Enabling Adaptability Through Elastic Clocks

Emre Tuncer
Elastix Corporation
Los Gatos, CA, USA
emre@elastix-corp.com

Jordi Cortadella
Universitat Politècnica de Catalunya
Barcelona, Spain

Luciano Lavagno
Politecnico di Torino
Torino, Italy

ABSTRACT

Power and performance benefits of scaling are lost to worst case margins as uncertainty of device characteristics is increasing. Adaptive techniques can dynamically adjust the margins required to tolerate variability and recover a significant part of the benefits lost due to worst-case conditions. Additionally, the stringent timing requirements for the synthesis of low-skew clock trees involve higher power consumption, and limit the adaptability to varying operating conditions. This paper introduces an elastic clocking scheme as an adaptive technique to confront variability and provide substantial power savings by dynamically adjusting to operating conditions. The synthesis and sign-off analysis of the elastic clocks is fully automated. Changes to the design flow and sign-off analysis of elastic clocks are addressed by automation of design flow support.

Categories and Subject Descriptors

B.8.2 Performance Analysis and Design Aids.

General Terms

Design, Reliability, Economics.

Keywords

Adaptive voltage scaling, desynchronization, GALS, low power design.

1. INTRODUCTION

Increasing process variability and decreasing operating voltage, as feature sizes scale down, reduce potential power-performance gains. Statistical design methods can reduce overdesign due to unrealistic worst-case assumptions [1], [6]. Large volume parts can be binned and sold at different price points to recover some portion of performance versus yield trade-off. However, binning is not applicable to ASICs due to commercial reasons, and statistical timing analysis does not address the margins needed for environmental variations such as temperature and voltage changes.

DAC'09, July 26-31, 2009, San Francisco, California, USA

Ad-hoc recovery of design margins is common place in today's world. Off-the-shelf processor parts can be over-clocked well beyond their rated speeds by employing sophisticated cooling. By the same token, reducing the supply voltage to run a fast part at the specified frequency can save energy and power, which are becoming a primary concern for all electronic systems. Adaptive Voltage Scaling (AVS) provides this capability by sensing on-chip conditions dynamically and reducing or increasing the supply voltage to run the part at the required speed [2]. The power gains in AVS, however, are limited by the ability of predicting data path delays across the variation space [3].

AVS addresses static (process) or slowly varying (temperature, and to some degree aging) variations. The response time of the voltage regulation loop is usually hundreds or thousands of clock cycles. Cycle-to-cycle variations, such as IR-drop due to dynamic loads, must be handled by increasing the margins, thus reducing the achievable power gains.

Fine-grained application of AVS to individual cores or blocks in an SOC further improves power gains, based on load and performance requirements. However, the clock skew due to voltage domain crossing quickly becomes the limiting factor for performance, and increases the hold time fixing overhead. A solution to overcome this limitation is the adoption of asynchronous communication techniques between blocks [4]. The GALS (Globally Asynchronous Locally Synchronous) approach provides the flexibility to have each block driven by its own separate clock, and possibly supply voltage, while still enabling safe communication with other blocks. The main drawback of a GALS approach is the synchronization latency required to cross different clock domains, which may have a significant impact on the performance of the system.

Elastic clocks, where the period is dynamically adjusted to data path delays at the current operating conditions, provide the ability to minimize AVS margins due to IR-drop and clock skew. They also reduce latency in inter-block communication due to the asynchronous nature of the local clock controller protocol. Elastic clocks are implemented in a synchronous design flow through the desynchronization process.

2. DESYNCHRONIZATION

The separation between functionality and performance has always been a cornerstone of digital circuit design, enabling the development of tools that support functional specification, using snthesizable Verilog or VHDL; logic synthesis; physical design; equivalence checking and static timing analysis. Even testing schemes based on coupling of full stuck-at functional testing with limited at-speed performance testing, benefit from this separation.

Asynchronous circuits are interesting for the performance, power, modularity and Electro-Magnetic Interference properties. However, they so far suffered from a fundamental violation of the separation between functionality and performance, which required new tools and languages to be developed and learned, thus making their widespread adoption virtually impossible. In recent years, extensive work has been done on a local clock signal generation technique known as "desynchronization" [5], [9].

The desynchronization process is automated by a tool that essentially augments Clock Tree Synthesis in a standard design flow, by creating localized clocks, and synthesizing handshake logic between the asynchronous controllers that generate them, thus creating elastic clocks. This control layer contains matched delays to guarantee the appropriate synchronization timing (setup and hold) between all pairs of communicating registers.

Desynchronization retains the separation between functionality and timing throughout the design flow, because it actually does not modify the logic and registers (apart from some transformations of flip-flops into latch pairs). In fact, the elastic circuit as a result of desynchronization still works using the notion of "cycles", except that now the cycles are determined by local combinational logic delays (including the effects of PVT variations) and handshaking, rather than by an external PLL or synchronous clock reference. As such, it permits the re-use of IP blocks, design tools and design flows that were originally created for synchronous design, but provides the advantages of asynchronous circuit to a "traditional" designer.

3. ELASTIC CLOCKS

The control layer, and the corresponding delay elements that generate the clocking for different partitions of the circuit, configure the elastic clocks that dynamically adjust their frequency to the data path delays. The delay elements are located close to the associated data path blocks, increasing the spatial correlation between the data path and the delay elements. Hence, the margins required to cover the variability can be reduced. The control layer and the delay elements can immediately react to the cycle-to-cycle variations that affect the data path, such as IR-drop and voltage ripple, and can delay the clock pulses as needed. This guarantees that the circuit operates correctly in a wider range of static or dynamic variations than traditional synchronous AVS.

The delay lines are synthesized such that each pair of communicating registers is always clocked to ensure setup and hold times. The false paths are ignored, as in usual Static Timing Analysis, and do not contribute to the synthesis of the delay lines. Multi-cycle paths require special attention. During calculation of the longest delay, the delays of the multi-cycle paths are normalized with the multi-cycle factor, and they contribute based on their slack. As a simplified example, if the minimum slack path between a pair of registers is due to a multi-cycle path of factor 3, then the delay line is synthesized such that it has one third the delay of the multi-cycle path.

Elastic clocks are generated on a per clock domain basis, multiple clock domains are handled by desynchronizing each domain separately. In this scheme, asynchronous clock domains are handled without any extra steps. The communication between asynchronous clock domains is left untouched through insertion of elastic clocks. Synchronous clock domains require explicit handshake signals to control data transfer between them.

The area overhead of elastic clocks is minimized by transforming only the interface flip-flops into master/slave latches. Interface flip-flops are defined as the registers that receive input from other partitions. Internal flip-flops have logic only from their partition in their transitive-fanin. Enable trees are automatically created for the master latches. The slave latches are connected to the slave enable trees, which also clock the original flip-flops that are internal to the partitions. The overhead of the master enable trees is minimized by appropriately choosing the number and location of the required master latches. The control logic has a constant overhead of a few hundred gates per partition; hence the relative area increase depends on the complexity of the partition connections which is mirrored in the control layer. The overall area overhead for a partition size of two hundred thousand gates is about 2%.

Scan testing is supported with the elastic clocks. At the tester non-overlapping clocks applied to transformed master/slave latches, hence operating them synchronously. The non-overlapping clocks can be generated internally from the test clock or provided externally. Overall, the elastic clocks use the same scan-based methodology and ATPG flows as before, and no change is necessary for testing flows.

Clock gating is a widely used technique to reduce dynamic power. During desynchronization, existing clock gating circuits are copied to slave enable trees. If a gating element controls registers across multiple slave enables trees, the gating element is cloned. The data path delays include clock gating checks and matched delay lines are adjusted to represent clock gating signal arrival times, if they become critical.

4. DESIGN FLOW

The insertion of elastic clocks is done after placement of a gate-level netlist, just before clock tree synthesis (Figure 1). The circuit is first partitioned into a set of logic blocks and elastic clocks are created for each block. Handshake signals are inserted to synchronize elastic clocks that drive the registers of each communicating block. The period of each elastic clock is determined by the post-placement timing characteristics of the corresponding block.

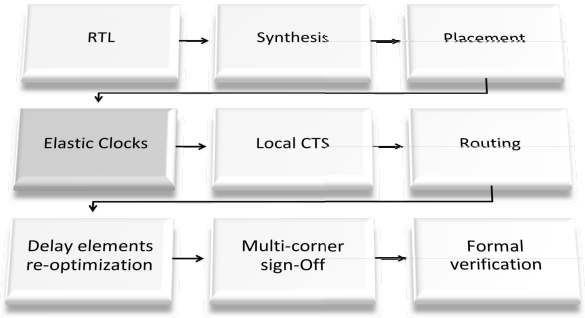

Figure 1. Elastic Clocks design flow

New timing constraints are generated automatically during the insertion of elastic clocks, to define new local clocks and constrain delay elements ("Delay" in Figure 2) for proper optimization. The delay elements have to match data path delays

at all sign-off corners. The margins between the delay elements and the corresponding data path delays can be optimized using statistical methods [7].

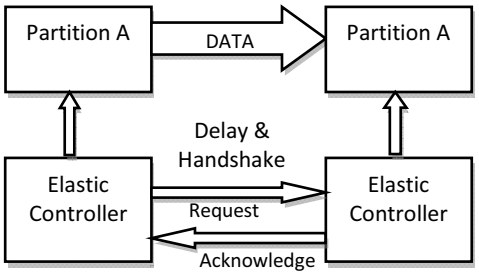

Figure 2. Elastic clock connections

During sign-off, explicit timing checks are done between the data path and the delay elements at all corners, to guarantee the correct clocking of the logic to meet the setup and hold conditions. In Figure 2, this corresponds to verifying that the data path delay between partitions A and B is always smaller than the delay between the associated controllers ("Delay") at all design corners, including the clock-tree insertion delays in each partition and the setup times of the receiving registers. These checks are performed using standard industrial sign-off tools by means of automatically generated scripts.

At the top level, each elastic block is connected to the desynchronized on-chip network asynchronously, by using the handshake signals. The handshake signals are used to "sense" the operating conditions and drive a voltage controller circuitry. This provides a unique opportunity to apply AVS where the sensing circuit is also an elastic clock generator, thus reducing the stringent clock skew requirements across multiple cores or blocks and reducing margins (e.g. due to cycle-to-cycle IR-drop). These signals are also used to control the voltage for a fine-grain AVS [8].

Figure 3 illustrates a connection scheme, where the clock controllers generate the elastic clocks for individual blocks and provide handshake signals ("HS") to the voltage controller and the asynchronous on-chip interconnect. The voltage controller drives a voltage regulator module to dynamically adjust the voltage, according to the relative timing of the handshake signals between the interconnect network and the elastic block. A purely asynchronous communication between the elastic blocks and the on-chip interconnect significantly reduces the communication latency.

5. CONCLUSION

An AVS scheme using elastic clocks eliminates stringent clock skew requirements across multiple cores and blocks, and reduces margins due to cycle-to-cycle variations. The asynchronous nature of elastic clocks avoids the latency penalty introduced by GALS schemes, and makes fine grain voltage scaling a possibility without performance overhead.

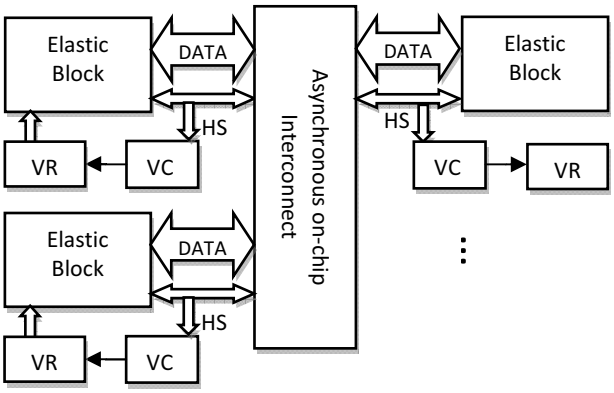

Figure 3. Scheme for voltage regulation

6. REFERENCES

[1] Visweswariah, C., "Statistical analysis and optimization in the presence of gate and interconnect delay variations," in *Proceeding of the 2006 international Workshop on System-Level interconnect Prediction*. Munich, Germany, March 04-05, 2006.

[2] Dhar, S., Maksimović, D., and Kranzen, B. Closed-loop adaptive voltage scaling controller for standard-cell ASICs. In *Proceedings of the 2002 international Symposium on Low Power Electronics and Design* (Monterey, California, USA, August 12 - 14, 2002). ISLPED'02. ACM, New York, NY, 103-107.

[3] Elgebaly, M. and Sachdev, M. Variation-aware adaptive voltage scaling system. *IEEE Trans. Very Large Scale Integr. Syst.* 15, 5 (May. 2007), 560-571.

[4] Iyer, A. and Marculescu, D. Power and performance evaluation of globally asynchronous locally synchronous processors. In *Proceedings of the 29th Annual international Symposium on Computer Architecture* (Anchorage, Alaska, May 25 - 29, 2002). International Symposium on Computer Architecture. IEEE Computer Society, Washington, DC, 158-168.

[5] J. Cortadella, A. Kondratyev, L. Lavagno, and C. Sotiriou, "De-synchronization: synthesis of asynchronous circuits from synchronous specifications," *IEEE Transactions on Computer-Aided Design*, vol. 25, no. 10, pp. 1904--1921, Oct. 2006.

[6] Cao, Y. and Clark, L. T. Mapping statistical process variations toward circuit performance variability: an analytical modeling approach. In *Proceedings of the 42nd Annual Conference on Design Automation* (Anaheim, California, USA, June 13 - 17, 2005). DAC'05. ACM, New York, NY, 658-663.

[7] Liu, Q. and Sapatnekar, S. S. Synthesizing a representative critical path for post-silicon delay prediction. In *Proceedings of the 2009 international Symposium on Physical Design* (San Diego, California, USA, March 29 - April 01, 2009). ISPD '09. ACM, New York, NY, 183-190.

[8] J. Cortadella, V. Singhal, E. Tuncer, and L. Lavagno. A variability-aware scheme for high-performance asynchronous circuit voltage regulation. US patent application. November 2008

[9] Andrikos, N., Lavagno, L., Pandini, D., and Sotiriou, C. P. A fully-automated desynchronization flow for synchronous circuits. In Proceedings of the 44th Annual Conference on Design Automation (San Diego, California, June 04 - 08, 2007). DAC '07. ACM, New York, NY, 982-985.