Scalable Stochastic Processors

Sriram Narayanan, John Sartori, Rakesh Kumar, and Douglas L. Jones
Department of Electrical and Computer Engineering, UIUC, 1308 West Main St., Urbana, IL 61801
Email:{spnaraya, sartori2, rakeshk, and dl-jones}@illinois.edu

***Abstract*—Future microprocessors increasingly rely on an unreliable CMOS fabric due to aggressive scaling of voltage and frequency, and shrinking design margins. Fortunately, many emerging applications can tolerate computational errors caused by hardware unreliabilities, at least during certain execution intervals. In this paper, we propose *scalable stochastic processors*, a computing platform for error-tolerant applications that is able to scale gracefully according to performance demands and power constraints while producing outputs that are, in the worst case, *stochastically correct*. Scalability is achieved by exposing to the application layer multiple functional units that differ in their architecture but share functionality. A mobile video encoding application here is able to achieve the lowest power consumption at any bitrate demand by dynamically switching between functional-unit architectures.**

I. Introduction

The emergent *ubiquitous computing* paradigm promises new applications in environmental monitoring, automation, and health care. For these new applications to be practical, the computing platform needs to offer high performance while operating within a very limited power budget (often micro- or even nano-watts). While technology scaling driven by Moore's law has offered continued reduction in power consumption and size, recent projections from the ITRS roadmap suggest that this scaling trend alone will not be sufficient to meet the demands of these future applications [1].

Conventional dynamic voltage/frequency scaling techniques [2] are necessarily limited, however, by the particular path-delay characteristics of the underlying architecture. In the present-day design flow, architectural choices for the various functional units (FUs) are made not with an intent to allow voltage- or frequency-scaling but to minimize power and area for operation at the nominal voltage/frequency. In traditional high-performance designs, all timing paths are tuned to match the length of the critical path. An implication of this design style is that when one timing path fails, a large number of other timing paths fail, since all path lengths are bunched around the critical path length [3]. This prevents effective deployment of hardware-based error tolerance mechanisms [4]. This suggests that computational platforms need to be designed from the ground-up to allow aggressive scaling.

Fortunately, a large class of emerging applications can tolerate a small number of timing errors in computations. Research efforts in [5], [6] exploit inherent error-tolerance of some applications. While these approaches have been shown to overcome hardware errors, they impose a nontrivial overhead when error-rates are low or zero. For instance, if there are execution phases when the system demands the maximum reliability, then it is desirable that the architecture scales to meet these demands.

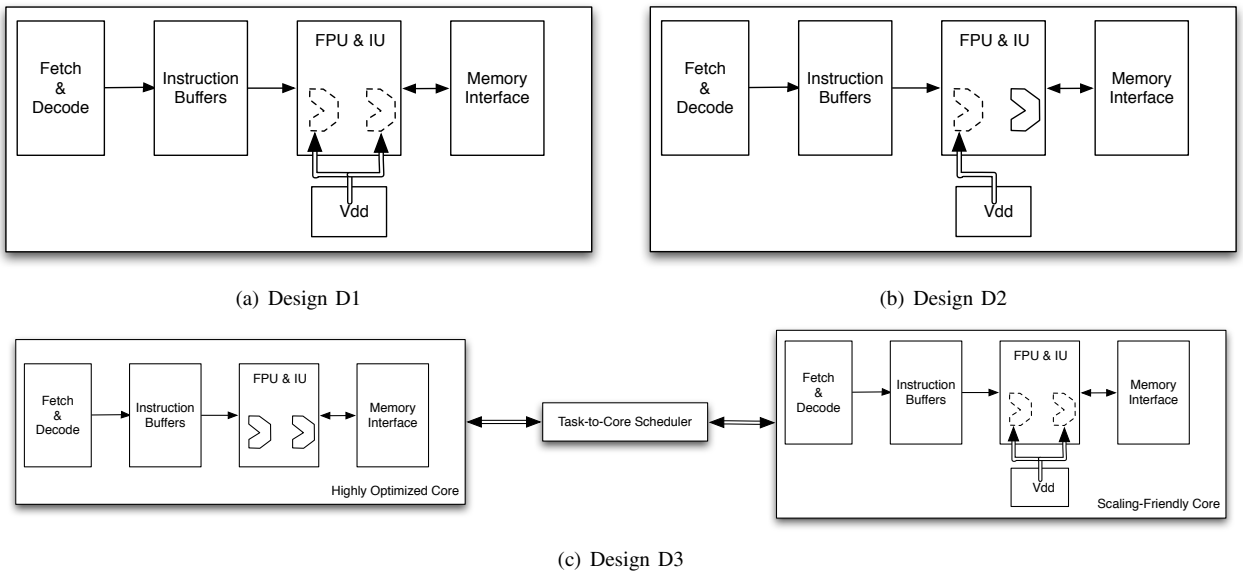
With the above in mind, we propose *scalable stochastic processors* [7]–[10], a computing platform for error-tolerant applications that is able to scale gracefully according to performance demands and power constraints while producing outputs that are, in the worst case, *stochastically correct*. Our proposed architecture gains power savings by exposing to the application multiple functionally equivalent units that exhibit several levels of reliability. This processor *scales* to changing application demands/constraints by dynamically switching between multiple functional units. We choose a video encoding application as an example to showcase the benefits of this design.

II. Scalable Architectures

Scalability can be achieved by replacing or supplementing traditional functional-units with gracefully degrading units. Timing-error-prone functional units may be incorporated into present-day systems at three broad levels listed below.

1) **D1. Fixed**: In this design, the baseline architecture is modified by replacing one or more functional units with an alternatively designed functional unit that is more conducive to voltage/frequency scaling. As shown in Figure 1(a), the execution unit consists of voltage scaling-friendly blocks (dotted lines) and is able to lower computation accuracy gradually with supply voltage. This design is suitable for applications that never demand maximum reliability, but impose a very limited power budget. This design point represents the least change to existing instruction set architecture (ISA) and programming models. But the ability to scale comes at the cost of compromising best power/performance when such scalability is not desired.
2) **D2. FU selectable**: In this design, the baseline processor is equipped with two different functional unit architectures. The application may choose to switch between the two functional units such that the overscaling range is extended. The execution unit in Figure 1(b) contains two types of logic blocks – traditional performance optimized version (solid lines) and an alternative design

This research is supported by the Gigascale System Research Center (GSRC), one of five research centers funded under the Focus Center Research Program (FCRP), a Semiconductor Research Corporation program, Texas Instruments, Inc., the National Science Foundation, Intel, and an Arnold O Beckman Research Award.



(a) Design D1

(b) Design D2

(c) Design D3

Fig. 1. The scalable architecture introduces alternative functional units at three levels. In (a), all the functional units of the core are replaced by scaling friendly versions; (b) shows two different FU architectures that can be selectively used; and (c) shows a reliability-defined heterogeneous multicore system.

that is friendly to voltage scaling (dotted lines). This design is suitable for applications that have time-varying power/performance demands. We envision a modified ISA that allows the application layers to choose particular functional units. Reliability requirements of applications can be annotated in software, and these annotations can be used to select the appropriate functional unit for a program or program phase. The current reliability target can be used to control the select lines of a MUX that routes an instruction through the most power efficient module. Since tuning a module for a specific error rate requires voltage scaling, module switching incurs overhead time for voltage scaling when the module must achieve different reliability targets within the same program.

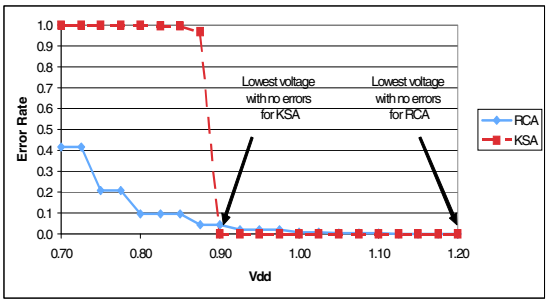
3) **D3. Core selectable**: This design consists of a multicore system where each core possesses a different architecture for the functional units. Figure 1(c) illustrates such a dual-core design along with a task-to-core scheduler that is responsible for assigning tasks according to their reliability requirement. Unlike other multicore designs such as [6], the scalable cores of this design can dynamically be made error-free by adjusting the supply voltage or clock frequency. This design is suitable for applications which can be decomposed into subcomputations that have different power/performance demands. This design may include design D2 if each core has selectable FU architectures. These cores may or may not share a common ISA. This design is in contrast to systems such as [11] where error-prone cores are avoided or healed, our system exploits them for power savings.

For a class of embedded applications that are data-dominated it is common for the execution units to significantly contribute to the total power dissipation. For an audio decoding benchmark, in the Philips TM3270 media processor [12] the execute module consumes around 0.255 mW/MHz out of a total processor power consumption of 0.935mW/MHz (a 27% contribution). We restrict our power-reduction design techniques to this class of processors.

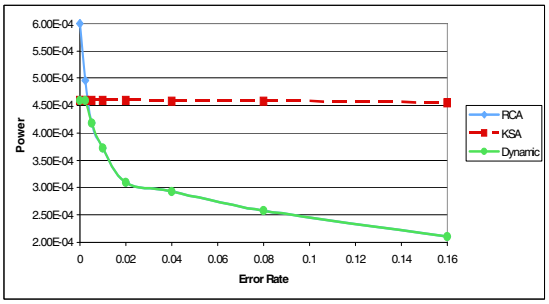
III. Functional-Unit Architectures

To characterize their power/reliability characteristics, functional units are synthesized in the IBM9SF 90nm CMOS technology with Synopsys DesignCompiler [13], and layout is performed in Cadence SoC Encounter [14]. To measure power and error rate across a range of voltages, we use voltage-specific Synopsys Liberty (.lib) files prepared with Cadence SignalStorm [15]. To obtain an accurate characterization of module behavior, we perform gate-level simulations using an input set of 180K random input samples.

As instances of different functional unit architectures, we consider the Kogge-Stone adder (KSA), which is highly optimized and fails catastrophically with voltage overscaling, and the ripple-carry adder (RCA), which fails gracefully but is much slower than the KSA. Figure 2(a) shows how the error rate of each adder architecture varies as voltage is scaled down. The KSA can be scaled to a much lower voltage (0.9V compared to 1.2V for the RCA) before producing errors. However, once errors occur, the adder fails catastrophically. On the other hand, the error rate of the RCA increases gradually as voltage is scaled down. However, the onset of erroneous behavior is much earlier than in the KSA so that a conservative voltage must be chosen to guarantee fidelity of the output. Because of these failure characteristics, the functionally equivalent modules have very different power/reliability characteristics.



(a) Error Rate vs. Voltage



(b) Power vs. Error Rate (no correction)

Fig. 2. The RCA allows power/reliability trade offs so that power is reduced as error rate is allowed to increase. The KSA, on the other hand, consumes less power for reliable operation, but does not allow power/reliability trade offs.

Figure 2(b) compares the power consumption of the adders at different error rates.

For reliable operation (0% error rate), the KSA consumes 25% less power than the RCA. This is because for the same frequency, voltage on the KSA can be scaled down to save power, while scaling down voltage on the RCA would cause timing errors. However, power/reliability trade offs are not possible for the KSA, since reducing voltage past the critical point causes massive failure, so power consumption is the same for all error rates. While the RCA is less efficient when operating completely reliably, its gradual failure characteristic allows reliability to be traded for power savings, making RCA favorable for noisy environments. For all non-zero error rates, the RCA consumes less power than the highly optimized KSA.

As the error rate increases, gate-level error-recovery mechanisms (*e.g.*, [16]) that exploit gracefully degrading architectures suffer recovery overhead that dominates power savings achieved through voltage scaling. The trend in Figure 3 suggests that gate-level techniques that seek to correct every instance of hardware errors may be inefficient in comparison with system-level approaches that do not correct every error instance and allow some errors to be masked. An architecture that allows instructions to be routed to the optimal module for a given system-level error rate can achieve benefits over a static module selection.

IV. Stochastic Applications

The scalable architecture developed in this work targets aggressive power reduction for a class of *stochastic* applications,

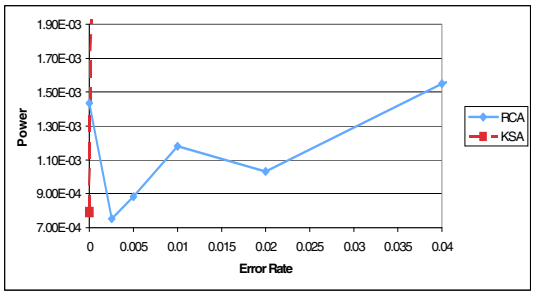


Fig. 3. Razor error recovery can provide some power savings for gracefully failing designs (RCA) after the point of first error. However, these benefits are limited, since only a small number of errors can be gainfully tolerated before recovery overhead outweighs voltage scaling power reduction. Note that the quantity on the X-axis is the error-rate prior to recovery.

i.e., applications that can operate with *a priori* known reliability requirements. The reliability requirement may change with time and execution phases within the application. Multimedia applications are typical examples of stochastic applications. Here, the input (such as a feed from a camera or a microphone) is already contaminated by measurement noise. Since these applications are increasingly implemented on fixed-point mobile platforms, quantization poses another source of noise. Furthermore, the output in these applications needs only to meet the fidelity discernible by the human sensory acuity. The scalable architectures described in this work can offer significant power/throughput gains to these applications, if we treat computational errors as a new source of noise.

As a particular example, we showcase the advantages of a scalable architecture to the popular H.264 video encoding application. The high compression efficiency of this new video encoding standard has enabled exciting applications in wireless video communication and is increasingly implemented on battery-constrained mobile devices. An important subsystem of the video encoder, the motion estimation, is often reported as contributing around 40% to 50% of the total encoder power consumption on ASIC implementations [17]. The main computational kernel of the motion estimation engine is the sum of absolute difference (SAD) that computes $|A - B|$ for two inputs A and B. We seek to gain power savings through voltage overscaling while allowing any resulting timing errors. A computational error in the motion estimation engine simply results in poorer encoding efficiency, that is a larger *bitrate*. Such errors could result in non-zero motion vectors even if the current and reference frames are identical (*i.e.*, there was no motion in the video sequence). This, in turn, will adversely impact the power overhead of wireless communication. Consequently, during periods of relative inactivity in the input video sequence or favorable wireless channel conditions, the motion estimation block may contain some slack that can be exploited for power reduction. Since models describing wireless communication overhead are beyond the scope of this paper, we will use the bitrate as a measure of performance. By controlling the occurrence of timing errors in motion estimation, a scalable processor can trade off bitrate

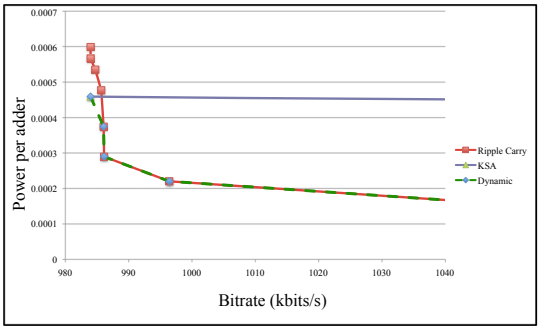


Fig. 4. The RCA is able to significantly lower the power consumed (per adder) without compromising bitrate of the output. But the KSA is able to offer around 20% lower power consumption when no bitrate degradation can be allowed

for power reduction. But for this approach to work, the bitrate must worsen gradually with scaling supply voltage or clock frequency.

In order to study the impact of voltage overscaling on compression efficiency of the H.264 video encoder, we used a PC-implementation of the JM reference software [18]. The experimental setup described in Section III was used to obtain the probabilities of bit-error for a 16-bit word-length. This probability model was used to inject errors into the motion-estimation block of the JM reference software. Similar probabilistic models for bit-errors caused due to voltage overscaling have also been developed by other researchers [19].

We used three frames of a QCIF format video source in 4:2:0 YUV format as our input. If there is demand for the lowest achievable bitrate the application chooses the KSA and is able to consume around 20% less power (when compared with the lowest power option for the RSA that offers this best bitrate). If the application is able to tolerate some worsening of the bitrate, then it switches to the RCA. A small increase in bitrate of around 12kbits/s (*i.e.*, a 1.2% loss) is able to reduce the power consumption by around 60%. The scalable architecture that is able to switch between the FU architectures at runtime is able to maintain optimal power consumption at all levels of bitrate demand as shown in Figure 4.

As an example implementation, consider the design D2. The scalable processor will receive input from the wireless subsystem (responsible for packetizing and communicating encoded video data) regarding the quality of the communication channel. Under adverse channel quality conditions, the processor will use the KSA adders by issuing the correspondingly annotated instructions. Under more favorable channel conditions, it will issue the instructions annotated to use the RCA adders. The processor will then proceed to lower the supply voltage according to channel information received from the wireless subsystem. By constantly adapting the issued instructions to changing wireless channel quality, this scalable architecture maintains lowest possible power consumption.

V. Conclusion

Many emerging applications can tolerate a small number of computational errors at least during some execution phases. The scalable stochastic processor architectures presented in this work expose multiple alternative designs of functional units to the application and thereby allow acceptable voltage/reliability tradeoffs over a wide range of voltages. The scalability leads to 20% to 60% power savings in the motion estimation block of a mobile video communication application. Our future work includes re-designing other components of a processor to allow voltage/reliability tradeoffs.

References

[1] ITRS, "ITRS 2008 update," ITRS, Tech. Rep., 2008.
[2] S. Herbert and D. Marculescu, "Variation-aware dynamic voltage/frequency scaling," in *15th International symposium on High-Performance Computer Architecture (HPCA'09)*, November 2009.
[3] J. Patel, "CMOS process variations: A critical operation point hypothesis," Online, April 2008, www.stanford.edu/class/ee380/Abstracts/080402-jhpatel.pdf.
[4] J. Sartori and R. Kumar, "Alleviating voltage scaling limitations of razor-based designs," in *Proceedings of the 18th IEEE Workshop on Logic and Synthesis (IWLS 2009)*, 2009.
[5] L. Chakrapani *et al.*, "Probabilistic system-on-a-chip architectures," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 12, no. 3, pp. 1–28, August 2007.
[6] J. Bau *et al.*, "Error resilient system architecture ERSA for probabilistic applications," in *IEEE Workshop on Silicon Errors in Logic - System Effects, SELSE.*, 2007.
[7] A. Kahng, S. Kang, R. Kumar, and J. Sartori, "Designing processors from the ground up t allow voltage/reliability tradeoffs," in *Proceedings of the 16th IEEE International Symposium on High-Performance Computer Architecture (HPCA-2010)*, 2010.
[8] ——, "Slack redistribution for graceful degradation under voltage overscaling," in *Proceedings of the 15th IEEE/SIGDA Asia and South Pacific Design and Automation conference (ASPDAC)*, 2010.
[9] R. Kumar, "Stochastic processors," in *NSF Workshop on Science of Power Management*, March 2009.
[10] S. Narayanan, G. Lyle, R. Kumar, and D. Jones, "Testing the critical operating point (cop) hypothesis using FPGA emulation of timing errors in over-scaled soft-processors," in *SELSE 5 Workshop - Silicon Errors in Logic - System Effects*, March 2009.
[11] D. Sylvester *et al.*, "ElastIC: An adaptive self-healing architecture for unpredictable silicon," *IEEE Des. Test. Comput.*, vol. 23, no. 6, pp. 484–490, June 2006.
[12] J.-W. van de Waerdt *et al.*, "The TM3270 media-processor," Nov. 2005, pp. 12–342.
[13] Synopsys, "Synopsys design compiler user's manual," Online: http://www.synopsys.com, 2009.
[14] Cadence, "Cadence SoC encounter user's manual," Online: http://www.cadence.com, 2009.
[15] ——, "Cadence signal storm user's manual," Online: http://www.cadence.com, 2009.
[16] D. Ernst *et al.*, "Razor: Circuit-level correction of timing errors for low-power operation," *IEEE Micro*, vol. 24, no. 6, pp. 10–20, Nov. 2004.
[17] T.-C. Chen *et al.*, "Fast algorithm and architecture design of low-power integer motion estimation for H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 5, pp. 568–577, May 2007.
[18] Joint Video Team, "JM reference software JM10.2," Online: http://iphome.hhi.de/suehring/tml/.
[19] B. Shim, S. R. Sridhara, and N. R. Shanbhag, "Reliable low-power digital signal processing via reduced precision redundancy," *IEEE Trans. VLSI Syst.*, vol. 12, no. 5, pp. 497–510, May 2004.