Dynamic Optimization of System Performance via Integrated Testbench

Nathan Monroe

I. INTRODUCTION

With decreasing feature size in CMOS devices, process variation is becoming an increasing concern in device fabrication. Process variation results in devices either being discarded as yield losses or operated below maximum possible performance[1]. There is need for the ability for a device to detect which process corner it is and accordingly calibrate its operating conditions, enabling maximum performance in the presence of process variations. The ring oscillator method[2] uses an on-chip RO's frequency as a proxy for process variation. However, this is not an effective proxy for process corner, being subject to same-die process variations and hot-electron effects. In addition, it incurs significant area and power costs. The Razor method attempts to solve this with a shadow register to detect a circuit's faulty operation, allowing optimized DVS[3]. However metastability, noise susceptibility, hold violations, increased power consumption are concerns, along with implementation cost due to a precisely delayed clock (source). Proposed here are two implementations of an integrated testbench for a system to self-assess critical path functionality independent of process variations, allowing a system to optimize operation parameters such as VDD, Fclk, or backgate bias. The two proposals are implemented for demonstration in 45nm CMOS in a 16b ripple carry adder, used in a DVS implementation at Fclk=200MHz.

II. LOOKUP TABLE METHOD

In the LUT method, upon startup a worst-case input vector is applied to a critical logic path- both predetermined by design and hardcoded into the logic (see fig. 1). The slowest output bit is compared to the expected output (based on WC input, predetermined by design and hardcoded). It is assumed that successful computation of the slowest output bit implies successful computation of all other output bits. If comparison is successful at the end of the clock cycle, then VDD is lowered by a fixed amount dv. If comparison fails, then VDD is raised by a fixed amount dv. This process is repeated iteratively until minimum VDD for functional operation is found, given Fclk and other system parameters. This will converge successfully independent of process variation. This method was implemented in SPICE using behavioral vdd sources (ideal voltage step). As seen in fig. 2, VDD converges to optimal value in the presence of process variations in this ten run Monte Carlo simulation. Selection of dv is a tradeoff between convergence time and granularity of optimal vdd. This has minimal area overhead- n muxes plus comparison logic. The inclusion of comparison logic slightly increases Tp of critical path logic due to increased capacitive loading on slowest bit. In addition, finite delay of comparison logic provides a small amount of margin from being on the absolute edge of functionality. Area cost of LUT implementation consists of a single xor gate, n muxes to input worst-case vector, scaling with n bits. Complexity also slightly increases in implementation of power supply control circuitry.

III. PID CONTROL METHOD

A PID control method improves upon the lookup table's convergence time and granularity. This is of particular utility in DVS implementations, where low bandwidth of power supplies can result in long convergence times for optimal vdd. In repeating arithmetic structures which are common critical paths, successive output bits have equal delays and thus observation of these signals can be used to track propagation of the critical path and determine negative timing margin. As seen in fig. 4, delay lines are added, driven by the last output bit. The delay lines are implemented as buffers, designed with delay equal to successive output bits in the critical path logic. Observation of delayed output bits provides information on positive timing margin. Information on positive/negative timing margin is used in a PID control, to quickly converge on minimum VDD for functional operation. For example, consider CLK_A (fig. 3). The next cycle begins before successful computation of S[13] and S[15], indicating that a significant increase in vdd is required to meet timing constraints. Conversely, CLK_C arrives after successful output computation plus 2 delayed versions of S[15], indicating significant positive timing margin and that vdd can be significantly lowered to save power while maintaining timing constraints given Fclk. CLK_B arrives at the optimal time, where the critical computation has been completed with little timing margin and therefore vdd is at it's optimal value given process variations, worst case-input and logic path, and desired Fclk.

In addition to improved convergence time, PID control can achieve finer granularity on VDD with minimal convergence time. In this implementation, negative margin is determined from output bits s[9,11,13,15], and four delay lines are used to determine positive margin, each delay line designed to have Tp equal to two successive output bits. This method was implemented in SPICE using behavioral vdd sources (ideal voltage step). As seen in fig. 5, VDD converges to optimal value rapidly in the presence of process variations in this ten run Monte Carlo simulation.

In general, this PID approach has increased area cost and critical path loading, in exchange for improved convergence time. There is a design tradeoff- adding comparison logic to more output bits of critical path increases granularity of the feedback loop, improving convergence time at the cost of added area and capacitive loading on critical path outputs. On the input side, area cost scales with n muxes, one per input bit. On the output side, area cost consists of one xor gate and delay-sized buffer per PID loop input, with increasing PID inputs improving convergence time to optimal value.

IV. CONCLUSION

Two integrated testbenches have been proposed for a system to optimize performance in the presence of process variations. The LUT method provides information solely if a critical path is meeting timing requirements. This comes at an area cost similar to the Razor method, while eliminating the need for a delayed clock. The PID control method provides a measure of by how much timing requirements are/are not being met, enabling a controller to improve convergence time and granularity at the expense of increased area cost and loading of critical path compared to the Razor method. Both methods are of utility in systems with one or few critical paths. The PID method is of extra utility in the case of DVS optimization, where VDD convergence time dominates. Both methods have been demonstrated in a DVS optimization, but could be easily modified for optimization of Fclk, backgate bias, or other parameters for desired device performance. The testbenches have been demonstrated for optimization at startup, however the implementation could be extended for dynamic optimization while changing operation modes during system operation. This allows devices to be pushed to maximum performance and minimize conservative engineering design margins in the presence of process variation.

Acknowledgements:

The author gratefully acknowledges Prof V. Sze and A. Suleiman for helpful discussion. The author also acknowledges the Dr. Pepper Company for supplying the necessary caffeine to enable this work.

References:

1. Chandrakasan et al. "Digital Integrated Circuits: A Design Perspective". Prentice Hall, 2003.
2. Das et al. Within-die gate delay variability measurement using re-configurable ring oscillator". IEEE Semi. Man. pp. 256-267, 2009.
3. Ernst et al. "Razor: Circuit-Level Correction of Timing Errors for Low-Power Operation". Micro. IEEE pp.10-20, 2004.

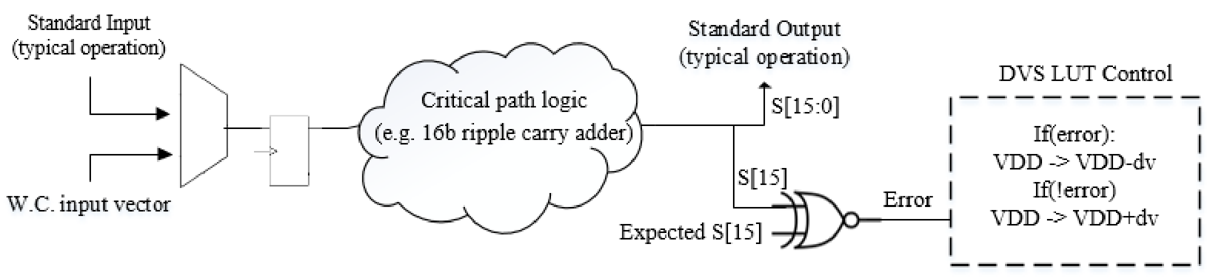

Figure 1: System block diagram: Lookup Table Method

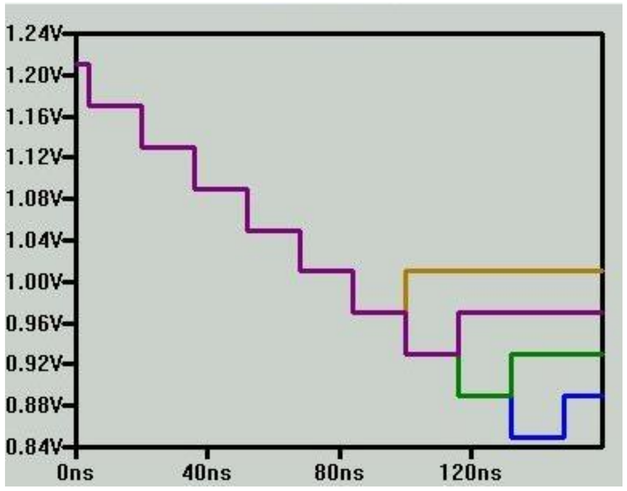

Figure 2: LUT Control Method Monte Carlo DVS convergence

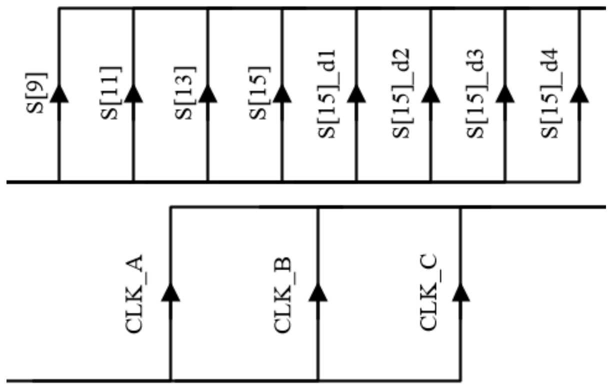

Figure 3: Example timing for PID Control Method

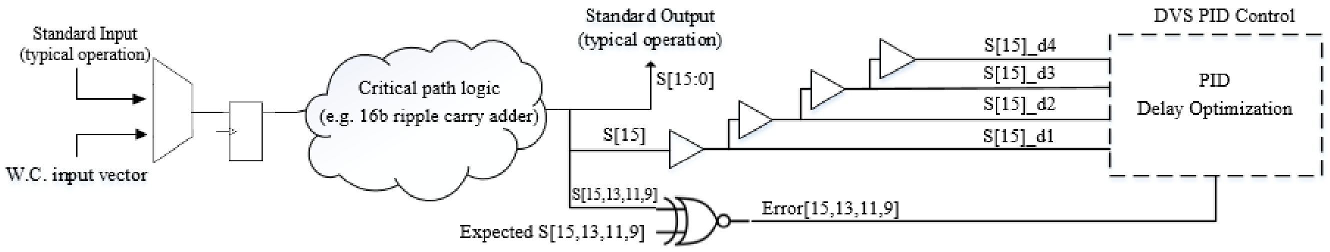

Figure 4: System block diagram: PID Control Method

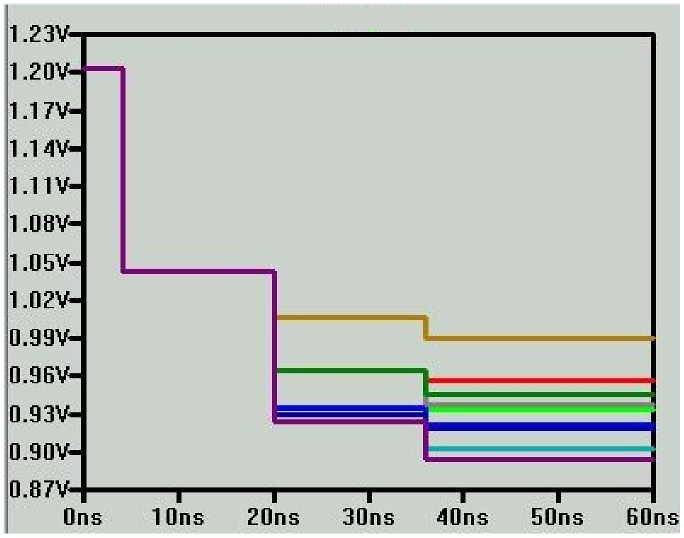

Figure 5: PID Control Method Monte Carlo DVS convergence