Microprocessors and Microsystems xxx (2013) xxx–xxx

Contents lists available at ScienceDirect

Microprocessors and Microsystems

journal homepage: www.elsevier.com/locate/micpro

A survey of cross-layer power-reliability tradeoffs in multi and many core systems-on-chip

Ahmed A. Eltawil [a], Michael Engel [b], Bibiche Geuskens [c], Amin Khajeh Djahromi [c], Fadi J. Kurdahi [a,*], Peter Marwedel [b], Smail Niar [d], Mazen A.R. Saghir [e]

[a] Center for Embedded Computer Systems, University of California, Irvine, CA, USA
[b] Chair for Embedded Systems, Informatik 12, TU Dortmund, 44221 Dortmund, Germany
[c] Intel Labs, Hillsboro, OR, USA
[d] LAMIH - University of Valenciennes, ISTV2 UVHC, Campus Mont Houy 59313, Valenciennes Cedex 9, France
[e] Texas A&M University at Qatar, Electrical and Computer Engineering Program , P.O. Box 23874, Doha, Qatar

ARTICLE INFO

Article history:
Available online xxxx

Keywords:
Multi-core
Many-core
Power
Performance
Reliability
Cross-layer

ABSTRACT

As systems-on-chip increase in complexity, the underlying technology presents us with significant challenges due to increased power consumption as well as decreased reliability. Today, designers must consider building systems that achieve the requisite functionality and performance using components that may be unreliable. In order to do so, it is crucial to understand the close interplay between the different layers of a system: technology, platform, and application. This will enable the most general tradeoff exploration, reaping the most benefits in power, performance and reliability. This paper surveys various cross layer techniques and approaches for power, performance, and reliability tradeoffs are technology, circuit, architecture and application layers.

1. Introduction

Multicore platforms are quickly becoming the platform of choice to implement complex Systems-on-Chips (SoCs). The transition to new process technologies has enabled significant on-chip device densities. The shrinking size of transistors has resulted in lower power consumption, thereby narrowing the power gap between programmable and ASIC approaches. In spite of these advantages, many challenges still remain in the design and implementation of specific applications onto multicore systems. One can identify three main challenges facing multicore designers. The first and foremost is power consumption which is on the rise due to the complex algorithms executing on these platforms that demand both a heavy use of computational resources, as well as a large volume of memory and communication. The second challenge is technology related, where scaling is both an enabler and a limiter: it enables unprecedented integration, including the ability to integrate large memories on chip, with the downside being a penalty in leakage power as well as reliability. Finally, the third challenge is cost, driven by a highly competitive marketplace that demands the smallest die size possible. Thus SoC designers are faced with the daunting dilemma of generating high yielding architectures that integrate vast amounts of logic and memories in a minimum die size with minimum power consumption.

In its current definition, yield indicates a 100% defect free chip, where circuits such as built-in self-test and built-in self-repair are used extensively to guarantee a high yield. Many traditional design approaches have focused on error-free design, and there has been significant research in attempting to guarantee error-free design. However, the International Technology Roadmap for Semiconductors (ITRS) trends clearly show that it becomes economically impractical to insist on a 100% error-free SoC in terms of area and power [1]. Thus, there is a critical need for a radically new approach to designing reliable multicore systems using inherently unreliable components: this approach must necessarily expand the design space across abstraction layers and cross-couple constraints across the circuit, architectural platform and the application abstractions.

Towards that end, one can broadly classify systems in two major categories:

1. Applications that are *inherently error tolerant* such as communications, multimedia and wireless which provide an opportunity to generate a range of acceptable designs for varying amounts of error in the system. For instance, communication and wireless systems have a high level of redundancy introduced at the system level, allowing for a tradeoff between attributes such as bit-error rate (BER) and signal-to-noise ratio (SNR). By

* Corresponding author. Tel.: +1 949 824 8104.
E-mail address: kurdahi@uci.edu (F.J. Kurdahi).

0141-9331/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.micpro.2013.07.008

removing the artificial barrier between the system level design and the circuit level implementation, designers can explore an entirely new design space (as shown in Fig. 1) where controlled hardware errors can be treated in a similar manner as "channel errors" thus contributing to the noise floor while still meeting stated system metrics. This scenario presents the most opportunity for innovation by actively exploiting errors across abstraction levels, e.g., aggressive voltage scaling may introduce errors at the circuit/hardware level, but these errors are made visible to, and handled at the system level.

2. Applications that are *stringently error-constrained*, where the error must be detected and corrected at a cost in terms of latency and performance. For instance, consider the cache of a processor in a multicore system: due to process variations, circuit-level techniques that enhance memory performance may result in errors that necessitate changes in the architecture of the circuit to both detect and correct the errors. This approach is in effect changing the statistics of the underlying error mechanisms. Such applications require the design of highly optimized hardware that utilize parallel architectures or time sharing to detect and correct for errors as well as microarchitecture approaches such as hardware shadowing or redundancy.

Thus the ability of the system to handle errors is highly dependent on the statistics of the errors and also on the algorithm running on the hardware, which implies that this has to be a dynamic process, *optimized at design time and managed during run-time*. To be able to extract the most benefit out of this error aware approach, it is important to examine the relationship between (a) the constituent components of an architecture and their vulnerability in terms of power consumption and reliability as a function of the operating conditions, and (b) the needs, assumptions and requirements of the application layers depending on this architecture. The intricate interaction between different controlling mechanisms and their benefits and costs creates an opportunity for finding a global optimum in terms of performance spanning across multiple levels of the design hierarchy.

The ITRS roadmap [1] indicates that embedded memories will dominate the die area in the near future, rising from 71% now, to close to 94% by 2014. The increasing market demand for having larger size memories on chip has flagged the power consumption of the SRAM/Cache as the major portion of chip power consumption. For this reason, we focus many (but not all) of our investigations in this paper on SRAM-related techniques for exploring power-performance-reliability design space exploration.

The remainder of the paper is organized as follows: Section 2 examines the technology layer and the generic concept of variability, Section 3 examines the platform layer at the hardware and micro architectural level, while Section 4 considers the software and compilation perspective. Section 5 considers the application layer. Conclusions are drawn in Section 6.

2. Technology scaling and challenges

Over the past few decades technology scaling has continued to follow Moore's Law. As this pursuit continues for technologies beyond 22 nm, the decrease in feature size (Fig. 2, [2]) has supported ever increasing on-chip device densities. In order to reap the full benefits of technology scaling, a variety of challenges needs to be managed: increasing process variations, transistor aging variations and exponentially increasing leakage currents. As transistors continue to shrink in size, the limits imposed by leakage currents on transistor threshold voltage make it difficult to continue reducing transistor supply voltage as required by Dennard's scaling rules [3]. Coupled with increasing transistor counts due to Moore's Law, the resulting increase in power density has come to be known as the power wall, and is a prime reason for reliability problems that are directly related to increasing die temperatures. These include electromigration, stress migration, electron tunneling, gate-oxide breakdown, time-dependent dielectric breakdown, and thermal cycling, which can all lead to permanent, catastrophic, hard failures. Low transistor supply voltages and noise margins also lead to timing violations and increase the susceptibility of sequential circuit elements and memories to single-event upsets (SEU's), which are due to atmospheric neutrons and alpha particles. SEU's flip the values of stored bits but do not otherwise cause permanent damage. Nonetheless, these transient, soft errors can result in logical and timing errors.

To ensure correct functionality, designers will need to rely on careful co-optimization of process, circuit and layout techniques to meet ever challenging performance and power targets. Traditionally, designs have built fixed margins into operating frequency and voltage to ensure error-free operation under worst-case conditions in the presence of variation. This worst-case approach does not consider circuit behavior or implementation details and hence decreases design efficiency. Accurate statistical modeling of variation and its inclusion into timing and power convergence is necessary to recover design circuit margin while preserving pessimism to ensure quality and yield [4].

2.1. Variation sources

The CMOS variation sources can be classified into two groups: historical and emerging variations [5]. Historical variation sources include patterning proximity effects, line-edge and line-width roughness, polish variation, gate oxide thickness variation, fixed charge, defects and traps. These sources continue to require innovative solutions for each subsequent technology node. The emerging variation resources used to have a minor impact, but now present major challenges. Chief among them are random dopant fluctuation, implant and anneal variation, variation associated with strain and gate material granularity.

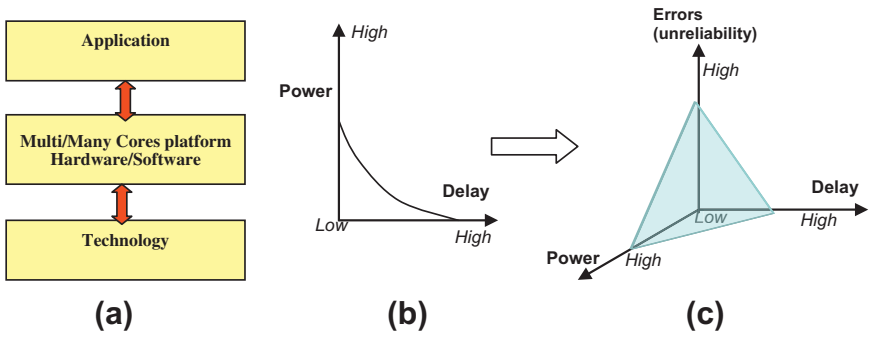

Fig. 1. (a) Layered system design, (b) traditional power-delay design space, (c) emerging powerdelay-(un)reliability design space.

Please cite this article in press as: A.A. Eltawil et al., A survey of cross-layer power-reliability tradeoffs in multi and many core systems-on-chip, Microprocess. Microsyst. (2013), http://dx.doi.org/10.1016/j.micpro.2013.07.008

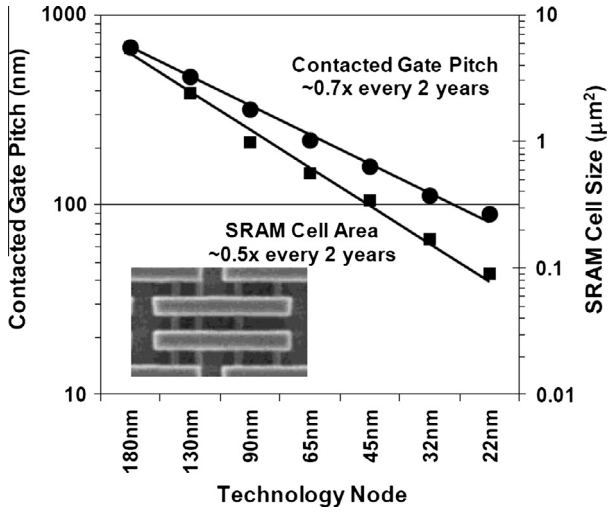

Fig. 2. Contacted gate pitch and SRAM cell size scaling trends, [2].

Process variation is bucketed into systematic and random variation. Systematic variation is the main driver for competitive yield results and can be defined as the movement of the mean (example: die-to-die variation on a wafer). Random variations are defined as a uniform variation around a mean and are proportional to the standard deviation, σ, of the difference between critical parameters, such as threshold voltage (V_T), for two neighboring devices (example: within die variation) (see Fig. 3).

2.2. Margining

In fact, low voltage operation has drastically increased the importance of maintaining a low random threshold voltage variation, σV_T, to enable the lowest operating voltage, $V_{\min}$, for a target yield and performance. For sub-micron technologies, random dopant fluctuation (RDF) is the major contributor for threshold voltage variation and is usually represented by [7]:

$$\sigma V_T = \frac{1}{\sqrt{2}}\left(\frac{c_2}{\sqrt{W_{eff}\cdot L_{eff}}}\right) \quad (1)$$

As supply voltage headroom, $V_{cc} - (V_T + 3\sigma_{V_T})$ [8], and feature sizes continue to shrink, $V_{\min}$ requirements push design and process techniques for large array elements, such as SRAM, ROM and Register File arrays. Similarly, transistor aging mechanisms generate additional shifts of device threshold voltage over time. To meet end of life (EOL) circuit requirements, accurate aging prediction models need to be included in the product design methodology. Among the several physical effects that cause transistor aging Negative Bias Temperature Instability (NBTI) [9], and Hot Carrier Injection (HCI) are the dominant effects [10]. These NBTI and HCI-induced wearout effects depend strongly on usage, temperature and supply voltage of the affected transistors.

2.3. Variation and low power design

Process variation is one of the main challenges that designers are facing in the SoC domain. To achieve high yields and guarantee operation under wide range of operating conditions under parameter variation, designers are forced to assign margins in the design. In older technology nodes, since the amount of the variation ($\frac{\sigma_{DP}}{\mu_{DP}}\%$, where DP is the design parameter) was relatively small, designing for the worst case condition based on the yield targets would meet the design requirements such as power, area, and performance. However, in the sub-100 nm regime, especially sub-32 nm, designing for worst case, which sometimes leads to assigning margins as large as up to 50%, will affect metrics of interest drastically. Moreover, one of the most important SoC design parameter which has gained a lot of attention in the past few years is power consumption. Several techniques have been developed and implemented to address the process variation, adaptation, and power consumption which will be explained here. In general, these works can be divided into two major different categories: (a) Software approach and (b) Hardware approach. Most software level approaches are done via what is termed "application-aware adaptation" where individual applications determine how best to adapt resources while preserving the ability of the system to monitor overall resources and enforce allocation decisions. On the other extreme are hardware/circuit oriented approaches which stress changing the hardware architecture to be ultra-efficient in power. In this section we focus on the hardware approach. Some of the hardware techniques are

I. Static Voltage Scaling (SVS): This technique is mostly used to address die-die variation. The idea is to design for the worst case and then programmable or fixed fuses are used to adjust the voltage that corresponds to the speed of the silicon. In other words, for each die, the lowest supply voltages that guarantee error-free operation are selected. In some

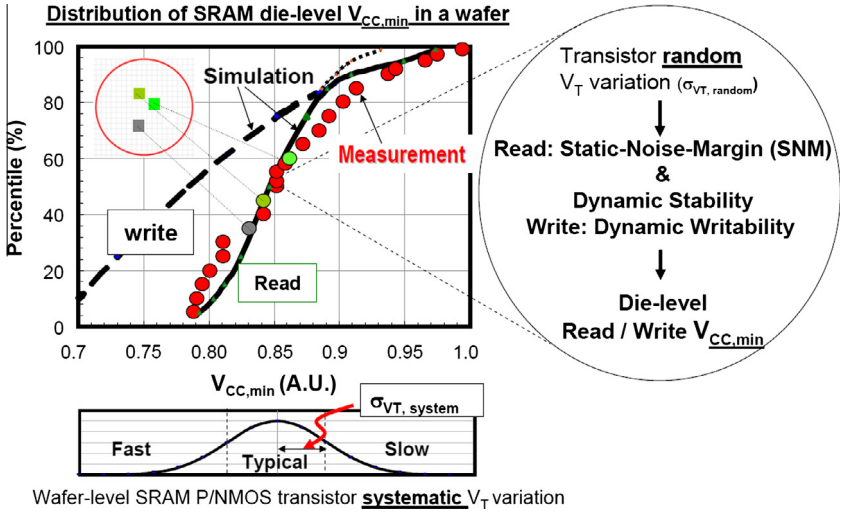

Fig. 3. 6T SRAM minimum operating voltage model with variation, [6].

Please cite this article in press as: A.A. Eltawil et al., A survey of cross-layer power-reliability tradeoffs in multi and many core systems-on-chip, Microprocess. Microsyst. (2013), http://dx.doi.org/10.1016/j.micpro.2013.07.008

cases, these values may be updated after the product is shipped. Power savings of up to 15% is reported by using this technique [23].

II. Dynamic Voltage Frequency Scaling (DVFS): DVFS is a method to provide a variable amount of energy for a task by scaling the operating voltage/frequency. Theoretically, infinite number of voltages and frequencies are needed to achieve the lowest energy consumption points, however in practice these sets are usually limited to a hand full or a couple and instead of DVFS, quantized DVFS is usually used. Although quantizing the voltages decreases complexity, it reduces savings if V_{dd} is constant for a given time constraint. To address this, techniques such as voltage dithering are proposed [22] which suggests switching V_{dd} in the timescale of the changing workload. Dithering within the time constraint gives average energy close to the optimum. Power savings of up to 30% have been reported using this technique [24].

III. Adaptive Voltage (Frequency) Scaling (AVS/AVFS): This technique is a more general version of DVFS and has been introduced to address some of the challenges in sub-32 nm technology nodes such as temperature effect, aging, etc. The main idea is to have sensors on each die (tunable replica circuit [25] , temperature sensor, performance sensors, etc.), and use the information from these sensors to adjust the operating conditions. In most cases a power management unit (PMU) is needed to decide on the power states. Savings of up to 40% are expected from this technique [26].

Traditionally, when any voltage scaling techniques (SVS, DVFS, AVS, etc.) is used, a design tradeoff is performed between power and delay where lower power is attained at the cost of larger delay [23,24], typically by running at a lower operating frequency which is set by the weakest performer in the overall system. In other words, the circuit that exhibits the most delay or stability issues limits the performance of the whole system. In a majority of scenarios the culprit is embedded memories, since they exhibit the highest vulnerability to supply changes as compared to logic [22].

2.4. Low power memory design trends

Random and systemic device variations pose significant challenges to SRAM V_{min} and low voltage performance. To overcome these challenges, industry has explored multiple design and process technology changes for the traditional 6T SRAM cell (see Fig. 4). Examples are the use of read and write-assist designs, ranging from V_{dd} Collapse during write [11], to write word line boosting [12,51], and read word line under-drive [13]. Most techniques are based on reducing the intrinsic contention inside the bit cell during read and write operation. Larger cell topologies, such as 8T register file cells [14], and 10T cells with column interleaving support [15], provide improved cell stability at the expense of area and power by decoupling read and write operations. Literature also includes more exotic examples of asymmetrical cells, such as 5T [16], and 7T [17], cells. Another significant design centric solution involved the change from a "tall" to a "wide" cell design [18]. The wide design improved critical dimension control and variation by aligning poly and eliminating diffusion corners and improved performance by shortening the overall bit-line length (see Fig. 4).

Increasing supply voltage has always been applied to counter process variations at the expense of increased leakage and dynamic power. Hence, dual supply techniques combining a low logic power supply with a slightly higher SRAM power supply to improve cell stability are one method to overcome large leakage and counter process variation [19]. In addition, repair techniques such as redundancy and error checking and correcting (ECC) are applied to reduce cell failure rate and improve V_{min} [20].

2.5. Modeling memory failures

At the system level embedded memories are typically abstracted away as buffers that can have an arbitrary low error rate. Hence physical layer designers have to ensure that memories have sufficient error detection and correction techniques to ensure a virtually error free view of the memory for system-level multicore designers. Typically, it has been assumed that row and column redundancy is sufficient to capture all the faults. While this is true in case of static manufacturing faults, this model cannot be sustained as scaling progresses due to the random nature of the fluctuation of dopant atom distributions. In fact, in sub-100 nm designs, Random Dopant Fluctuation (RDF) has the dominant impact on the transistors strength mismatch and is the most noticeable type of intra-die process variation that can lead to cell instability and failure in embedded memories, [32–34]. RDF has a detrimental effect on transistors that are co-located within one cell by creating a mismatch in their intrinsic threshold voltage V_t. Furthermore, these effects are a strong function of the operating conditions (voltage, frequency, temperature, etc.). Fig. 5 shows the effect of voltage scaling on six transistor SRAM memory bit cell failure using 65 nm Predictive Technology Model (PTM) [35]. ***This figure illustrates that designers can tradeoff error tolerance versus supply voltage for a fixed performance***. In other words, to achieve a low power solution, the designer must decide on the acceptable level of error tolerance that is permissible by the application and the overall system design while still maintaining the required performance. Given that level, and a required performance level (the cell speed), the designer can select the appropriate V_{dd} from Fig. 5.

3. Power-reliability tradeoffs in microprocessor designs

Microprocessor designers have traditionally relied on packaging and cooling technologies to dissipate heat and combat rising die temperatures, and radiation-hardening, package shielding, and cir-

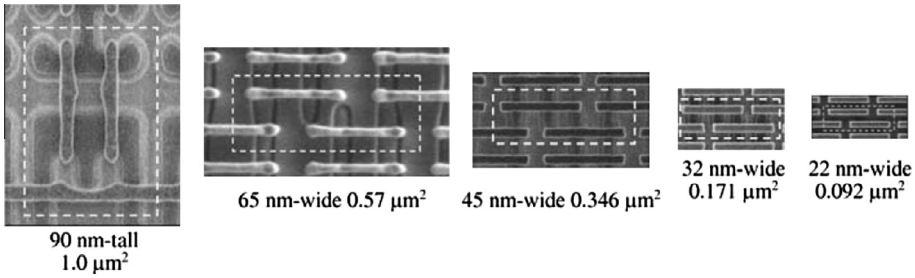

Fig. 4. Cell topology enhancements for variation mismatch improvement, [5].

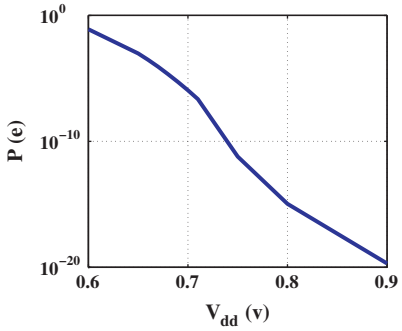

Fig. 5. Probability of error versus V_{dd}.

cuit redundancy to overcome soft errors. In this section, we survey recent circuit and micro architectural techniques that microprocessor designers are using to mitigate the effects of hard and soft errors.

3.1. Dynamic reliability management

To ensure reliable operation over a specific lifetime, server-class microprocessors are often overdesigned to withstand worst-case temperature and resource utilization conditions. Since typical workloads rarely exhibit worst-case characteristics, excess reliability qualifications can be leveraged to improve performance. Similarly, the costs and yields of commodity microprocessors can be improved by under-designing processors to achieve reliability levels for typical operating conditions. When workloads exceed a processor's operational reliability limit, performance is throttled to maintain reliability qualifications.

Dynamic reliability management (DRM) uses a reliability-aware microprocessor (RAMP) model to exploit the variability in reliability qualifications in over- and under-designed microprocessors, respectively [36,37]. The RAMP model is used to compute an instantaneous reliability measure, and relies on performance counters and thermal sensors to identify opportunities for boosting or throttling performance. Performance is controlled by scaling the operating frequency and corresponding supply voltage through DVFS techniques, or by adapting microarchitectural resource utilization to achieve specific levels of instruction-level parallelism.

ERSA [63] is a more drastic technique for saving power on a multicore architecture. It divides the cores into reliable and unreliable cores and proposes to reduce the voltage on all parts of the unreliable cores. Even though there is no error recovery method introduced and the frozen cores are restarted by reliable ones, the output remains more than 90% accurate for the tested set of benchmarks. Earlier works such as the Aura/Odyssey/Coda project [64] investigated mobility and adaptation at the software level via what was termed "application-aware adaptation".

3.2. Better-than-worst-case design

Another approach to overcoming the constraints of worst-case design is to adopt better-than-worst-case design techniques that maintain reliability while enabling microprocessors to be optimized for the performance and energy efficiency achievable under typical operating conditions [38,39]. This approach relies on the use of dedicated checker components to maintain logical and timing correctness.

For example, DIVA [68], is an instruction checker that is cascaded with an out-of-order execution processor core. Instructions are pre-executed on the processor core to compute results, dereference memory addresses, and resolve the outcomes of branch instructions. However, instead of retiring instructions and updating processor state, the instructions, along with their operands and results, are re-executed and validated in the DIVA checker in program order. If an error is detected, it is automatically corrected by the checker. The processor state is then cleared, and the pipeline is flushed and restarted at the instruction following the errant instruction. The processor core acts like an instruction predictor that can leverage techniques such as DVFS to enhance performance and power efficiency without consideration for hard or soft errors. At the same time, the DIVA instruction checker provides a simple but effective solution for maintaining reliability.

Razor logic [30], is another form of checker component that is used to overcome the effects of timing and soft errors. Timing errors result from circuit delay variations that are due to operating at sub-critical voltage levels or very high frequencies. With razor logic, timing-critical flip–flops are coupled with shadow latches that operate using delayed clock signals that ensure the latch set-up times are always met. Logic gates are used to compare the outputs of the flip–flops and their shadow registers. When the outputs differ, the value in the shadow latch is used to propagate the correct result. Razor logic has been used in microprocessor pipelines to automatically detect and correct errors before they are propagated to subsequent pipeline stages. Earlier pipeline stages are also flushed to restart the pipeline at the instruction immediately following the errant instruction.

3.3. Power-aware slack redistribution

Microprocessor circuits are characterized by multiple timing paths that are very close to the critical path. This results in a massive incidence of timing violations once the supply voltage is scaled beyond a critical operating point. Power-aware slack redistribution [40] is an iterative, design-time technique that optimizes frequently exercised critical timing paths with negative slack. By carefully choosing which paths to optimize for a given operating point, the resulting rate of timing errors can be controlled. The timing characteristics of the paths are modified by swapping smaller but faster logic cells along a path by equivalent cells that are larger but slower. This enables a power-reliability trade-off to be made where more aggressive operating points and higher levels of power efficiency can be achieved at the expense of a higher but more gradual incidence of timing errors. This also enables the use of the most suitable error mitigation techniques for different error rates.

3.4. Stochastic processors

A large class of applications such as video and image processing, communications, and net-working are based on algorithms that are inherently noise tolerant. By relaxing the constraints on hardware correctness and relegating the handling of errors to software, microprocessors can be designed to trade-off gradually increasing error rates for higher levels of performance and power efficiency.

Stochastic processors [41,42] are microprocessors that leverage circuit and microarchitecture techniques to operate at sub-critical voltages and super-critical frequencies at the expense of gradually increasing timing error rates. The circuit techniques are based on the redistribution of timing slack along frequently-exercised critical paths, while the microarchitectural techniques involve reducing the number of critical timing paths inside the processor. The latter can be achieved by using fewer or smaller regular structures such as register files, cache memories, or multiple instruction issue logic. Due to the regularity of these hardware structures, circuit paths have very similar timing characteristics making it more difficult to redistribute slack. Another microarchitectural technique to reduce the number of critical timing paths is to use functionally

Please cite this article in press as: A.A. Eltawil et al., A survey of cross-layer power-reliability tradeoffs in multi and many core systems-on-chip, Microprocess. Microsyst. (2013), http://dx.doi.org/10.1016/j.micpro.2013.07.008

equivalent computational units having different reliability characteristics. For example, a ripple-carry adder is slower than a Kogge–Stone adder, but is more resilient to timing errors when the supply voltage drops below critical levels. Microprocessors can therefore be designed to dynamically select the most suitable computational unit based on the required level of performance, power, or reliability. Since stochastic processors can control the error rates at different operating points, they can also be used in conjunction with the most suitable circuit- and software-based error mitigation techniques.

3.5. Dark silicon

The proliferation of general-purpose multi-core processors is a direct consequence of the power wall and its resulting limits on operating frequencies. Although microprocessor manufacturers continue to increase the number of cores per chip [43], this trend cannot continue without reaching the state of dark silicon [44] where power limits necessitate powering off most of the cores in a chip. To avoid this problem, designers are exploring heterogeneous, many-core microarchitectures that include a mix of general- and special-purpose cores. Specialized cores such as GPUs and DSPs are more energy-efficient than general-purpose cores. The graphics or signal processing applications that run on these cores are also based on algorithms that are more resilient to errors, making it easier to trade-off hardware reliability for higher levels of power- and energy-efficiency. Moreover, such specialized cores can often achieve higher levels of performance than general-purpose cores by exploiting customized microarchitectural features. This enables operating specialized cores at lower clock frequencies, which lowers thermal dissipation and enhances reliability.

New research projects such as DeSyRe [46] are exploring the design of reliable, energy- efficient, multi-core architectures using a mix of reliable and unreliable components. The reliable components will mainly be used to detect errors and manage the use of the unreliable but power-efficient components to provide the required functionality.

3.6. Reconfigurable processors

Reconfigurable processors achieve high levels of performance and energy efficiency through instruction-set or microarchitectural specialization. They use programmable logic fabrics to implement application-specific hardware structures, and are programmed using configuration bit streams that are commonly stored in SRAM-based configuration memory banks. SEU's can cause the contents of the configuration memory to be corrupted, and this introduces logical or functional errors. One way to overcome these errors, especially in mission-critical systems, is to first use classic triple-modular redundancy (TMR) to identify the errors, and then use full or partial dynamic reconfiguration to reload the configuration bit stream from a reliable (e.g. radiation-hardened) data store. This process is called *scrubbing* [45].

3.7. Cache reliability mitigation

Today's caches are very carefully designed and "protected" because of noise and defect concerns, at the cost of a significant loss of opportunity in power reduction. Given the projected impact of variability and other factors on memory reliability, it is important to develop cache architectures that are tolerant to process and environment variations and that lend themselves to significant reduction in power consumption while sustaining a negligible (if any) performance penalty. Fault-tolerant, variation-aware caches can indeed get the best of both worlds: reduce power consumption significantly (expected > 50%), and maintain the same level of performance (operating frequency). In fact, given the anticipated rise in device densities in the coming decade, defect tolerant computing might very well be the only feasible approach. Finally, it is important to note that most dynamic voltage scaling (DVS) research in the past has focused exclusively on logic circuit performance while neglecting memories. More recently, researchers have studied the effect of dynamically and aggressively controlling the voltage, speed and error tolerance for caches to allow exploration of different cache alternatives and achieve a true optimal operation point.

In [50,51], the effects of process variation on the proper operation of the memories was investigated and the reliability issues, the defect types, problem sources and change in the memory cell robustness with process variation were identified. Based on these studies two categories of SRAM based memory structures can be considered (Fig. 6). The first category reduces the effect of process variation on the proper operation of the memory cells, resulting in reduction of the memory cell failure rate sensitivity to the voltage scaling [52,12]. The second category introduces a very high degree of tolerance against process variation by providing a fault tolerant mechanism based on moderate resizing of the memory structure [53,47] or dynamic redundancy (RDC-CACHE[50]) with far larger tolerance compared to previous work and slower slope of downsizing the effective memory size. Both allow the V_{cc}^{Min} of the SRAM cell array to be safely reduced. The cache described in [48] employs multiple copies of each data item in the cache and, unlike most previous approaches, does not require a priori knowledge of defect locations in the memory array.

4. Software and compilers

4.1. Software and the reliability-energy tradeoff

Reducing the energy consumption is one of the most important optimization objectives for future hardware/software systems. However, reductions in energy consumption by common methods like DVFS approach physical limitations. Instead, unconventional methods are in the focus of current research approaches. These methods include operating semiconductors at or below their threshold voltage or intentionally reducing the reliability or precision requirements of a system. Either way, these new approaches share a common idea – they exploit the tradeoff between energy consumption and reliability of a system.

However, when the reliability of a system is reduced directly or indirectly, a paradigm that was one of the most basic assumptions for software development for the last decades is no longer applicable. This paradigm, believed to be an adamant truth by most software developers, presumes that the hardware platform the code is executed on is predictable in terms of program results, i.e. the repeated execution of a given program using the same input data set always yields the same result. One notable exception from this

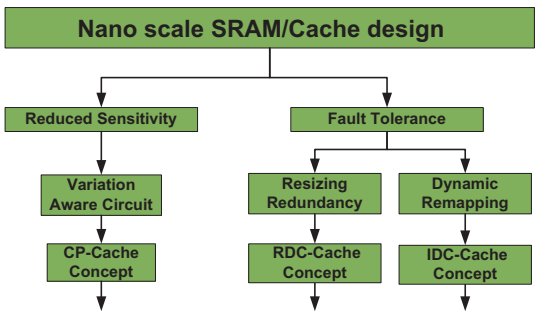

Fig. 6. Cache designs for variation tolerance.

Please cite this article in press as: A.A. Eltawil et al., A survey of cross-layer power-reliability tradeoffs in multi and many core systems-on-chip, Microprocess. Microsyst. (2013), http://dx.doi.org/10.1016/j.micpro.2013.07.008

fact is programs that require randomness – however, it is well-known that software developers have to invest significant work to generate sequences of pseudo-random numbers.

In the following paragraphs, we describe possible effects of exploiting the reliability-energy tradeoff on the execution of software and give an overview and examples of compiler-based approaches to mitigate the impact of these effects.

4.2. Error effects

On the software layer, the introduction of unreliability into a system usually expresses itself as a deviation of the system state from the originally expected state. In most cases, one or more bits of the system are temporarily or permanently altered. As a consequence, the execution of instructions as well as the processing of data in a program can potentially be disturbed. Depending on the component affected by the altered bit(s), the *effect* of this deviation in sys-tem state can range from no effect at all (an unused bit was hit) over errors in the output of a program (so-called silent data corruption) to program crashes.

For obvious reasons, reducing the energy consumption should still result in a system that does not crash. Small deviations in the output of a program, however, can be acceptable for a broad range of applications. As an example, signal processing applications like audio and video codecs or radio baseband operation can actually accept a given level of incorrectness in the results. For video decoding, e.g., this could result in a change in the color of a pixel for the duration of a frame. Such an effect is hardly discernible; it would definitely be acceptable if a considerable amount of energy could be saved this way.

When voluntarily accepting deviations of calculated results, however, it is important to identify those code and data objects in a program that are critical to the stability of its execution. This information is not readily available in commonly used programming languages.

4.3. Code annotations, analyses, and transformations

In order to master this energy-reliability tradeoff, thus, *application knowledge* has to be added to a program using annotations or extracted by applying static code analysis techniques. Annotations can apply to different kinds of program objects. The examples detailed below show different approaches. The two most common approaches include the annotation of code sections with criticality information and the annotation of the criticality of data objects.

While it is possible to annotate every code and data object this way, for large programs, creating all annotations manually is a tedious task. Thus, it is preferable to *generate* most annotations automatically using information on pivotal objects or objects which cannot be identified automatically, like input and output data. A compiler has the opportunity to propagate given annotations on reliability by using code and data flow analyses. These analyses have the additional benefit that dangerous inconsistencies in a program's data flow, such as the possible propagation of a value possibly affected by an error, can be detected and flagged as an error.

After this analysis and propagation step, all data and control flow objects of a program can be expected to be annotated with the *worst-case effect* an error in this object can have on the execution of the given program. This application meta data can now be exploited in various different stages in the compilation process or the execution of the program.

As a general guideline, all stages at compile and runtime *transform* the given program in order to circumvent or correct critical errors. Possible transformations on source-code or binary code level in the compiler can, e.g., introduce *redundancy* into a program's execution by applying software-level redundancy techniques like repeated execution of code sections. Detailed information on software-based error correction approaches can be found in [57]. Additional transformations can take place after the executable program is compiled. Either as part of linking process or as a post-link stage, the location of code and data objects can be modified. This way, systems that employ hardware components with heterogeneous reliability characteristics can be utilized by mapping critical code and data objects to known reliable components. An example for such a system is a system with two separate memories, one operated at its regular supply voltage level, whereas the other is supplied with a lower voltage. Application knowledge can also be used at runtime. Here, possible transformations include the modification of a program's executable code by using a JIT compiler or binary rewriting system [56]. This is useful in cases where a permanent defect, e.g. a defect in a CPU register caused by electromigration, is detected at runtime. Using binary rewriting, the executed code can be modified to no longer use the affected register.

Finally, meta data can also be useful to the operating system controlling the resources of a system. Here, it can be employed to decide at runtime if sufficient resources are available to correct an upcoming non-critical error in order to increase the service quality of a system [59]. Obviously, it is also possible to combine several of the discussed approaches.

One interesting and as-yet open research question here concerns yet another tradeoff. If application knowledge is to be used at runtime, the amount of meta data added to a program as well as the overhead required to process that information should be minimized. Otherwise, this overhead may well dwarf the energy savings achieved by utilizing the reliability-energy tradeoff in the first place. Here, tool support at compile time is required to determine the appropriate granularity for meta data and to actively reduce the size of meta information by grouping together sets of objects with identical reliability characteristics.

4.4. Use cases

Application meta data is already used in several research projects that aim to achieve improved fault-tolerance and/or reduced energy consumption for software. In this section, we outline approaches that showcase different approaches to provide annotations. Relax [54] is a framework for software recovery of hardware faults. Using a CPU ISA extension that allows software to mark regions of code for software recovery and simplified hardware that is more energy-efficient than complex error-correcting hardware as well as the required compiler support, the authors achieve an energy reduction of up to 20% for a set of benchmark applications when attempting to counter the effects of process variation. An example for code annotations in Relax is shown in Fig. 7.

Here, a block of code is braced by a set of relax and recover statements. The parameter to relax specifies the expected error rate for further analyses, whereas the code block containing retry specifies the action taken in case of an error.

The focus of the FEHLER project is to provide a flexible approach to error handling [6] in embedded real-time systems. In contrast to the approach featured by Relax, FEHLER extends a C compiler to support *type qualifiers*. These qualifiers, as shown in Fig. 8, indicate if the data stored in a given object is expected to be error-free (reliable) or can accept errors (unreliable). Static analysis methods in the compiler propagate reliability information to non-annotated data objects and detect possible inconsistencies. In the current version of FEHLER, this meta data is used at runtime to decide *if*, *when*, and *how* a given error has to be corrected. Using FEHLER, it is possible to clearly separate critical from non-critical operations and avoid all errors that threaten to crash an application while per-

Please cite this article in press as: A.A. Eltawil et al., A survey of cross-layer power-reliability tradeoffs in multi and many core systems-on-chip, Microprocess. Microsyst. (2013), http://dx.doi.org/10.1016/j.micpro.2013.07.008

```
int sum(int *list, int len) {
        relax (rate) {
                int sum = 0;
                for (int i = 0; i < len; ++i) {
                        sum += list[i];
                }
        } recover { retry; }
        return sum;
}
```

Fig. 7. Code annotations in Relax.

```
reliable int foo;
unreliable int bar;
...
foo = bar; // error
bar = foo; // propagation permitted
```

Fig. 8. Type qualifiers in FEHLER.

```
@Approx int a = ...;
int p = ...; // precise by default
```

Fig. 9. Type qualifiers in EnerJ.

forming best-effort error correction for the remaining errors given the currently available system resources.

FEHLER type qualifiers have also shown to be useful for employing *probabilistic* hardware components. Using the reliable/unreliable qualifiers, operations on a program's data objects can be mapped to probabilistic or non-probabilistic hardware components as required [55].

Similar to FEHLER, the final technique discussed here also employs type qualifiers. However, the EnerJ project [58] does not specify reliability requirements in terms of criticality to the system, but rather if a variable is allowed to contain an approximate result. By extending the Java language with the @Approximate type qualifier, as shown in Fig. 9, the JIT compiler is instructed to store the related variable in unreliable memory and possibly operate on it using unreliable arithmetic components. Again, analysis techniques are used to prohibit the propagation of approximate data to variables that are expected to be precise. Using EnerJ, the authors manage to achieve energy savings of 7–38% at little accuracy cost.

In summary, problems arising due to the energy-reliability tradeoff can be handled in software if additional meta data is provided. This meta data allows numerous different ways to selectively handle errors at compile and runtime using source code or binary transformations. These transformations enable adaptation; possible manifestations of this approach are the topic of various current research projects. In the future, generic concepts for transformations are required that are able to adapt to the large amount of different possible error models. However, the examples presented already clearly show the potential for exploiting the energy-reliability tradeoff on software level.

5. Exploiting application domain features

5.1. Application-aware adaptive margining techniques

As mentioned earlier, in most cases embedded memories exhibit the highest vulnerability to supply changes as compared to logic. For this reason, when voltage scaling is used, memories are typically treated separately to maintain the margins such that the device will meet timing 100% of the time with the new settings. Furthermore, it is interesting to note that unlike error aware logic design approaches such as TEAtime [27] and Razor [28,30], traditional memory design approaches have generalized the use of embedded memory without incorporating application knowledge in factoring the required operational failure rates of memories. The unwavering design assumption is that the failure rates should be minimized. While clearly, this is an ideal design goal to achieve; modern scaling laws have rendered this goal extremely costly resulting in excessive margining to achieve manufacturing goals.

In other words, the major goal of all these different approaches is to adapt the system to provide the optimum performance based on the current operating conditions while regulating the power consumption. The universal underlying assumption made is that the adaptation technique used has to maintain 100% correctness of the computational engine of the system regardless of the application. In other words, the adaptation technique is not designed to be fault tolerant. Fortunately, many application domains are inherently error-aware [49], providing an opportunity to generate a range of acceptable designs for varying amounts of error in the system. Specifically, multimedia, communication, and wireless systems have a high level of redundancy introduced at the system level, allowing for a tradeoff between attributes such as bit-error rate (BER) and signal-to-noise ratio (SNR). The research in [65–67] promoted the use of "Algorithmic noise tolerance" and proposed using adaptive filters and replication to minimize the impact

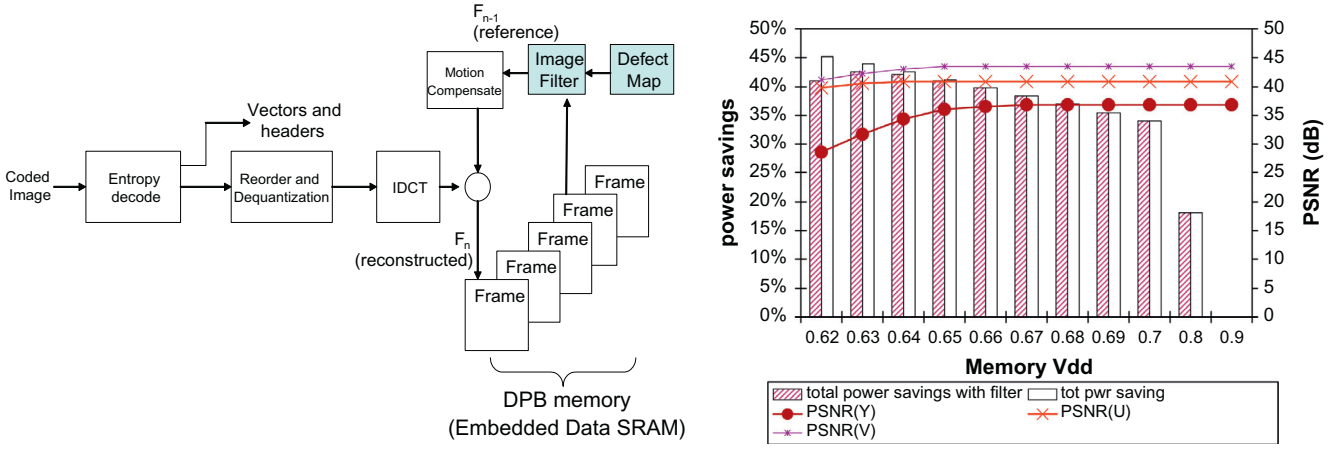

Fig. 10. H.264 decoder with filtering (left) & Image quality (PSNR) versus power savings at different V_{dd} levels (Foreman Video) (right).

Please cite this article in press as: A.A. Eltawil et al., A survey of cross-layer power-reliability tradeoffs in multi and many core systems-on-chip, Microprocess. Microsyst. (2013), http://dx.doi.org/10.1016/j.micpro.2013.07.008

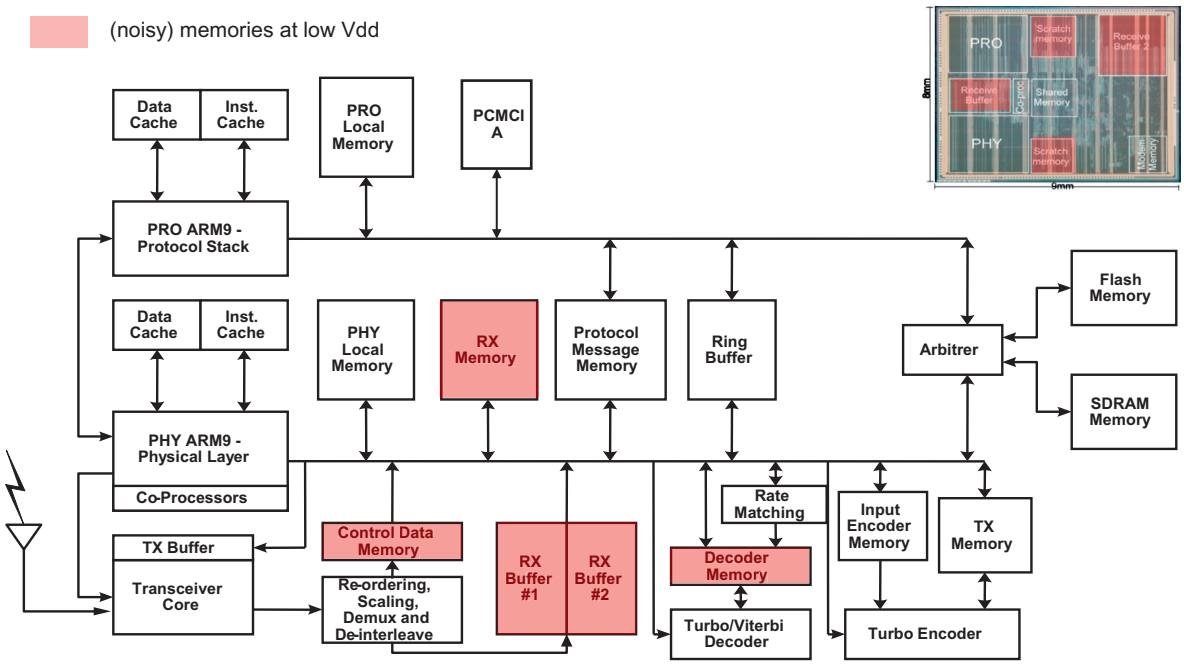

Fig. 11. WCDMA architecture and error-tolerant memories.

of scaling V_{dd} beyond the critical region for basic DSP functions, e.g. filtering and other communication blocks. Similarly, multimedia applications have soft real-time constraints that provide an opportunity to tradeoff Quality of Service (QoS) for errors in the data stream. It is conjectured that cross-layer error-awareness opens a large space of feasible designs exhibiting a range of power, performance and cost attributes, allowing SoC designers to optimize and select designs that otherwise would not have been available [29].

5.2. Error-resilient multimedia applications: H.264 decoder case study

Consider the H.264 system shown in Fig. 10 as a representative application for mobile multimedia systems. One of the biggest challenges is power consumption, which is typically addressed by power management, mainly by reducing the supply voltage. However the range of such a reduction is limited by: (1) performance constraints and (2) component reliability under very low V_{dd}.

By design, these systems have built-in error resiliency that has been exploited in many different compression and transmission schemes mainly as a quality tradeoff. In [60] aggressive voltage scaling is applied to *embedded memories* resulting in low power, high frequency operation, albeit, with errors due to scaling. Based on the error statistics the performance and overhead (in terms of power and area) of various filtering and mapping techniques that compensate for the errors were analyzed and quantified, thus enabling the system to operate at lower voltages while meeting system specifications. Finally, the expected system power savings due to the above mentioned approach were also quantified. Fig. 10(right) shows the results of such an exploration. When V_{dd} on the decoder memories is scaled aggressively, their reliability decreases and as a result, the output quality (defined as Peak Signal to

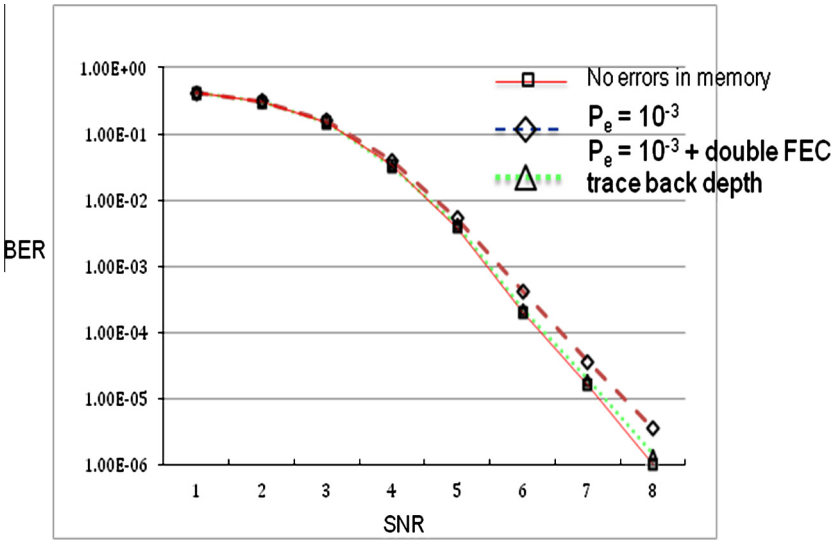

Fig. 12. Effect of the WCDMA memory errors on the system bit error rate.

Please cite this article in press as: A.A. Eltawil et al., A survey of cross-layer power-reliability tradeoffs in multi and many core systems-on-chip, Microprocess. Microsyst. (2013), http://dx.doi.org/10.1016/j.micpro.2013.07.008

Noise Ratio, or PSNR) drops. This can be compensated for by image filtering, which consumes power so the gains from V_{dd} reduction tend to lessen as error rates increase. However, the overall results indicate that good performance can be maintained even at very low V_{dd}, while saving over 40% in overall system power consumption.

5.3. Error-resilient wireless modem application: 3G case study

Similar to multimedia, wireless systems can benefit from trading off power for reliability. This is investigated using WCDMA as a representative of a wireless physical layer. Fig. 11 depicts the top-level block diagram of a diversity enabled WCDMA SoC modem developed by the PIs in prior work [60]. The SoC includes the modem section (RAKE receiver), the coding layer and the protocol layer of the standard. It is based on a dual embedded microcontroller architecture. The symbols from the modem are soft values with 10 bit precision that are available for all the data and control symbols transmitted on the data channels. Naturally, control symbols are very important and thus must be stored in a protected memory with minimum loss. However, data symbols possess a high degree of redundancy typically inserted by the channel coding scheme. Specifically in WCDMA, both Turbo and Viterbi schemes are supported [61]. Thus the data memory can be partitioned into defect tolerant and non-defect tolerant sections. A defect tolerant memory, is a memory that is used primarily to buffer data and thus can be a target of aggressive power management. It is interesting to note that the data buffering memories (defect tolerant candidates) consume approximately 50% of the overall memory required for the entire modem.

Fig. 12 shows the effect of the memory errors on the WCDMA BER for different SNR values. As expected, given the same SNR, introducing memory errors in the buffers before and/or after the Viterbi decoder result in higher BER at the output of the modem. One means of improving the performance of a Viterbi decoder is increasing the trace back depth (TBD), at the cost of a larger internal memory and power consumption of the decoder core. In other words, by increasing the trace back depth, the decoder can detect and correct more errors. Fig. 12 shows the effect of the trace back depth on WCDMA bit error rate. As shown in the figure, a WCDMA system with $P_e = 10^{-3}$ errors in the above-mentioned error resilient memories performs almost identical compared to the system with no error in those memories and half of the trace back depth. A power analysis of the architecture indicates that the overall memory consumes roughly 45% of the total power. In [62] it was shown that by applying error aware dynamic voltage scaling a savings of 46% in leakage power and 44% in dynamic power is possible in the error tolerant memories. It is important to note that these savings are independent of other power savings methods such as reducing frequency of operation.

While these case studies highlight a significant opportunity in power savings, it requires an important paradigm shift in today's system design flow. Current flow emphasizes compartmentalization between system level designers and backend (chip) designers, thus necessitating 100% correctness in hardware. The new paradigm de-compartmentalizes this flow and allows system designers to be aware of the physical layer through model abstractions.

6. Summary and conclusion

As systems-on-chip increase in complexity, the underlying technology presents us with significant challenges due to increased power consumption as well as decreased reliability. Today, designers must consider building systems that achieve the requisite functionality and performance using components that may be unreliable. In order to do so, it is crucial to understand the close interplay between the different layers of a system: technology, platform, and application. This will enable the most general trade-off exploration, reaping the most benefits in power, performance and reliability.

References

[1] International Technology Roadmap for Semiconductors. <http://www.itrs.net>.
[2] C. Auth et al., A 22nm high performance and low-power CMOS technology featuring fully-depleted tri-gate transistors, self-aligned contacts and high density MIM capacitors, in: IEEE Symposium on VLSI Technology, June 2012, pp.131–132.
[3] R.H. Dennard, F.H. Gaensslen, V.L. Rideout, E. Bassous, A.R. LeBlanc, Design of Ion-Implanted MOSFET's with very small physical dimensions, IEEE Journal of Solid-State Circuits SC-9 (5) (1974) 256–268.
[4] S. Sutanthavibul et al., Statistical approach to low power and high volume pineview atom-based SoC design, in: International SoC Design Conference, November 2009, pp. 228–231.
[5] K. Kuhn, Variability in nanoscale CMOS technology, Science China Information Sciences 54 (5) (2011) 936–945.
[6] Y. Wang et al., A 4.0 GHz 291Mb voltage-scalable SRAM design in 32nm high-K metal-gate CMOS with integrated power management, in: IEEE Solid State Circuits Conference, February 2009, pp. 456–457.
[7] P. Stolk et al., Modeling statistical dopant fluctuations in MOS transistors, IEEE Transactions on Electron Devices 45 (9) (1998) 1960–1971.
[8] A. Bhavnagarwala et al., The impact of intrinsic device fluctuations on CMOS SRAM cell stability, IEEE Journal of Solid-State Circuits 36 (4) (2001) 658–665.
[9] S. Ramey, Frequency and recovery effects in high-K BTI degradation, in: IEEE International Reliability Physics Symposium, April 2009, pp. 1023–1027.
[10] S. Nassif et al., High performance CMOS variability in the 65nm regime and beyond, in: IEEE Electron Devices Meeting, December 2007, pp. 569–571.
[11] Y. Wang et al., Dynamic behavior of SRAM data retention and a novel transient voltage collapse technique for 0.6V 32nm LP SRAM, IEDM Dig. Tech. Papers, December 2011, pp. 742–744.
[12] J. Kulkarni et al., Capacitive-coupling wordline boosting with self-induced VCC collapse for write VMIN reduction in 22-nm 8T SRAM, in: IEEE Solid State Circuits Conference, February 2012, pp. 234–236.
[13] E. Karl et al., A 4.6GHz 162Mb SRAM design in 22nm tri-gate CMOS technology with integrated active V MIN-enhancing assist circuitry, in: IEEE Solid State Circuits Conference, February 2012, pp. 230–232.
[14] L. Chang et al., An 8T-SRAM for variability tolerance and low-voltage operation in high-performance caches, IEEE Journal of Solid-State Circuits 43 (4) (2008) 956–963.
[15] I.-J. Chang et al., A 32kb 10T subthreshold SRAM array with bit-interleaving and differential read scheme in 90 nm CMOS, in: IEEE Solid State Circuits Conference, February 2008, pp. 388–389.
[16] S. Nalam et al., 5T SRAM with asymmetric sizing for improved read stability, IEEE Journal of Solid-State Circuits 46 (10) (2011) 2431–2442.
[17] M. Chen et al., A 260mV L-shaped 7T SRAM with bit-line (BL) swing expansion schemes based on boosted BL, asymmetric-VTH read-port, and offset cell VDD biasing techniques, in: IEEE Symposium on VLSI Circuits, June 2012, pp. 112–113.
[18] K. Zhang, A SRAM design on 65nm CMOS technology with integrated leakage reduction scheme, in: IEEE Symposium on VLSI Circuits, June 2004, pp. 294–295.
[19] R. Joshi et al., Statistical-aware design for the nm Era, IC Design and Methodology (May) (2009) 207–210.
[20] H. Yamauchi, A discussion on SRAM circuit design trend in deeper nanometer-scale technologies, IEEE Transactions on VLSI Systems 18 (5) (2010) 763–773.
[21] B. Calhoun, A. Chandrakasan, Ultra-dynamic voltage scaling (udvs) using sub-threshold operation and local voltage dithering, IEEE Journal of Solid-State Circuits 41 (1) (Jan. 2006) 238–245.
[22] A. Andrei, M.T. Schmitz, P. Eles, Z. Peng, B.M. Al Hashimi, Quasi-static voltage scaling for energy minimization with time constraints, Design, Automation and Test in Europe, 2005. Proceedings, vol. 1, 7–11 March 2005, pp. 514–519.
[23] Bo Zhai, D. Blaauw, D. Sylvester, K. Flautner, The limit of dynamic voltage scaling and insomniac dynamic voltage scaling, IEEE Transactions on Very Large Scale Integration (VLSI) Systems 13 (11) (2005) 1239–1252.
[24] James Tschanz, Keith Bowman, Steve Walstra, Marty Agostinelli, Tanay Karnik, Vivek De, Tunable replica circuits and adaptive voltage-frequency techniques for dynamic voltage, temperature, and aging variation tolerance, in: 2009 Symposium on VLSI Circuits, 16–18 June 2009, pp. 112–113.
[25] K.A. Bowman, J.W. Tschanz, S.L. Lu, P.A. Aseron, M.M. Khellah, A. Raychowdhury, B.M. Geuskens, C. Tokunaga, C.B. Wilkerson, T. Karnik, V.K. De, A 45 nm resilient microprocessor core for dynamic variation tolerance, IEEE Journal of Solid-State Circuits 46 (1) (2011) 194–208.
[26] A.K. Uht, Going beyond worst-case specs with TEAtime, Computer 37 (3) (2004) 51–56.
[27] M. Fojtik, D. Fick, Yejoong Kim, N. Pinckney, D. Harris, D. Blaauw, D. Sylvester, Bubble Razor: an architecture-independent approach to timing-error detection and correction, in: Solid-State Circuits Conference Digest of

Please cite this article in press as: A.A. Eltawil et al., A survey of cross-layer power-reliability tradeoffs in multi and many core systems-on-chip, Microprocess. Microsyst. (2013), http://dx.doi.org/10.1016/j.micpro.2013.07.008

Technical Papers (ISSCC), 2012 IEEE International, 19–23 February 2012, pp. 488–490.
[28] Amin Khajeh Djahromi, Ahmed M. Eltawil, Fadi J. Kurdahi, Rouwaida Kanj, Cross layer error exploitation for aggressive voltage scaling, ISQED, 2007.
[29] D. Ernst, Nam Sung Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin, K. Flautner, T. Mudge, Razor: a low-power pipeline based on circuit-level timing speculation, in: Proceedings of 36th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 36), 3–5 Dec. 2003, pp. 7–18.
[30] Fadi J. Kurdahi, Ahmed M. Eltawil, Young-Hwan Park, Rouwaida N. Kanj, Sani R. Nassif, System-Level SRAM Yield Enhancement, ISQED 2006, pp. 179–184.
[31] S. Mukhopadhyay, H. Mahmoodi, K. Roy, Modeling of failure probability and statistical design of SRAM array for yield enhancement in nanoscaled CMOS, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 24 (12) (2005) 1859–1880.
[32] A. Bhavnagarwala, X. Tang, J.D. Meindl, The impact of intrinsic device fluctuations on cmos sram cell stability, JSSC (April) (2001).
[33] Predictive Technology Model (PTM). <http://www.eas.asu.edu/~ptm>.
[34] J. Srinivasan, S.V. Adve, P. Bose, J.A. Rivers, The case for lifetime reliability-aware microprocessors, in: Proceedings of the 31st International Symposium on Computer Architecture (ISCA-04), IEEE, June 2004, pp. 276–287.
[35] J. Srinivasan, S.V. Adve, P. Bose, J.A. Rivers, The impact of technology scaling on lifetime reliability, in: Proceedings of the 34th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, IEEE/IFIP, 2004, pp. 177–186.
[36] T. Austin, V. Bertacco, D. Blaauw, T. Mudge, Opportunities and challenges for better than worst-case design, in: Proceedings of the 2005 Asia and South Pacific Design Au- tomation Conference (ASP-DAC'05), ACM, 2005, pp. 2–7.
[37] A.B. Kahng, S. Kang, R. Kumar, J. Sartori, Designing a processor from the ground up to allow voltage/reliability tradeoffs, in: Proceedings of the 16th International Symposium on High Performance Computer Architecture (HPCA), IEEE, 2010, pp. 1–11.
[38] S. Nararanan, J. Sartori, R. Kumar, D.L. Jones, Scalable stochastic processors, in: Proceedings of the Conference on Design, Automation and Test in Europe (DATE 2010), European Design and Automation Association, 2010, pp. 335–338.
[39] J. Sartori, J. Sloan, R. Kumar, Stochastic computing: embracing errors in architecture and design of processors and applications, in: Proceedings of the 14th International Conference on Compilers, Architectures, and Synthesis (CASES 2011), ACM, 2011, pp. 135–144.
[40] M. Hachman, Intel Demos 48-Core Prototype Chip, December 2, 2009. <www.pcmag.com>.
[41] N. Hardavellas, M. Ferdman, B. Falsafi, A. Ailamaki, Toward Dark Silicon in Servers, IEEE Micro, IEEE, July/August 2011, pp. 6–15.
[42] C. Bolchini, A. Miele, C. Sandionigi, A novel design methodology for implementing reliability-aware systems on SRAM-based FPGAs, IEEE Transactions on Computers 60 (12) (December 2011) 1744–1758.
[43] DySyRe: On-Demand System Reliability. <www.desyre.eu>.
[44] Fine-Grain Voltage Tuned Cache Architecture for Yield Management under Process Variations, J. Kong, Y. Pan, S. Ozdemir, A. Mohan, G. Memik, and S. W. Chung, IEEE Trans. VLSI, 2012.
[45] C. Wilkerson et al., Trading off Cache Capacity for Reliability to Enable Low Voltage Operation, Proc. ISCA, 2008.
[46] A. Chakraborty et al., E < MC2 : Less Energy through Multi-Copy Cache, Proc. CASES, 2010.
[47] Fadi Kurdahi, Ahmed Eltawil, Amin K. Djahromi, Mohammad Makhzan, Stanley Cheng, Error Aware Design, in: proceedings of the 10th Euromicro Conference on Digital System Design Architectures, August 2007, pp. 8–15.
[48] Mohammad A. Makhzan (Avesta Sasan), Houman Homayoun Ahmed Eltawil, Fadi Kurdahi, Process Variation Aware SRAM/Cache for aggressive Voltage-Frequency Scaling, DATE 2009.
[49] Avesta Sasan, Houman Homayoun, Ahmed Eltawil, Fadi J. Kurdahi, A Fault Tolerant Cache Architecture for Sub 500mv Operation Resizable Data Composer Cache (RDC-Cache), CASES 2009, Grenoble, France.
[50] M.A. Makhzan, A. Khajeh, A. Eltawil, F. Kurdahi, Limits on voltage scaling for caches utilizing fault tolerant techniques, in: proceedings of 25th International Conference on Computer Design ICCD, 2007, pp. 488–495.
[51] M. de Kruijf, S. Nomura, K. Sankaralingam. Relax: an architectural framework for software recovery of hardware faults, in: Proceedings of ISCA '10, ACM, New York, NY, USA, 2010, pp. 497–508.
[52] A. Heinig, V.J. Mooney, F. Schmoll, P. Marwedel, K. Palem, M. Engel. Classification-based improvement of application robustness and quality of service in probabilistic computer systems. in: Proceedings of ARCS'12, Springer-Verlag, Berlin, Heidelberg, 2012, pp. 1–12.
[53] G.A. Reis, J. Chang, D.I. August, R. Cohn, S.S. Mukherjee. Configurable transient fault detection via dynamic binary translation, in: Proceedings of the 2nd Workshop on Architectural Reliability, 2006.
[54] G.A. Reis, J. Chang, N. Vachharajani, R. Rangan, D.I. August. SWIFT: Software implemented fault tolerance, in: Proceedings of CGO'05, IEEE Computer Society, Washington, DC, USA, 2005, pp. 243–254.
[55] A. Sampson, W. Dietl, E. Fortuna, D. Gnanapragasam, L. Ceze, D. Grossman. EnerJ: approximate data types for safe and general low-power computation, in: Proceedings of PLDI '11, ACM, New York, NY, USA, 2011, pp. 164–174.
[56] F. Schmoll, A. Heinig, P. Marwedel, M. Engel, Improving the fault resilience of an H.264 decoder using static analysis methods, ACM Transactions on Embedded Computing Systems (TECS), (in press).
[57] F.J. Kurdahi, A. Eltawil, K. Yi, S. Cheng, A. Khajeh, Low Power application-aware multimedia system design by aggressive voltage scaling, IEEE Transactions on VLSI 18 (5) (2010).
[58] Ahmed M. Eltawil, Eugene Grayver, Hanli Zou, Jean Francois Frigon, Gennady Poberezhskiy and Babak Daneshrad, Dual antenna UMTS mobile station transceiver ASIC for 2 Mbps data rate, in: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 46(February) (2003) 146–47.
[59] A.K. Djahromi, A.M. Eltawil, F.J. Kurdahi, Fault tolerant approaches targeting ultra low power communications system design, in: Proc. 2007 65th IEEE Vehicular Technology Conference VTC2007-Spring, April 2007, pp. 2600–2604.
[60] L. Leem, H. Cho, J. Bau, Q.A. Jacobson, S. Mitra. ERSA: Error resilient system architecture for probabilistic applications, In DATE, 2010.
[61] B. Noble, System support for mobile adaptive applications, IEEE Personal Communications 7 (1) (2000).
[62] B. Noble, M. Satyanarayanan, Experience with adaptive mobile applications in Odyssey, Mobile Networks and Applications 4 (1999).
[63] L. Wang, N.R. Shanbhag, Low-power filtering via adaptive error-cancellation, IEEE Transactions on Signal Processing [see also Acoustics, Speech, and Signal Processing, IEEE Transactions on] 51 (2) (2003) 575–583.
[64] N. Shanbhag, Reliable and energy-efficient digital signal processing, in: Design Automation Conference, 2002. Proceedings, vol. 39, 2002, pp. 830–835.
[65] R. Abdallah, N. Shanbhag, Error-resilient Low-Power Viterbi Decoder Architectures, IEEE Transactions on Signal Processing 9 (2009) (vol. 57 #12).
[66] Todd M. Austin, DIVA: a reliable substrate for deep submicron microarchitecture design, in: Proceedings of the 32nd Annual ACM/IEEE International Symposium on Microarchitecture (MICRO 32). IEEE Computer Society, Washington, DC, USA, 1999, pp. 196–207.

Ahmed M. Eltawil is an Associate Professor at the University of California, Irvine. He received the Doctorate degree from the University of California, Los Angeles, in 2003 and the M.Sc. and B.Sc. degrees (with honors) from Cairo University, Giza, Egypt, in 1999 and 1997, respectively. Since 2005, he has been with the Department of Electrical Engineering and Computer Science, University of California, Irvine. He is the founder and director of the Wireless Systems and Circuits Laboratory. (http://newport.eecs.uci.edu/~aeltawil/). His current research interests are in low power digital circuit and signal processing architectures for wireless communication systems where he has published more than 90 technical papers on the subject, including four book chapters. Dr Eltawil has been on the technical program committees and steering committees for numerous workshops, symposia and conferences in the area of VLSI, and communication system design. He has received several distinguished awards, including the NSF CAREER award in 2010 supporting his research in low power systems.

Michael Engel studied Computer Engineering and Applied Mathematics at the University of Siegen, Germany. He received his Dr. rer. nat. degree in Computer Science in 2005 from the University of Marburg, Germany. In 2006 and 2007, Dr. Engel held an interim position as professor for operating systems at Technical University of Chemnitz, Germany. Since September 2007, Dr. Engel is an assistant professor at the Faculty for Computer Science of TU Dortmund. His research interests include dependability, energy optimization and operating systems for embedded systems

Bibiche M. Geuskens received the B.S. degree in electrical engineering from the Vrije Universiteit Brussel, Brussels, Belgium in 1992 and the M.S. and Ph.D. degrees in electrical engineering from Rensselaer Polytechnic Institute, Troy, NY in 1993 and 1997, respectively. In 2006, she joined Intel Labs (IL) at Intel Corporation in Hillsboro, OR, as a Staff Research Scientist. From 1997 to 2006, she worked as a Staff/Senior Component Design Engineer in the Memory Design Unit of the Microprocessor Design Division at Intel in Hillsboro, OR. She was responsible for the design, implementation and validation of numerous circuit blocks. Since joining IL in 2006, her research has focused on the development of low power circuit design techniques for on-chip memories, CMOS bio sensor design applications and on-chip power delivery circuit solutions. Her current research interests include low power SoC design and efficiency.

Please cite this article in press as: A.A. Eltawil et al., A survey of cross-layer power-reliability tradeoffs in multi and many core systems-on-chip, Microprocess. Microsyst. (2013), http://dx.doi.org/10.1016/j.micpro.2013.07.008

Amin Khajeh Djahromi was born in Birmingham U.K. He received his Bachelor of Science in Electrical Engineering and Communication from Shiraz University, Iran in 2002 and Master of Science in Electrical Engineering from University of Texas at Arlington in 2005, and PhD in EECS from University of California Irvine in 2010. He was an intern in research and development division of Siemens Company in summer of 2002 and He later joined Siemens as a research staff member from 2002 to 2003. Dr. Khajeh's research interests include low power SoC design, design of low power high performance circuits for communication and multimedia applications, cross-layer optimization, fault tolerant adaptation, high performance high yield memory design, and low power DSP design where he has published more than 30 technical papers on these subjects. He has 5 US patents pending. Dr. Khajeh was the recipient of CPCC fellowship in 2006 and 2007, 2010 from Center for Pervasive Communications and Computing, CA. He was with Qualcomm low power DSP team from 2010 to 2012 researching and implementing advance low power technique for DSP cores for wireless multimedia applications. He is currently a research scientist at SoC Design Lab at Intel Labs researching ultra-low power SoC design, low power integration methods, and near threshold voltage design.

Fadi Kurdahi received his PhD from the University of Southern California in 1987. Since then, he has been a faculty at the Department of Electrical & Computer Engineering at UCI, where he conducts research in the areas of Computer Aided Design of microchips, and serves as the Director for the Center for Embedded Computer Systems (CECS). His research was disseminated in over 150 publications and has received numerous accolades. He serves on a several journal boards and conference committees. He also received the Distinguished Alumnus award from the American University of Beirut in 2008. He is a Fellow of the IEEE and the AAAS.

Peter Marwedel studied physics at the University of Kiel, Germany. He received a Dr. rer. nat. degree in physics in 1974 and a Dr. habil. degree in computer science in 1987. Since 1989, he is holding a chair for computer engineering and embedded systems at the computer science department of TU Dortmund. He is also chairing ICD, a local spin-off. His research interests include design automation for embedded systems, in particular the generation of efficient embedded and cyber-physical system software. Focus is on energy efficiency, timing predictability, reliability and tradeoffs between design objectives. Dr. Marwedel is an IEEE Fellow and recipient of the EDAA Lifetime Achievement Award in 2013.

Smail Niar (University of Valenciennes & CNRS, France) received his PhD in computer Engineering from the University of Lille in 1990. Since then, he has been professor at the University of Valenciennes. He is leader of the "Mobile & Embedded Systems" research group at the *Laboratory* For Automation, Mechanical and Computer Engineering, a joint research unit between CNRS and university of Valenciennes. He is member of the European Network of Excellence on "High Performance and Embedded Architecture and Compilation" (Hipeac). His research interests are in multi-processor system-on-chip (MPSoC) architectures, power/energy consumption optimization, dynamically reconfigurable embedded systems (FPGA), simulation acceleration techniques for MPSoC design space exploration, and hybrid MPSoC in Intelligent Transportation Systems.

Mazen A.R. Saghir is an Associate Professor of Electrical and Computer Engineering at Texas A&M University at Qatar. He received his BE in Computer and Communication Engineering from the American University of Beirut in 1989, and his M.A.Sc. and Ph.D. in Electrical and Computer Engineering from the University of Toronto in 1993 and 1998, respectively. His research interests include reconfigurable computing, computer architecture, compilers and EDA tools, and embedded systems design. He is a senior member of the IEEE.

Please cite this article in press as: A.A. Eltawil et al., A survey of cross-layer power-reliability tradeoffs in multi and many core systems-on-chip, Microprocess. Microsyst. (2013), http://dx.doi.org/10.1016/j.micpro.2013.07.008