_{21}. If multiple V_{DD}s are allowed, such as in voltage island, we have more choices to take advantage of this process of slack distribution. If C_{14} contains the worst timing path of `block 1`, we may reduce the supply voltage of `block 1` instead of trying to reduce the number of logic through re-synthesis. For C_{31}, we may either re-synthesize it to accommodate *tighter* timing constraints or increase the supply voltage of `block 3`, depending on which decision has favorable effect in view of power consumption. This illustrates the motivation of our approach to slack distribution and its nature of optimization process.

Power-aware slack distribution is investigated in [3]. A block with the largest switched capacitance is selected and its timing slack is determined in proportion to the percentage of its switched capacitance to the total switched capacitances of blocks that belong to the path from source to sink. The process is repeated until slacks are assigned to all blocks. The scope of the approach is quite limited, because it can be applied to combinational networks where each component is of relatively large size so that its supply voltage can be controlled independently. Thus, it cannot be applied to general sequential logic networks such as the one in Fig. 1.

III. Power-Aware Slack Distribution

Starting from hierarchical blocks with chip-level timing assertions, our objective in PASD is to assign timing assertions (ATs and RATs) at block boundaries in such a way that we minimize the sum of power consumption of each hierarchical block while all timing paths (either those in the blocks or those crossing block boundaries) are remained in non-negative slacks. Since lowering V_{DD} is the most effective way to reduce the power consumption of CMOS circuits and circuit delay increases with decreased V_{DD}, we may want to assign early ATs and late RATs for each block, thus allowing more circuit delay that can be turned into decreased V_{DD}. This is not possible, however, because late RATs of one block implies late ATs of the blocks that are connected to it as explained in the previous section.

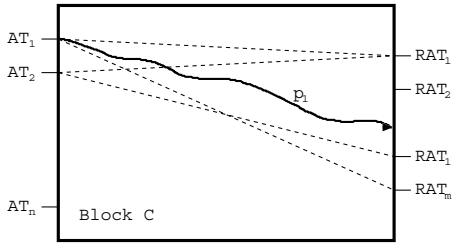

Fig. 2. An n-input m-output combinational circuit.

We may arbitrarily assign ATs and RATs, adjusting V_{DD} of each block while its timing is satisfied, and checking the power consumption of all blocks. This process may be repeated until we are satisfied with the power consumption. Since this is time-consuming process and optimality is not guaranteed, in PASD, we instead take all worst timing paths and translate them into formal constraints that are expressed as a set of inequalities. Each timing path is then modeled in terms of V_{DD}, which turns the timing constraint inequalities into those with parameters of V_{DD}s. These are then solved through nonlinear optimization formulation, which outputs V_{DD} of each block. ATs and RATs at block boundaries are determined by relations derived from timing constraints inequalities. When outputs of a block under lower operating voltage drives inputs of other block under higher operating voltage, level shifters are required. However we omit the power and delay overhead of level shifters for simplicity.

A. Timing Constraint Inequalities

Consider a block of combinational circuit (C) with n-inputs and m-outputs as shown in Fig. 2. We have n ATs, which are either given in case of chip-level inputs (e.g. ATs of C_{01} in Fig. 1), implicit in case of latch outputs (e.g. C_{14}), or to be assigned in the process of slack distribution (e.g. C_{12}). The same holds for RATs. Let the worst timing path in a fanout cone of an input i (thus accompanied with AT_i) be denoted as

p_i and its corresponding delay as $d(p_i)$. The timing constraint of C can be expressed as

$$\min_{\forall i \in I(C)} (RAT(p_i) - AT_i - d(p_i)) \geq 0, \quad (1)$$

where $I(C)$ is a set of inputs of C and $RAT(p_i)$ denotes RAT of an output of C that belongs to p_i. In this expression, it is assumed that all primary outputs in the fanout cone of an input i are connected to the same block and have the same RAT.

We can re-express (1) depending on the types of timing paths: those from block inputs to latch inputs (see timing paths in sub-blocks C_{11} or C_{12} in Fig. 1 for example), between latch boundaries (C_{13}), from latch outputs to block outputs (C_{14}), and from block inputs to block outputs (C_{15}). As an example of a `block 1` in Fig. 1, the timing constraints can be expressed as follows[2]:

$$\begin{aligned}
C_{11}: &\quad P - \max_{\forall i \in I(C_{11})} (AT_i + d(p_i)) \geq 0 \\
C_{12}: &\quad P - \max_{\forall i \in I(C_{12})} (AT_i + d(p_i)) \geq 0 \\
C_{13}: &\quad P - \max_{\forall i \in I(C_{13})} d(p_i) \geq 0 \\
C_{14}: &\quad \min_{\forall i \in I(C_{14})} (RAT(p_i) - d(p_i)) \geq 0 \\
C_{15}: &\quad \min_{\forall i \in I(C_{15})} (RAT(p_i) - AT_i - d(p_i)) \geq 0
\end{aligned}$$

where P denotes cycle time[3]. The timing constraints of other blocks can be expressed similarly.

If we change AT and/or RAT of one block and re-synthesize it, the timing constraints do not hold for a new network, since for each input of a sub-block we have a new worst timing path (p_i) thus a new delay ($d(p_i)$). However, we do not allow re-synthesis in our approach, since we take advantage of a new slack assignment toward finding an optimal set of V_{DD}s. In summary, our approach is as follows: we are given a network of hierarchical blocks with chip-level timing assertions; identify combinational sub-blocks of each hierarchical block and those in top-level (e.g. C_{01} in Fig. 1); synthesize each sub-block in case it is given as RTL instead of a netlist; build a set of timing constraints for each block; model the timing path in terms of V_{DD}, which is to be explained in the next subsection; combine timing constraints of blocks that are connected to each other; and solve the inequalities that describe timing constraints of all blocks.

B. Modeling of Timing Path

In CMOS digital circuits, a gate delay can be expressed as [4]:

$$t_{pd} \propto \frac{C_L V_{DD}}{(V_{DD} - V_{TH})^{\alpha}}, \quad (2)$$

where C_L is the load capacitance and V_{TH} is the threshold voltage. α is a constant and equals to 2 for long channel MOSFETs and equals to about 1.3 for short channel ones. We may use (2) to capture *the delay of a timing path*, which allows us to model $d(p_i)$ in (1) in terms of V_{DD}. However, we use a more simplified relationship to approximate the delay of a timing path as follows:

$$d(p_i) = \frac{K_i}{V_{DD,i}}, \quad (3)$$

where K_i is a constant and $V_{DD,i}$ is a supply voltage.

Thus, for each combinational sub-block C_{ij}, we measure its worst path delay while we vary its supply voltage, which we take for curve-fitting to find K_i. Fig. 3 shows the results of this process for examples of four combinational sub-blocks. Each circle shows the worst path delay for one of seven available V_{DD}s and the curve corresponds to (3) with K parameters shown in the plot. All plots show very good match between measured delay and approximated delay model.

If we substitute $d(p_i)$ in (1) with timing path model in (3), we have timing constraints with unknown V_{DD}. Furthermore, if we combine timing constraints of sub-blocks that are on the path from latches to latches (see C_{14}, C_{02}, and C_{21} in Fig. 1 for example), ATs and RATs (except for those at chip-level inputs and outputs, which are given as constants) are removed. This is possible because RATs of one sub-block can be derived from ATs of sub-blocks that drive it. As an example, timing constraints of C_{14}, C_{02}, and C_{21} can be merged into

$$P - \left(\frac{K_{14}}{V_{DD,1}} + \frac{K_{02}}{V_{DD,0}} + \frac{K_{21}}{V_{DD,2}}\right) \geq 0,$$

where K_{ij} denotes K value of sub-block C_{ij} and $V_{DD,i}$ corresponds to the supply voltage of a block i that contains the sub-block (note that we control V_{DD} block by block, not sub-block wise.). If we merge timing constraints of C_{14} and C_{31}, that yields

$$P - \left(\frac{K_{14}}{V_{DD,1}} + \frac{K_{31}}{V_{DD,3}}\right) \geq 0.$$

Since we want to minimize the power consumption, our objective function for minimization can be expressed as

$$Power = \sum_i W_i V_{DD,i}^2, \quad (4)$$

where i is for all hierarchical blocks. The resulting formulation (a set of merged inequalities and objective function) can be turned into nonlinear optimization problem that can be solved with any optimization package [5].

Once we solve the inequalities and obtain V_{DD} of each block, we know the worst path delay of each sub-block through (3). This allows us to set up relations for ATs and RATs, which we solve for final slack distribution.

As an example, consider C_{14}, C_{02}, C_{21}, and C_{31} in Fig. 1, which are re-drawn in Fig. 4. We can derive the following relations from the figure:

$$\begin{aligned}
d(p_1) \leq RAT_1 &\leq RAT_2 - d(p_2) \\
RAT_1 + d(p_2) &\leq RAT_2 \leq P - d(p_4) \\
d(p_1) \leq RAT_1 &\leq P - d(p_3)
\end{aligned}$$

Any RATs can be selected as long as it satisfies the above relations.

[2]Set up and hold times of storage elements should be taken into account in the timing constraints, which we drop here for simplicity of notation.

[3]We assume the same phase tags for all timing assertions, thus the same clock for simplicity.

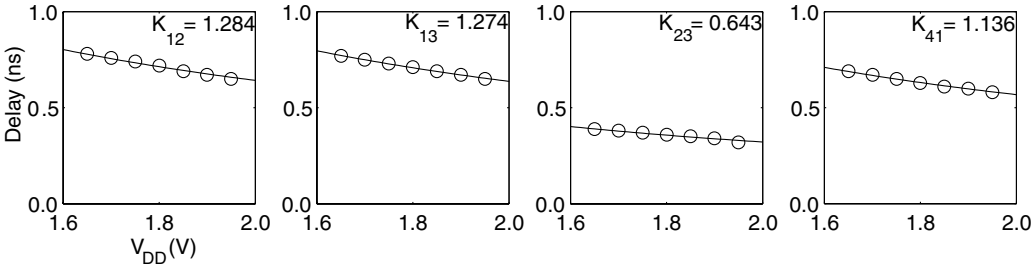

Fig. 3. Estimated K value curves.

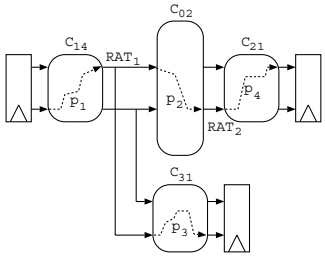

Fig. 4. Derivation of RATs.

TABLE I

CHARACTERISTICS OF EXAMPLE DESIGNS FOR EXPERIMENTS

Designs	# gates	# I/Os	V_{DD} (V)	Freq. (MHz)	Power (mW)
simple_spi	1600	28	1.90	556	10.81
idct	42130	28	1.90	250	159.11

TABLE II

RESULT OF POWER-AWARE SLACK DISTRIBUTION

Designs	V_{DD}s	Power (mW)	Power saving (%)
simple_spi	1.90, 1.65, 1.75	9.34	13.6
idct	1.90, 1.65, 1.90, 1.65	144.44	9.2

IV. EXPERIMENTAL RESULTS

To evaluate the efficiency of the proposed slack distribution method, we perform experiments for two example designs: enhanced version of the serial peripheral interface available on Motorola's MC68HC11 family of CPUs (simple_spi) from [6] and inverse discrete cosine transform (idct). The first example is a simple design, which we use to investigate the effects of our methodology in detail as well as for the experiment. The second one is a fairly complex design, which has a distributed arithmetic structure for high performance. Each design is described in Verilog, followed by functional simulation, and synthesized using Synopsys Design Compiler. The circuits are mapped onto a 0.18 μm gate library developed for an industrial CMOS process. TABLE I shows the characteristics of example designs. The second column shows the number of gates after logic synthesis. The number of inputs and outputs are shown in the third column. We initially fix V_{DD} at 1.9 V that leads to the target frequency as shown in the fifth column of TABLE I. The last column shows the power consumption when each design is powered by the single V_{DD} of 1.9 V. The V_{DD} of 1.9 V is the lowest operating voltage to obtain target frequencies. In this case, all RATs and ATs are optimal value to get that V_{DD} and those frequencies.

There are three hierarchical blocks and top-level glue logic in idct, meaning that for our PASD formulation we have four controllable V_{DD}s (three V_{DD}s for three hierarchical blocks plus one chip-level V_{DD}). There are three controllable V_{DD}s in case of simple_spi. The V_{DD}s for the technology we use range from 1.65 V to 1.95 V. For each design, K parameters are obtained following the steps in the last section, which is then followed by setting up a set of inequalities that describe timing constraints. The inequalities are solved through a nonlinear optimization package, which gives us the values of V_{DD}s as shown in the second column of TABLE II. In order to check the validity of the design with multiple V_{DD}s, we run the static timing analysis with each hierarchical block powered by its own V_{DD}.

V. CONCLUSION

The paper presents a new slack distribution methodology targeting power minimization. When we allow multiple supply voltages, we show that the approach can be formulated as nonlinear optimization problem, which can be solved very efficiently. Once the supply voltage of each hierarchical block is obtained, we can set up a set of relations that drives derivation of ATs and RATs.

REFERENCES

[1] Y.-H. Chan, P. Kudva, L. Lacey, G. Northrop, and T. Rosser, "Physical synthesis methodology for high performance microprocessors," in *Proc. Design Automat. Conf.*, Anaheim, Califonia, USA, June 2003, pp. 696–701.

[2] J. Hu, Y. Shin, N. Dhanwada, and R. Marculescu, "Architecting voltage islands in core-based System-on-a-Chip designs," in *Proc. Int'l Symposium on Low Power Electronics and Design*, Newport Beach, Califonia, USA, Aug. 2004, pp. 180–185.

[3] K. Choi and A. Chatterjee, "HA^2TSD: Hierarchical time slack distribution for ultra-low power CMOS VLSI," in *Proc. Int'l Symposium on Low Power Electronics and Design*, Monterey, Califonia, USA, Aug. 2002, pp. 207–212.

[4] T. Sakurai and A. R. Newton, "Alpha-power law mosfet model and its applications to cmos inverter delay and other formulas," *IEEE Journal of Solid-State Circuits*, vol. 25, no. 2, pp. 584–594, Apr. 1990.

[5] G. L. Nemhauser, A. H. G. R. Kan, and M. J. Todd, *Handbooks in Operations Research and Management Science: Optimization*. Amsterdam, Netherlands: Elsevier Science Publishers B.V., 1989.

[6] OPENCORES.ORG. (2004) OPENCORES. [Online]. Available: http://www.opencores.org