Adaptive Voltage Over-Scaling for Resilient Applications

Philipp Klaus Krause
Goethe University of Frankfurt
Institute of Computer Science
D-60325 Frankfurt, Germany
Email: philipp@informatik.uni-frankfurt.de

Ilia Polian
University of Passau
Department of Informatics and Mathematics
D-90342 Passau, Germany
Email: ilia.polian@uni-passau.de

***Abstract*—We present an energy-reduction strategy for applications which are resilient, i. e. can tolerate occasional errors, based on an adaptive voltage control. The voltage is lowered, possibly beyond the safe-operation region, as long as no errors are observed, and raised again when the severity of the detected errors exceeds a threshold. Due to the resilient nature of the applications, lightweight error detection logic is sufficient for operation, and no expensive error recovery circuitry is required. On a hardware block implementing texture decompression, we observe 25% to 30% energy reduction at negligible quality loss (compared to the error introduced by the lossy compression algorithm). We investigate the strategy's performance under temperature and process variations and different assumptions on voltage-control circuitry. The strategy automatically chooses the lowest appropriate voltage, and thus the largest energy reduction, for each individual manufactured instance of the circuit.**

I. Introduction

Voltage scaling, i. e. lowering the power supply voltage V_{DD} during system operation, is a key energy-reduction methodology [1], [2]. Since gate delays increase upon voltage reduction, conventional *dynamic voltage and frequency scaling* (DVFS) approaches complement voltage scaling by a simultaneous decrease of frequency, thus reducing the system's performance. In DVFS, the system always operates in a "stable" voltage-frequency operating point, where the critical-path delay is never exceeded, even taking some safety margins due to possible temperature or process parameter variations into account. This means that the circuits are conservatively designed to meet the timing constraints even if the temperature exceeds its typical value and the delays on the critical path are elevated due to statistical process variations.

This conservatism implies that the majority of actual manufactured circuits cannot utilise the full potential of voltage scaling most of the time: even though V_{DD} could be lowered further without compromising the system's stability, this is not done, since voltage control has no temperature or process-variation information and thus cannot decide whether a further V_{DD} decrease would be safe. As this paper shows, significant energy savings are prevented by assuming the worst case.

We present an adaptive voltage over-scaling (AVOS) strategy for *resilient applications*, which can tolerate error-induced output deviations. The concept of application resilience has been known under headings such as error tolerance [3], application-level correctness [4] or imprecise computation [5]. Resilient applications are found in fields like multimedia, where a human end-user may not notice small deviations in e. g. images calculated by hardware blocks affected by errors [6], and artificial intelligence where algorithms including belief propagation and support vector machines have intrinsic mechanisms to deal with erroneous or uncertain data [4]. The AVOS strategy lowers voltage, while (unlike DVFS) leaving frequency unchanged. The circuit's flip-flops have a lightweight Razor-style error detection circuitry [7], which reports small-delay errors. All detected errors are accumulated in a time-decaying error sum counter, but no error recovery is performed. Once the error counter exceeds a pre-defined threshold, V_{DD} is raised again and is lowered back when no errors have been observed for a while. By doing so, every particular manufactured circuit instance tends to be operating very close to its individual boundary between error-free and erroneous operation. The strategy adapts the circuit with respect to delays of circuit paths sensitised by the input data being processed as well as the temperature, and can react to short-term changes in these parameters. If the strategy is allowed to choose voltages which exceed the nominal V_{DD}, it can even ensure the circuit's operation outside its specifications.

We perform a series of simulations on an ASIC implementation of texture decompression algorithm ftc1 [8], which is an instance of a typical embedded application that is both resilient and energy-constrained. Compared to its version with no AVOS, we observe energy savings of 25% to 30%. The impact on the quality of the images produced is negligible, as it's dwarfed by the quality loss due to the compression error itself, even though we don't employ any extra circuitry for recovery. We show the efficiency of our technique for a number of scenarios with inter-die and intra-die variations and for a range of temperatures.

The remainder of this paper is organised as follows: Section II presents the AVOS strategy in more detail. Section III describes the ftc1 block used for experiments and discusses the hardware implementation of error detection and voltage control. Section IV focuses on the flow used for evaluation, and results are reported in section V. Related work is summarized in section VI. Possible directions for future work are discussed in section VII. Section VIII concludes the paper.

II. Adaptive Voltage Over-Scaling

In AVOS the voltage at which a circuit operates is changed dynamically depending on the circuit's behaviour. Among a set of possible voltages a desired one is selected by a *voltage control strategy*. Conservative assumptions about other parameters, such as process variations and temperature are not necessary, allowing aggressive lowering of voltage and high energy savings.

Assuming a synchronous circuit let V be the set of voltages, Z the set of possible observations from the circuit's behaviour and S the internal state of the voltage control. In this work we assume $V = \{V_0, \ldots V_h\}, Z$ and S to be finite sets. Then voltage control is a Mealy Machine with transfer function $f : S \times Z \to S \times V$. One way of observing the circuit's behaviour is by trying to detect delay errors caused by too low voltage and report an error severity as a binary number of $\log_2 n$ bits, resulting in an *error function* $e : Z \to N = \{0, \ldots, n-1\} \subseteq \mathbb{N}$, and let the voltage control rely on these reported values: $v : S \times N \to S \times V, f = v(\cdot, e(\cdot))$.

Function e is discussed in section II-A, v in section II-B.

A. Error Function

In a circuit there are multiple locations where errors can occur. Similar to prior work [9], we assign each location i a *severity* e_i. Severity of 0 represents the absence of errors, with higher numbers indicating more severe errors. These severities are generally application-specific. The circuit considered in this paper calculates pixel values. The severity of an error is defined as the numerical difference between the pixel value in the erroneous and the error-free case.

In order to simplify error-detecting hardware, we employ the following technique for the (unlikely) case that multiple pixel values are corrupted simultaneously. Let the individual severities of the affected outputs be $e_1, \ldots, e_m \in N$. The accurate overall severity calculation would sum up all these severities. Instead of implementing the expensive circuitry for calculating the sum, we use the bit-wise OR of the e_i's to yield the overall severity. It can be shown that

978-3-9810801-7-9/DATE11/©2011 EDAA

$$\forall (e_1, \ldots, e_m) \in N^m \colon \max_{i=1,\ldots,m} e_i \leq \text{OR}(e_1, \ldots, e_m) \leq \sum_{i=1}^{m} e_i.$$

holds. Hence, this approximation is close enough to actual severity while being cheap in terms of gate count and delay at the same time.

B. Voltage Control Strategy

For voltage control a combination of a time-decaying *error counter* E and a voltage change *cool-down counter* Γ has been chosen. A configurable *error threshold* E_t, cool-down counter *reset value* Γ_0 and *threshold* Γ_t allow for a trade-off between quality and adaption speed to varying parameters, such as temperature.

Voltage control operation is shown in figure 1. Both E and Γ use saturating arithmetic. At each clock E is decremented when no error is detected and Γ is decremented when no voltage change occurs. When an error occurs its severity is added to E. Whenever the voltage changes Γ is reset to Γ_0. When $E = \Gamma = 0$ the voltage is decreased. When the last voltage change was a decrease and $E \geq E_t$ the voltage is increased. When the last voltage change was an increase and $E \geq E_t$ and $\Gamma \leq \Gamma_t$ for an implementation defined cool-down threshold Γ_t the voltage is increased. Γ_0 must be large enough to prevent too rapid voltage fluctuations.

For applications where a higher error rate is tolerable, the condition of $E = 0$ for voltage decrement could be relaxed to $E \leq E_0$ for a configurable threshold E_0, resulting in potentially higher energy savings. This energy-vs.-quality trade-off has been explored in classical voltage over-scaling literature [10]. Our proposed strategy aims at getting as close as possible to an individual chip's limits of the error-free operation and derives its energy savings from eliminating the safety margin. We accept only occasional errors when the chip's limits are exceeded and try to return to stable operation as soon as possible, resulting in very low quality loss. Selecting larger values of E_t and E_0 would allow to trade further energy savings for a systematic quality deterioration (as done in [10]) on top of the energy reduction reported.

III. The ftc1/DXT1 decoder

In computer graphics achieving high visual quality typically requires high-resolution textures. Texture compression achieves higher graphics quality with given memory and bandwidth or reduced memory and bandwidth consumption without degrading quality too much. GPUs have dedicated hardware for texture decompression.

DXT1 [11] is a texture compression system supported by nearly all 3D graphics hardware today. It is required by DirectX and available on most OpenGL implementations. ftc1 [8] is a state of the art texture compression system derived from DXT1 that offers higher image quality at the same compression ratio. In a combined ftc1/DXT1 implementation most of the hardware can be shared, so that compared to a plain DXT1 decoder the additional amount of hardware required to implement DXT1 is minimal ($\approx 11\%$ in gate count).

The decoder used in this work has two inputs, a 64 bit compressed data block representing 16 pixels in the texture (called texels) and a texel selection signal indicating which of these 16 pixels to retrieve. The output is a 32 bit RGBA pixel value.

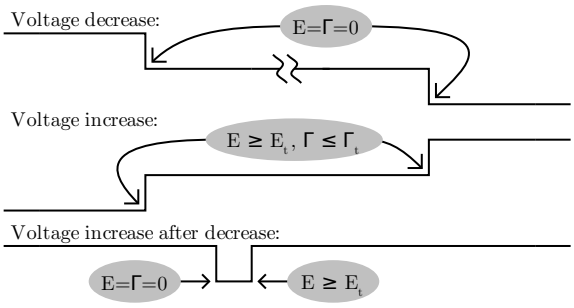

Fig. 1. Voltage control strategy

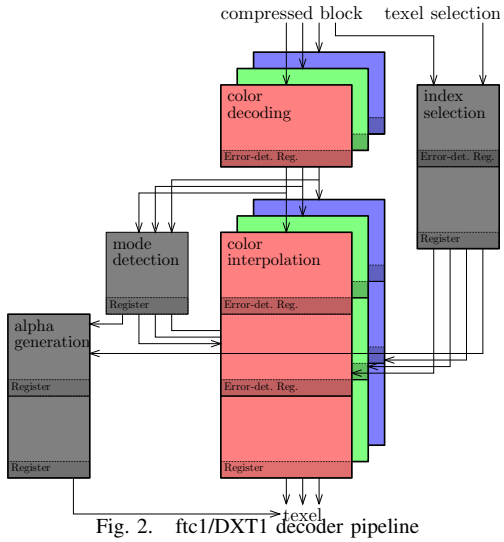

Fig. 2. ftc1/DXT1 decoder pipeline

For this work a pipelined ftc1/DXT1 decoder has been implemented (figure 2). It consists of four pipeline stages. The first stage performs colour endpoint decoding (and is the only stage added for ftc1 compared to a DXT1 decoder). The last three stages interpolate colour, generate additional colours from the endpoint colours and then choose one based on the index.

The decoder has been implemented using hierarchical Verilog, with error detection at registers after critical pipeline stages.

A. Error Detection

The error detection has been realised similar to Razor [7] by adding delay error detection at the registers in the pipeline. Figure 3 shows a second flip-flop driven by a second clock delayed by time δ and the same input line. When a signal change is delayed it doesn't arrive in time for the rising main clock, but arrives before the delayed clock (i. e. the delay is less than δ) the flip-flops contain different values. The output of the XOR gate becomes true and the delay error is detected (figure 4).

When the clock delay δ is too short, many delay errors (when the delayed signal arrives after the rise of the delayed clock) will not be detected. When the clock delay δ is too long, false positives may occur (a delay error cannot always be distinguished from a regular signal change in the next clock cycle). This problem is worst when a signal depends on inputs from the previous stage through both long and shorts paths, such as the most significant bit output of a carry-ripple adder. There are multiple ways to mitigate this problem, e. g. adding delay elements such as additional buffers or latches that gate the signal [12]; this could be done automatically in synthesis tools; chip area is increased by the additional gates required. In this work no such additional measures have been taken. The clock delay δ has been chosen to match the minimum delay of the shortest path at the input of a flip-flop with error detection. δ is changed dynamically with the supply voltage to detect more errors.

We found that AVOS is only minimally affected by error detection imperfections: if the error detection logic misses an error, it results in a (slight) deterioration of the output quality; if it mistakenly reports an error, an unnecessary voltage raise is performed and soon taken back, resulting in a minimal energy cost. Therefore, lightweight error detection circuitry with a smaller energy consumption than in the original Razor approach targeting general-purpose, non-resilient applications, can be employed.

The reported error severity e_i is derived from the error's impact on the decoder's output. For an n-bit pipeline register holding a value that will end up in a colour channel of an output texel, we directly map the error outputs from the register's bits to the bits of the severity (i. e. errors in the most significant bit of the colour channel are being considered as much more severe than errors in the least significant bit).

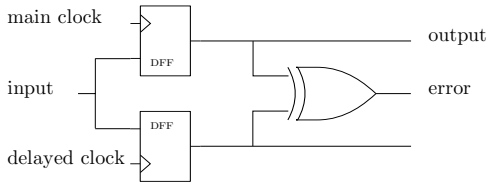

Fig. 3. Delay error detection hardware

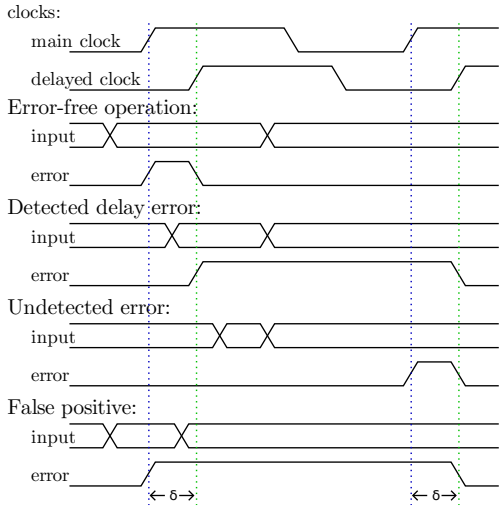

Fig. 4. Operation of error detection

B. Voltage Control Implementation

The implementation of the strategy from section II-B has a (registered) input for errors and an output to select a supply voltage. Error counter E and a voltage change cool-down counter Γ are kept in registers. Both are decremented at each clock cycle until they reach 0 unless an error occurs. When an error occurs its severity (as reported by e above) is added to the error counter E. When the error counter reaches the configurable threshold E_t (and Γ is low enough or the last voltage change was a decrement), it is reset to 0 and the voltage is increased. When both the error counter E and the cool-down counter Γ are 0 and there is no error, the voltage is decreased. Whenever the voltage is changed, Γ is reset to its configurable maximum value Γ_0.

AVOS strategy used for experiments selects the voltage in steps of $\frac{1}{20}$V from $V_0 = 1.15$V to $V_{15} = 1.9$V (the nominal voltage for the process used is 1.8V). All voltage changes are assumed to take place immediately. We will evaluate this assumption and the number of voltages used in section V-B.

IV. Experimental Setup

An implementation of the ftc1/DXT1 decoder has been simulated at different temperatures and with different input images to evaluate energy savings and image quality reduction due to AVOS. The simulation flow (figure 5) consists of free software only.

To obtain gate delays OSU standard cells [13] have been simulated using gnucap [14], a free SPICE, at different supply voltages and temperatures for a 0.18 μm TSMC process and an output load of 0.1pF. From the resulting delay data further values have been generated to simulate process variations: *Inter-die process variations* affect all transistors on a die uniformly. They were modelled by multiplying all gate delays by a constant 0.9 or 1.1 (according to [15], [16] a standard deviation of about 10% of the mean is typical for the 0.18 μm TSMC process). *Intra-die variations* can be modelled by multiplying the delays of individual gates with different, Gaussian distributed constants. These variations are typically smaller than inter-die variations (a standard deviation of about 5% of the mean has been observed for 0.18 μm CMOS technologies [17], including the TSMC one [18]).

The Verilog implementation of ftc1/DXT1 has been compiled into BLIF-MV using vl2mv [19], the BLIF-MV was flattened and converted to BLIF using VIS [20]. Combinatorial optimisation and technology mapping has been done using ABC [21]. Using the delay values from the gnucap simulation the design has been simulated at gate level using Icarus Verilog [22] at 130MHz, the highest frequency at which it operates error-free at 85°C, 1.8V assuming the absence of any process variations. Note that this margin setting is very aggressive; a realistic chip designer would consider process variations, voltage fluctuations, modelling uncertainties and aging.

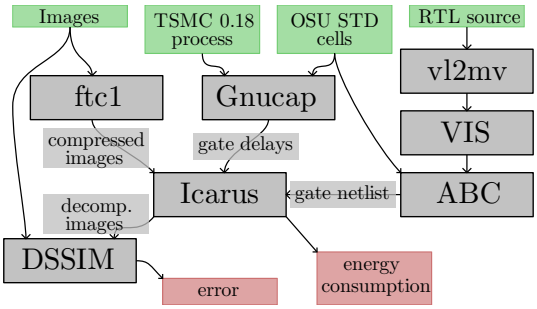

Fig. 5. Experimental setup flow

A set of 50 images has been used as inputs in the experiments. It includes images typically used for evaluation of image processing systems, such as the lena and lorikeet images and textures from the strategy game Glest.

For each image the difference between the uncompressed image and the ftc1-compressed and decompressed image has been measured both for error-free decompression and decompression using AVOS, which is affected by delay errors. The difference has been measured using structural dissimilarity (DSSIM) [23], a distance measure derived from structural similarity (SSIM) [24], which corresponds well to perceived visual quality. The C++ implementation [25] used in [8] has been used to measure DSSIM. Results obtained using the classical error measures mean absolute error and mean squared error [26] showed the same tendencies as those obtained using DSSIM.

Since energy consumption is proportional to square of the supply voltage V_{DD}^2 [27] the energy consumption has been measured in cycles multiplied by V_{DD}^2: Let $V_0, \ldots, V_h$ be the possible supply voltages, and c_i the number of clock cycles during which the circuit has been operating at supply voltage V_i. Then the energy consumption is assumed to be proportional to $\sum_{i=0}^{h} c_i V_i^2$. For simulation parameters $E_t = 3, \Gamma_0 = 2080, \Gamma_t = 2047$ have been used.

V. Results

Figure 6 shows the distribution of errors in a part of one image (out of 50) with an outstandingly large number of errors. Most errors (lower right cluster) are barely noticeable to a human observer, some (the two in the middle) are noticeable but not very prominent. Only one (in the upper left) is so intense that it's immediately noticed even by a casual observer. Even in this unusually error-rich part of the image compression artifacts are more noticeable than the AVOS-induced errors.

A. Idealized Nominal Case

For the nominal case the ftc1 decoder has been simulated at 45°C assuming the absence of process variations and a response time of 0 (i.e., all voltage changes take place immediately). Figure 7 shows energy consumption of the ftc1 decoder with AVOS relative to a ftc1 decoder without VOS and the total errors (i. e. the result of decompression versus the original uncompressed image) in DSSIM at a temperature of 45° C. At this typical operating point we see a reduction in energy consumption of 22 to 40%. Recall that we assume a very aggressive safety margin; most real designers would choose a much larger margin and the energy savings would be much higher. The error introduced by AVOS is negligible compared to the error introduced by the image compression algorithm. Except for two images the total error is still far smaller than the error that would have been introduced by using the DXT1 compression algorithm, the de-facto standard for compression of RGB textures.

The average reduction in energy consumption over all considered images was 30% at quality loss of 0.0624. As mentioned in Section

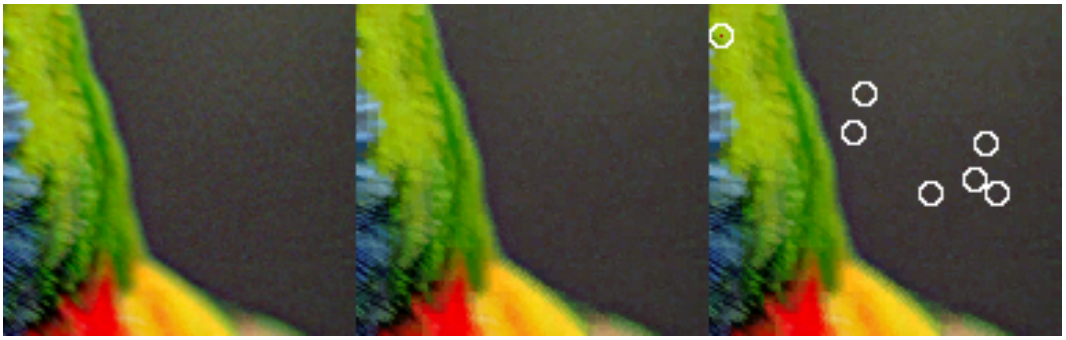

Fig. 6. An outstandingly error-rich region from the lorikeet image. From left to right: Uncompressed image, ftc1 without VOS, ftc1 with AVOS at 45°C. Errors are encircled.

III-B, we repeated the experiment using four and eight voltage levels instead of 16 to trade off smooth voltage control for hardware complexity. For eight V_i ($V_0 = 1.2$V through $V_7 = 1.9$V in 0.1V steps), the energy consumption was reduced by 27% and the quality loss was 0.0626. Using four voltage levels ($V_0 = 1.2$V through $V_3 = 1.8$V in steps of 0.2V), these numbers were 23% and 0.0626, respectively. We observed similar trends in other measurements and only report results for 16 voltage levels from here on.

B. Voltage Control Parameters

In this section, we study the influence of the number of V_{DD} levels and the response times of the voltage-control circuitry on the results. The results are summarized in Table I. A large number of intermediate voltages reduces the absolute difference between V_i's and thus prevent dI/dt during switching. Note that the AVOS strategy enforces some time between transitions, such that the voltage is always increased or decreased smoothly. The first three rows of Table I show that the effectiveness of AVOS is barely affected by the number of voltages, so that the ultimate choice can be guided by the resolution of the available voltage regulator and the severity of dI/dt issues.

The transitions between voltage levels will take some time in practice. This means that, once the AVOS strategy decides to change V_{DD}, the circuit will have to operate at the old level for some number of cycles given by the response time of the external voltage regulator module. In [28], increase times of 40 μs and decrease times of 5.5 ms are reported. Results of AVOS incorporating response time are shown in the final row of Table I. We assumed negligible increase time and decrease times of 2^{16} cycles or 504 microseconds. It can be seen that the quality stays almost constant (recall that quality loss refers to the combined effect of AVOS and lossy compression) while energy consumption worsens slightly. Note that, unlike DVFS, where transitions are expensive due to PLL relock times requiring

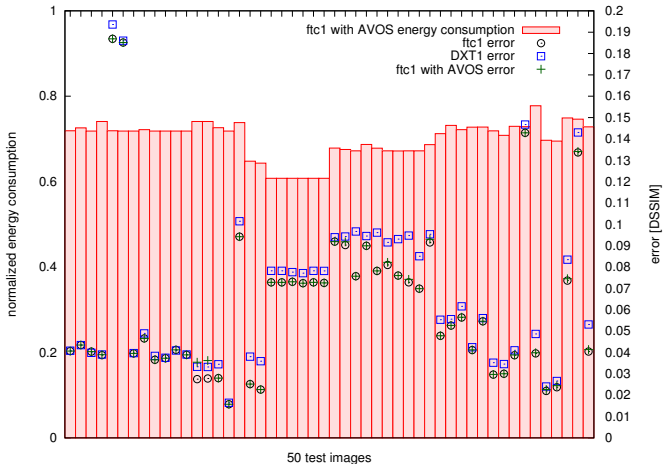

Fig. 7. Error/energy consumption for 50 images at 45°C

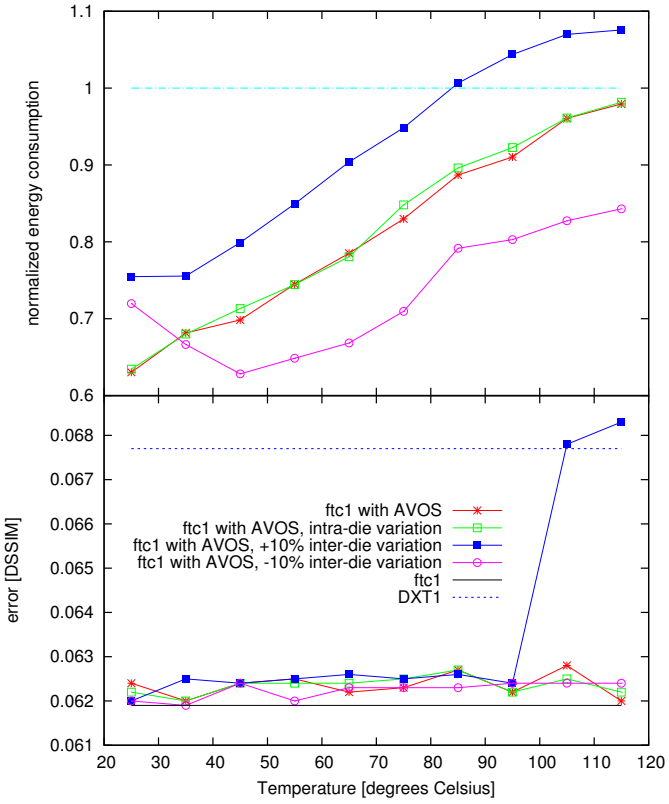

Fig. 8. Average energy consumption and error for the 50 test images

pipeline stalls of hundreds to thousands of clock cycles AVOS keeps the frequency constant and thus transitions are much cheaper. Interestingly, the adaptive mechanism allows for somewhat imprecise transitions. If, for instance, the voltage regulator sets V_{DD} to 1.62V instead of 1.65V, AVOS will monitor whether errors show up and simply switch to the next level.

C. Temperature and Process Variations

The average energy consumption and quality loss over the 50 test images is shown in figure 8 over a 25°C to 115°C temperature range and for several process variation scenarios. We see that with no process variations we get energy savings at all temperatures, with

TABLE I
AVERAGE ENERGY CONSUMPTION AND QUALITY LOSS

Voltage levels	Response time [μs]		Energy cons.	Qual. loss
	Increase	Decrease	reduction [%]	[DSSIM]
16 (1.15V to 1.9V)	0	0	30	0.0624
8 (1.2V to 1.9V)	0	0	27	0.0626
4 (1.2V to 1.8V)	0	0	23	0.0626
16 (1.15V to 1.9V)	0	504	28	0.0619

an energy saving of over 25% at 45°C. This is similar for intra-die process variations (Gaussian, $\sigma = 0.05\mu$). For inter-die variations reducing all gate delays by 10% we see that the adaptability of our AVOS can achieve even higher energy savings (we also see a small rise in energy consumption at low temperatures due to false positives in the error detection causing unnecessary voltage increases). For all these scenarios the additional error introduced by AVOS (less than 0.001 in terms of DSSIM) is much smaller than the one introduced by the ftc1 compression algorithm (about 0.062) and the total error is much smaller than the error introduced by the standard DXT1 texture compression algorithm.

A special case occurs for the chip with inter-die variations causing a 10% increase in gate delays. Without AVOS it would not be able to operate reliably over the specified temperature range (unacceptable DSSIM error of about 0.137 at 85°C). With AVOS we see it operating reliably at all temperatures up to 85°C and still get substantial energy savings over most of the temperature range. At 85°C a small increase in energy consumption (due to raising V_{DD} above 1.8V) over a chip with no AVOS makes reliable operation possible. At even higher temperatures, outside the specified range the error gets larger, but is still comparable to the DXT1 error.

The results of 32 Monte Carlo simulations of intra-die process variations are shown in figure 9 for the lorikeet image (the results are similar for other images). The savings in energy consumption are consistently at about 25%, independent of the process variations. On the other hand the intra-die variations do affect the additional error introduced by AVOS. Still, this error is far less than 1% of the error introduced by the ftc1 compression algorithm in all cases, resulting in a total error much smaller than that of the DXT1 algorithm.

VI. Related Work

Several techniques have been published to take advantage of the circuit operation outside the safety margins. The *Razor* approach [7] lowers V_{DD} adaptively. The flip-flops in the circuit are equipped with error detection logic. Once the voltage has been lowered too much, the circuit starts producing small-delay faults which are detected. Recovery is initiated, and the voltage is raised again. Error recovery may involve techniques such as pipeline flushing which result in performance loss. Furthermore, the error detection and -recovery logic itself can consume a significant portion of the energy savings obtained by over-scaling. Finally, the requirements on the error detection logic are high, as it may not miss any errors; reporting a false positive comes with a recovery cost.

In the *CRISTA* [29] approach additional logic detects the activation of paths that under process variation could potentially be critical. In these cases the pipeline is stalled, switching to two-cycle operation. It thus comes with both a performance and area overhead (18%).

Voltage over-scaling (VOS) [30], [31], [10] deliberately lowers V_{DD} to a value for which the circuit is known to occasionally produce erroneous outputs. Like AVOS, VOS is applicable to resilient applications and has also been considered for RF systems [32]. Voltage over-scaling in its current form is not adaptive: it explores the trade-off between energy saving and output quality reduction for a number of V_{DD} values but it does not dynamically switch between different voltages. VOS can be complemented by synthesis measures. In [33], the path-length distribution is considered as an own optimization objective. In the application-specific approach in [34], a color-interpolation block is modified such that delays induced by over-scaling can only affect "less critical" parts of the circuit. This is achieved by both modifying the algorithm and applying gate-level resynthesis.

Figure 10 compares VOS and AVOS under process variations. It shows hypothetical energy-voltage and quality-voltage characteristics for three manufactured instances of the same circuit. With rising voltage, the energy consumption will increase roughly quadratically, and the quality will improve until it reaches maximum when no errors show up any more. Due to process variations, the delays of different gates will be different in the three circuits, leading to slightly different characteristics. In VOS, the voltage is set to a pre-defined value, resulting in different and unpredictable energy consumption

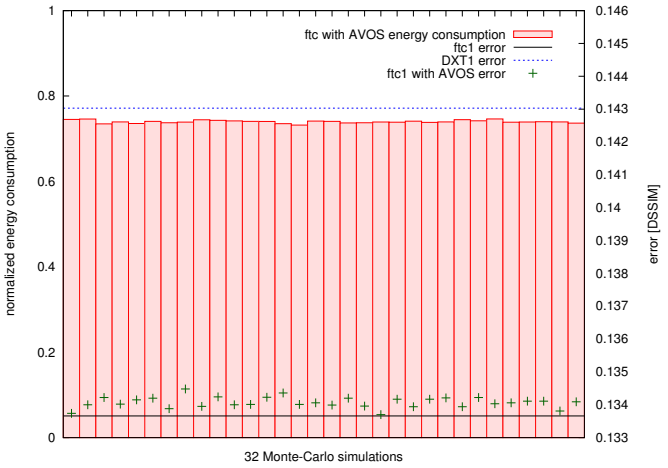

Fig. 9. Gaussian ($\sigma = 0.05\mu$) intra-die delay variations at 45°C using the lorikeet image

and quality deterioration among the three circuits. In AVOS, V_{DD} is adjusted such as to achieve negligible quality deterioration at the minimal individual energy consumption for a particular circuit.

In BiVOS [9], different voltages are applied to different circuit elements. The work focuses on errors due to a noise mechanism which affects a circuit element independently from other elements, rather than on delay effects distributed over a path. Using multiple voltages is shown to significantly improve the scheme's efficiency compared with the uniform-voltage case. In [35], the BiVOS idea is extended to delay effects. Employing multiple voltages, an energy saving of 21.7% is reported for calculating an image with no visible artifacts, while uniform-voltage VOS with the same energy efficiency leads to a significantly deteriorated image.

Table II summarizes the key characteristics of the methods discussed in this section and AVOS. Only Razor and AVOS are adaptive and thus can adjust themselves to aging effects and changing ambient parameters such as temperature. DVFS has performance overhead due to reduced frequency, while Razor and CRISTA must occasionally insert pipeline stall cycles. In contrast, AVOS has no performance overhead. DVFS, Razor and CRISTA are designed to avoid any errors on the circuit outputs while VOS and BiVOS provide a trade-off between quality deterioration (which can be significant) and energy saving. In AVOS, occasional errors are allowed, however the adaptation strategy keeps their number and magnitude in check, operating each circuit very close to its individual limit and achieving maximal energy saving with nearly-zero quality deterioration. All techniques except BiVOS target delay effects distributed along paths. AVOS does not require expensive analog circuitry for frequency control and avoids overhead for providing multiple V_{DD} levels. Comparing AVOS with Razor, AVOS does not need error-correction logic, and its error-detection logic can be simpler because it may occasionally fail to detect errors. Moreover, Razor requires short-path aware synthesis to avoid the false-positive problem [12] while AVOS has been shown to perform well when this problem is not addressed explicitly. In summary, AVOS appears to be a viable alternative

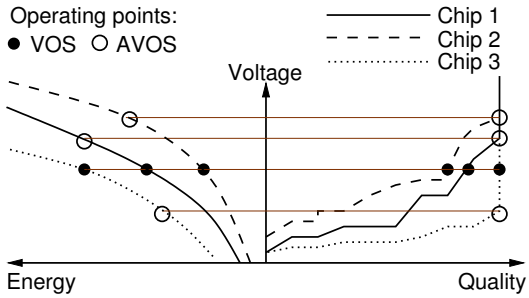

Fig. 10. VOS vs. AVOS under process variations

TABLE II
COMPARISON OF AVOS WITH RELATED TECHNIQUES

Technique	Adaptive	Performance overhead	Quality impact	Error mechanism	Extra circuitry (all schemes require V_{DD} control)
DVFS [1], [2]	No	Yes	None	Delay	Frequency control
Razor [7]	Yes	Yes	None	Delay	Error-det.&corr., Short-path elimination, Adaptation FSM
VOS [30], [31], [10]	No	No	Significant (trade-off vs. energy)	Delay	Possibly, resynthesis [33], [34]
BiVOS [9]	No	No		Noise	Multi-V_{DD} control
BiVOS+VOS [35]	No	No		Delay	
CRISTA [29]	No	Yes	None	Delay	Prediction, two-cycle control
AVOS	**Yes**	**No**	**Negligible**	**Delay**	**Lightweight error-detection, Adaptation FSM**

for resilient applications, avoiding some of the drawbacks of other methods.

VII. FUTURE WORK

The results reported in this paper are based on gate-level simulation. More accurate results could be obtained by more elaborate simulation or measurements on real hardware. Simulation accuracy could be improved by simulating at a lower level (e. g. considering individual line capacitances); this would be far slower though. Realistic memory access patterns (obtained by observing the behaviour of 3D graphics applications) could be used. Hardware implementations could be done in ASICs or programmable hardware such as CPLDs or various FPGA technologies.

Experiments applying the technique to other texture compression systems, such as DXT1, ftc2 and DXT5 and further resilient applications such as audio processing, pattern recognition or robotics could be done.

There could be potential for further reduction in energy consumption or error rate by improving the voltage control (exploring alternatives to the currently used time-decayed error sum, or e. g. the modification mentioned at the end of section II-B) or error detection (e. g. checksums).

AVOS could be incorporated into fault-tolerant general-purpose processors. For error-recovery with low overhead, such as [36] the overall solution may provide fault tolerance at little or no energy cost.

VIII. CONCLUSIONS

Adaptive voltage over-scaling (AVOS) allows energy savings of over 25% at typical operating conditions for resilient applications. AVOS has been applied to texture decompression, an application in which energy and bandwidth savings are paid by tolerating quality loss. Experiments show that AVOS achieves significant energy reduction for a large or a small number of intermediate voltages employed while the additional error is negligible compared to the quality loss due to lossy compression. The results are stable for different input data and under temperature and process variations.

ACKNOWLEDGMENT

Parts of this work have been performed at the University of Freiburg and supported by DFG grants GRK 1103/2 "Embedded Microsystems" and Be 1176/15-2 "RealTest". We are thankful to Prof. David Blaauw from the University of Michigan, Ann Arbor, for comments on the realistic modelling of voltage-control circuitry.

REFERENCES

[1] A. Chandrakasan and R. Brodersen, "Minimizing power consumption in digital CMOS circuits," *Proc. IEEE*, vol. 83, no. 4, pp. 498–523, 1995.
[2] P. Pillai and K. Shin, "Real-time dynamic voltage scaling for low-power embedded operating systems," in *ACM Symp. on Operating Systems Principles*, 2001, pp. 89–102.
[3] M. A. Breuer, "Multi-media applications and imprecise computation," in *Euromicro Conf.*, 2005, pp. 2–7.
[4] X. Li and D. Yeung, "Application-level correctness and its impact on fault tolerance," in *HPCA*, 2007, pp. 181–192.
[5] J. Liu *et al.*, "Imprecise computations," *Proc. IEEE*, vol. 82, no. 1, pp. 83–94, 1994.
[6] D. Nowroth, I. Polian, and B. Becker, "A study of cognitive resilience in a JPEG compressor," in *DSN*, 2008, pp. 32–41.
[7] D. Ernst *et al.*, "Razor: A low-power pipeline based on circuit-level timing speculation," in *Int'l Symp. on Microarchitecture*, 2003, p. 7.
[8] P. Krause, "ftc—floating precision texture compression," *Computers & Graphics*, vol. 34, no. 5, pp. 594–601, 2010.
[9] J. George, B. Marr, B. Akgul, and K. Palem, "Probabilistic arithmetic and energy efficient embedded signal processing," in *CASES*, 2006.
[10] G. Varatkar and N. Shanbhag, "Error-resilient motion estimation architecture," *IEEE Trans. VLSI*, vol. 16, no. 10, pp. 1399–1412, 2008.
[11] P. Brown and M. Agopian, "EXT_texture_compression_dxt1," http://opengl.org/registry/specs/EXT/texture_compression_dxt1.txt.
[12] D. Bull and S. Das, "Latch to block short path violation," 2008, US Patent 2008/0086624 A1.
[13] "OSU design flows for MOSIS SCMOS_SUBM design flow FreePDK 45nm variation-aware design flow," http://vcag.ecen.okstate.edu/projects/scells/.
[14] A. Davis, "An overview of algorithms in gnucap," in *University/Government/Industry Microelectronics Symp.*, 2003, pp. 360–361.
[15] J. Le, X. Li, and L. Pileggi, "STAC: Statistical timing analysis with correlation," in *DAC*, 2004, pp. 343–348.
[16] Y. Kawakami *et al.*, "A power grid optimization algorithm by observing timing error risk by IR drop," *IEICE Transactions*, vol. 91-A, no. 12, pp. 3423–3430, 2008.
[17] A. Agarwal, D. Blaauw, and V. Zolotov, "Statistical timing analysis for intra-die process variations with spatial correlations," in *ICCAD*, 2003, p. 900.
[18] D. Iparraguirre-Cardenas, J. Garcia-Gervacio, and V. Champac, "A design methodology for logic paths tolerant to local intra-die variations," in *ISCAS*, 2008, pp. 596–599.
[19] S.-T. Cheng and R. Brayton, "Compiling Verilog into Automata, UCB ERL Technical Report M94/37," 1994.
[20] R. Brayton *et al.*, "VIS: A system for verification and synthesis," in *Formal Methods in CAD*, 1996, pp. 428–432.
[21] Berkeley Logic Synthesis and Verification Group, "ABC: A system for sequential synthesis and verification, release 70930," http://www.eecs.berkeley.edu/ alanmi/abc/.
[22] S. Williams and M. Baxter, "Icarus Verilog: open-source Verilog more than a year later," *Linux Jour.*, vol. 2002, no. 99, p. 3, 2002.
[23] A. Loza *et al.*, "Structural similarity-based object tracking in video sequences," in *Int'l Conf. on Information Fusion*, 2006, pp. 1–6.
[24] Z. Wang *et al.*, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.
[25] P. Krause, "C++ implementation of DSSIM," http://colecovision.eu/graphics/DSSIM/.
[26] C. Gauss, "Theoria combinationis observationum erroribus minimis obnoxiae," *Commentationes Societatis Regiae Scientiarum Gottingensis recentiores*, vol. 5, pp. 33–90, 1823.
[27] G. Ohm, *Die galvanische Kette*, 1827.
[28] J. Pouwelse, K. Langendoen, and H. Sips, "Dynamic voltage scaling on a low-power microprocessor," in *Int'l Conf. Mobile Comp.&Netw.*, 2001.
[29] S. Ghosh, S. Bhunia, and K. Roy, "CRISTA: A new paradigm for low-power, variation-tolerant, and adaptive circuit synthesis using critical path isolation," *IEEE Trans. CAD*, vol. 26, no. 11, pp. 1947–1956, 2007.
[30] R. Hegde and N. Shanbhag, "Soft digital signal processing," *IEEE Trans. VLSI*, vol. 9, no. 6, pp. 813–823, 2001.
[31] ——, "A voltage overscaled low-power digital filter IC," *IEEE Jour. Solid-State Circuits*, vol. 39, no. 2, pp. 388–391, 2004.
[32] J. Natarajan, G. Kumar, S. Sen, M. Nisar, D. Lee, and A. Chatterhee, "Aggressively voltage overscaled adaptive RF systems using error control at the bit and symbol levels," in *IOLTS*, 2009, pp. 249–254.
[33] A. Kahng *et al.*, "Designing a processor from the ground up to allow voltage/reliability tradeoffs," in *HPCA*, 2010.
[34] N. Banerjee *et al.*, "Design methodology for low power and parametric robustness through output-quality modulation: Application to color-interpolation filtering," *IEEE Trans. CAD*, vol. 28, no. 8, pp. 1127–1137, 2009.
[35] L. Chakrapani *et al.*, "Highly energy and performance efficient embedded computing through approximately correct arithmetic," in *CASES*, 2008.
[36] P. Subramanyan *et al.*, "Power efficient redundant execution for chip multiprocessors," in *Works. Depend. & Secure Nanocomputing*, 2009.