Addressing Design Margins through Error-tolerant Circuits

Shidhartha Das[1], David Blaauw[2], David Bull[1], Krisztián Flautner[1] and Rob Aitken[1]
[1]ARM Ltd., Cambridge, U.K
[2]Department of EECS, University of Michigan, Ann Arbor, U.S.A

ABSTRACT

We review adaptive design techniques with particular emphasis on error-tolerant techniques. We compare and contrast traditional adaptive approaches with error-tolerant techniques and analyze the margins eliminated by each of them. We discuss the applications of the latter to on-chip communication and signal-processing. Finally, we focus on a specific example of an error-tolerant technique for general-purpose computing called Razor.

I. INTRODUCTION

Increasingly, design margins are required to address rising PVT variations leading to substantial power and performance losses. Adaptive techniques mitigate the impact of margins by dynamically tuning circuit parameters to compensate for variations. However, such techniques cannot track localized transients, the margins for which can be significant, especially at advanced process nodes. This has motivated recent research efforts into, so-called, error-tolerant approaches for dynamic compensation. In such techniques, intermittent timing errors during circuit operation are detected and recovered from. Allowing circuits to fail enables elimination of worst-case margins leading to significant improvements in energy-efficiency and performance.

II. CATEGORIZING VARIATIONS

At smaller geometries, inter- and intra-die process variations worsen due to inherent limitations in accurately controlling the manufacturing process. Environmental uncertainties such as power supply droops, temperature hot-spots, coupling noise and clock jitter as well as transistor ageing contribute to performance variations of transistors. Variations can be classified as local (e.g. temperature hot-spots) or global (e.g. ambient temperature fluctuations) based on their spatial reach. Depending on their temporal rate of change, variations can be categorized as slow-changing (e.g. process variations and ageing effects) or fast-changing (coupling noise). Compensating for such variations requires operating at higher voltages or lower frequencies to account for unforeseen slow-down caused due to worst-case PVT variations. This process of margining ensures correctness at the expense of higher power and performance impact.

III. TRADITIONAL ADAPTIVE TECHNIQUES

Instead of operating at a single operating point, adaptive techniques tune voltage and frequency as determined by silicon and operating conditions. The traditional adaptive design approaches [1-4], or the "always-correct" techniques, use a pre-characterized look-up table or "canary" circuits to predict the failure limit of a chip and operate the system close to the predicted limit.

A. Look-up table based approach

In the look-up table based approach, the processor is pre-frequency for a given supply voltage.

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

The safe voltage-frequency pairs are obtained by performing conventional timing analysis on the processor. Typically, the operating frequency is decided based on the deadline under which a given computational task needs to be completed. Accordingly, the supply voltage corresponding to the frequency requirement is "dialed in". The table look-up approach exploits periods of low CPU utilization by dynamically scaling voltage and frequency, thereby leading to energy savings. However, its reliance on conventional timing analysis performed at the combination of worst-case process, voltage and temperature corners implies that none of the safety margins are eliminated at a particular operating point.

B. Canary-circuits based approach

An alternative approach relies on the use of the so-called "canary" circuits to predict the failure point [1-4]. Canary circuits are typically implemented as delay chains which approximate the critical path of the processor. Voltage and frequency are scaled to the extent that this replica-delay path fails to meet timing. The replica-path tracks the critical-path delay across inter-die process variations and global fluctuations in supply voltage and temperature, thereby eliminating margins due to global PVT variations. However, the on-die location of the critical-path and its replica differs. Consequently, margins are added to the replica-path in order to budget for delay mismatches due to intra-die process and local variations in temperature and supply voltage. Margins are also required to address fast-changing transient effects, such as coupling noise, which are difficult to respond to in time using this approach. Furthermore, mismatches in the scaling characteristics of both paths require additional safety margins. These margins ensure that the processor still operates correctly even at the point of failure of the replica-path.

IV. ERROR-TOLERANT TECHNIQUES

As process technology scales, the local variations worsen thereby undermining the efficacy of traditional adaptive techniques. "Error-tolerant" techniques address local variations by scaling voltage and frequency till the point where the processor incurs timing errors. Error-detection circuits flag such an occurrence and engage a recovery mechanism to restore correct state. This eliminates all worst-case safety margins and enables significantly improved performance and energy efficiency over the traditional techniques. Their relative complexity makes the general applicability of such systems difficult. However, they are naturally amenable for communications and signal-processing where existing mechanisms can be overloaded to detect and correct timing errors.

Worm et al. [5] apply this concept to self-calibrating on-chip interconnects wherein bit-transfers occur at voltages below the safe

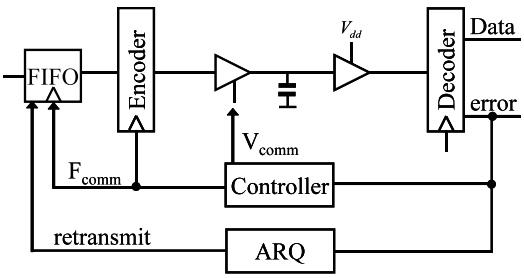

Figure 1 Self-calibrating interconnects

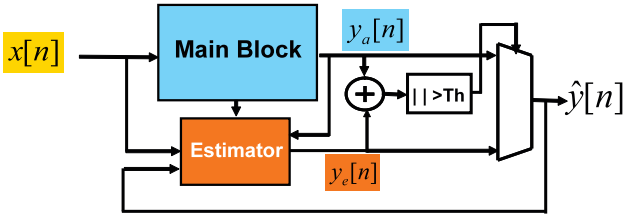

Figure 2 Algorithmic Noise Tolerance

limit. This enables bit-transfers at the lowest possible operating voltage while still guaranteeing the required performance and the targeted Bit Error Rate (BER). Error-detection occurs by encoding data words with so-called self synchronizing codes before transmission. The receiver is augmented with a checker unit that decodes the received code word and flags timing errors. Correction occurs by requesting re-transmission, as shown in figure 2. Furthermore, an additional controller obtains feedback from the checker and accordingly adjusts the voltage and the frequency of the transmission. By reacting to the error-rates, the controller is able to adapt to the operating conditions and thus eliminate worst-case safety margins. This improves the energy efficiency of the on-chip busses with negligible BER degradation.

Algorithmic Noise Tolerance [6] by Shanbhag et al. uses voltage-overscaling to significantly reduce energy consumption of processing blocks such as FIR filters while incurring intermittent timing errors. The main block is voltage scaled beyond the point of failure. Error-detection occurs by comparing the output of the main processor block against an estimator block which computes correct result based on previous history. Error-correction occurs by overwriting the result of the main block with that of the estimator block. The estimator block is significantly cheaper in terms of area and power as compared to the main block which is being voltage-scaled. At low error-rates, the benefits of aggressive scaling on the main block compensates for the overhead of correction, leading to significant energy savings.

V. Error-Tolerant Techniques for General-Purpose Computing

In general-purpose computing, timing errors should not corrupt the

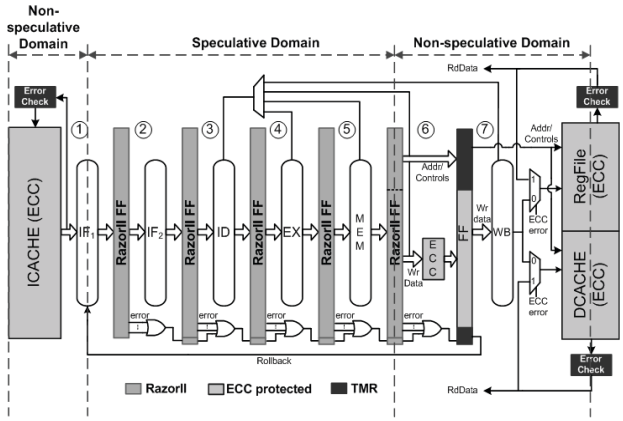

Figure 3 RazorII processor pipeline diagram

committed architectural state. We proposed Razor [7] as the first "error-tolerant" adaptive technique applied to general-purpose computing. Razor uses a delay-error tolerant flip-flop which detects timing errors by flagging spurious transitions on critical-path endpoints [8]. Recovery is achieved through a conventional architectural replay mechanism. This enables the supply voltage to be scaled to the point of first failure (PoFF) of a die for a given frequency. Thus, all margins due to global and local PVT variations

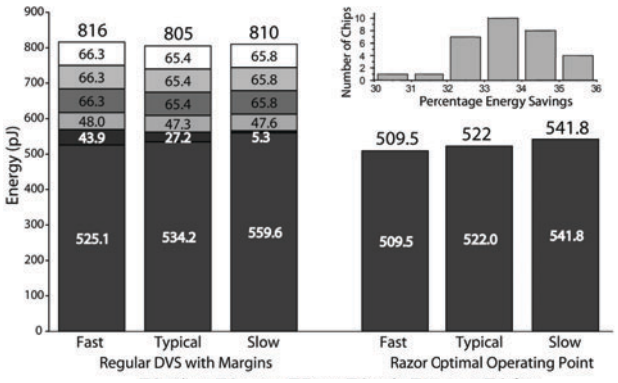

Figure 4 RazorII energy savings

are eliminated, resulting in significant energy savings. In addition, the supply voltage can be scaled even lower than the PoFF into the sub-critical region, deliberately tolerating a targeted error rate. Razor error-detection also enables tolerance to logic and register SER.

We designed and implemented a Razor-enabled 64-bit processor implemented in 0.13µm technology. The architecture (Figure 3) is divided into a pipeline with speculative state protected using RazorII flip-flops (FF), and a non-speculative memory and register file protected by ECC or triple-module redundancy (TMR). The 7th stage was designed to be non-timing critical to stabilize the pipeline state. In the event of an error, the pipeline is flushed and the failing instruction is re-executed. In case of repeatedly failing instructions, the error controller switches the clock frequency by half for 8 cycles. Figure 4 shows the measured energy dissipation for 3 die when operating at 0.04% error rate. Gains were 33.1 to 37.5% compared to the energy when the supply voltage is elevated to ensure correct operation for all 31 fabricated die at 85C with 10% margin for wearout, supply fluctuation and safety.

Bowman et al. [9] developed a similar approach for high-performance chips. Instead of keeping frequency fixed and reducing the supply voltage, they keep the supply voltage constant and use error-detection to improve throughput by 25-32% at the same operating voltage.

VI. Conclusion

In this paper, we compared traditional adaptive techniques with error-tolerant techniques and discussed the margins eliminated by them. We discussed a specific "error-tolerant" approach for general-purpose computing called Razor.

References

[1] A. Drake, et al., ISSCC, 2007
[2] T. Burd, et al. ,JSSC, Vol. 35, No. 11, 2000
[3] M. Nakai, et al., JSSC, Vol. 40, No. 1, 2005
[4] K. Nowka, et al., JSSC, Vol. 37, No. 11, 2002
[5] Worm et al., *TVLSI*, Vol. 13, No. 1, January 2005.
[6] R. Hegde and N. R. Shanbhag, *JSSC,* Vol.39, No. 2, February 2004.
[7] S. Das, et al., JSSC, Vol. 41, No. 4, 2006
[8] Blaauw et al., ISSCC 2008
[9] Bowman et al., ISSCC 2008