Significance-Driven Computation on Next-Generation Unreliable Platforms

Georgios Karakonstantis[1], Nikolaos Bellas[2], Christos Antonopoulos[2], Georgios Tziantzioulis[2], Vaibhaf Gupta[3], Kaushik Roy[3]

Swiss Federal Institute of Technology (EPFL), Switzerland[1] University of Thessaly, Greece[2] Purdue University, U.S.A[3]

georgios.karakonstantis@epfl.ch, {nbellas, cda, getziadz}@uth.gr, {gupta64, kaushik}@purdue.edu

ABSTRACT

In this paper, we propose a design paradigm for energy efficient and variation-aware operation of next-generation multicore heterogeneous platforms. The main idea behind the proposed approach lies on the observation that not all operations are equally important in shaping the output quality of various applications and of the overall system. Based on such an observation, we enable all levels of the software design stack, including the programming model, compiler, operating system (OS) and run-time system to identify the critical tasks and ensure correct operation of such tasks by assigning them to dynamically adjusted reliable cores. Specifically, based on error rates and operating conditions identified by a sense-and-adapt (SeA) unit, OS selects and sets the right mode of operation of the overall system. The run-time system identifies the critical/less-critical tasks based on special directives and schedules them to the appropriate cores that are dynamically adjusted for highly-accurate/approximate operation by tuning their voltage/frequency. Cores that execute less significant operations can operate at voltages less than what is required for correct operation and consume less power, if required, since such tasks do not need to be always exact as opposed to the critical ones. Such scheme can lead to energy efficient and reliable operation, while reducing the design cost and overheads of conventional circuit/micro-architecture level techniques.

Categories, Subject Descriptors – C.4 [**Performance of Systems**]: Design Studies, Reliability

General Terms – Algorithms, Design, Reliability

Keywords – Energy Efficient, Software, Approximate Computing

1. INTRODUCTION

As transistors are getting smaller, spatial and temporal variations can have an unpredictable impact on gate delay and consequently lead to delay failures threatening the functionality and operational reliability of next generation systems [1]. Conventional wisdom for error-free operation dictates designing with margins (up-scaling voltage or clock) and adding extra hardware for detection and correction of any error [2]. However, such techniques come at high design and fabrication costs since they require major architecture changes and increased power, limiting the exploitation of energy and performance benefits of technology scaling. In addition, techniques such as voltage scaling [3] that can effectively reduce power consumption aggregate the random variations, and thus cannot be applied abruptly. Therefore, in order to ensure correct operation, while staying under the power and performance envelopes, new design paradigms are required that can efficiently tackle any error induced by variations or voltage scaling.

In this paper, we depart from conventional techniques and propose a software level approach for coping with variations and ensuring energy efficient operation. The main novelty of our approach is the synergy between all levels of software stack that are able to identify, schedule and execute the tasks that are necessary for obtaining acceptable quality under the given operating conditions. We classify tasks according to degrees of criticality, spanning from operation critical control tasks that need to execute reliably (such as OS functions), to non-critical tasks that do not contribute substantially to the output quality. Such classification allow us to reduce the requirement of exact computation of less-critical tasks and reduce power in case that best quality is not achievable or required by the user by mapping them to less-reliable cores that run at low voltage.

DAC'11, June 5-10, 2011, San Diego, California, USA

2. VARIATION AWARE SOFTWARE STACK

The proposed scheme is depicted in Fig. 1. Significant computations of an application are clustered in structures by using directives (#pragmas) at the programming level. By considering such pragmas, the compiler analyzes the data/control flow graph of the application and performs optimization that benefits the critical tasks. Run-time system is then responsible for mapping each task to different cores depending on their significance and the mode of operation set by the OS. Note that the mode of operation is set by OS dynamically based on a list of policies, the degree of spatial and temporal variations identified by the SeA unit and the quality/performance/energy requirements. In addition, OS monitors the progress of the application checking control points, identifying stale pointers and estimating error rates. In case that something goes wrong, OS intervenes and replays at least the critical tasks.

3. SOFTWARE-HARDWARE SYNERGY

Apart from software level support, robust operation is also assisted by tuning parameters at the hardware level. Specifically, based on the selected operational mode, and the criticality of scheduled tasks, SeA unit adjusts the voltage and frequency of

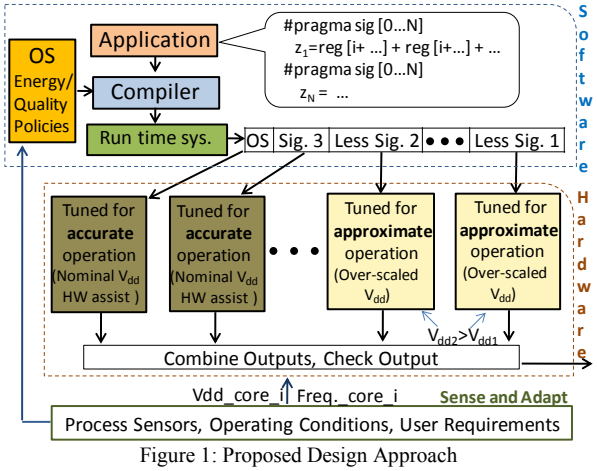

Figure 1: Proposed Design Approach

each core (if needed reliable circuit schemes are activated [2]) to meet the required reliability, energy, performance and quality requirements. Computationally significant tasks and OS operations are mapped to cores that operate under higher voltage and thus are reliable, as opposed to the less-significant tasks that are mapped to approximate cores running at lower voltage, thus consuming lower power. For instance, as shown in Fig. 1, the core that is scheduled to execute task of significance 1 is running at a lower $V_{dd1} < V_{dd2}$ than the core executing the task of higher significance equal to 2. Note that voltage values V_{dd1} and V_{dd2} are typically lower than the nominal voltage required for correct execution, thus leading to potential timing violations. However, such violations affect only the less-critical tasks. Therefore, our scheme allows the reduction of power, if needed, with minor quality reduction (approximate computation) since only the less critical tasks can get affected by potential errors induced by voltage scaling/variations. Note that some parts of the hardware that are responsible for executing critical OS and control tasks run always at a 'safe' mode, utilizing also some circuit techniques [2]. Overall, the proposed approach departs from existing works on approximate and variation-aware computing [2,3,4,5] that mainly target hardware level modifications rather than software level solutions. The proposed scheme provides a generic variation-aware solution, requiring limited software changes and dynamic tuning of the accuracy of each core based on the operational modes and critical tasks.

4. DESIGN EXAMPLE

Our approach lies on the fact that in almost all DSP systems some computations are less significant and can be approximated if best quality is not required or is not achievable due to hardware non-idealities. An example of such an application is the inverse discrete cosine transform (iDCT) that is ubiquitous part of video and imaging standards. iDCT transforms an 8x8 block of frequency components to spatial components (pixels). Based on the human visual system the high frequency components of each 8x8 block can be classified as less-significant since any error in their computation cannot be easily identified by the human eye. On the other hand, human eye is very sensitive to low-frequency components (usually contained at the upper 4x4 block of iDCT) and thus are classified as critical operations [5]. To be more specific let us write a part of the IDCT computation:

$$[w_0\ w_1\ w_2\ w_3]^T = C_1 z_0 + C_2 z_4 + C_2 [z_1 + z_2 + z_3]^T + C_4 [z_5 + z_6 + z_7]^T$$

where C_i contain the iDCT coefficients and z_i are input frequency coefficients [5]. This equation represents each output w_i in terms of significant and less significant computations. Specifically, C_1 computations represent low frequency (thus significant) components, whereas computations C_2 to C_4 represent gradually increasing frequency components (decreasing importance). According to the proposed approach C_1 represents significant tasks that need to be executed correctly under any condition. On the other hand, the exactness of C_2 to C_4 computations can be relaxed offering room for trade-offs between power and quality.

The significant tasks are clustered in segments and their significance is denoted by appropriate directives as shown in pseudo code of Fig. 2. Computations of higher significance are marked with a *sig* pragma with a larger number. The directives are identified by the compiler and the run-time system appoints each task to cores of different degree of accuracy. Specifically, C_1 is appointed at an accurate core running at nominal V_{dd}, whereas $C_{2\text{-}4}$, at cores with scaled V_{dd}s. Depending on the power, quality requirements computations $C_{2\text{-}4}$ can be executed at cores with higher degree of voltage scaling. For instance, under ultra low power requirements all less-significant tasks operate at low V_{dd}, risking their exact computation. In case that low power needs to be combined with good quality, then only C_4 is scheduled for

```
[w0, w1, w2, w3] = 0;
#pragma sig 3
  [w0, w1, w2, w3] += C1 z0 ;
#pragma sig 2
  [w0, w1, w2, w3] += C3 [z1 z2 z3 ] T;
#pragma sig 1
  [w0, w1, w2, w3] += C2 z4 ;
#pragma sig 0
  [w0, w1, w2, w3] += C4 [z5 z6 z7 ] T;
```

Figure 2: Significance-driven directives in iDCT code.

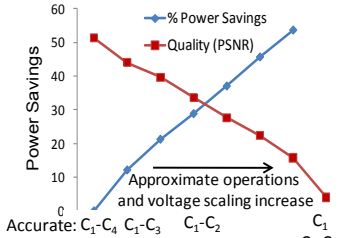

Figure 3: Power-Quality Trade-offs

(a) (b) (c)

Figure 4: Quality at (a) mode 1 (all operations correct), (b) mode 2 operations $C_{1\text{-}2}$ are computed accurately, (c) mode 3 only C_1 is correct

execution at a core with lower V_{dd}, thus only C_4 may not be computed correctly resulting in an approximate output.

5. PRELIMINARY EVALUATION

To evaluate the efficacy of the proposed software based approach and configure the OS policies we implemented a digital camera on a Altera FPGA DE-2 board utilizing the on board Nios processor. Ubiquitous parts of such system are the JPEG encoder/decoder which are based on DCT/iDCT blocks. Following our approach we clustered significant tasks in segments as described above. In case that low power is required then user can tune the on-board switches and the system adjusts to them by allowing errors to affect some of the less-significant tasks while ensuring accurate operation of tasks with higher significance. In this case voltage can be reduced leading to lower power at minor quality degradation. The power-quality trade-offs under the different operational modes along with some resultant images are shown in Figs. 3 and 4, respectively. Results show that over 50% power savings at a cost of 30% quality reduction are possible (in terms of peak- signal-to-noise-ratio (PSNR)) in case that only the most critical task C_1 is computed accurately.

6. CONCLUSION

A unique design paradigm for energy efficient and variation-aware operation based on synergy among all levels of software stack along with minimum hardware support is presented. The proposed scheme monitors the error rates under various environmental/operating conditions and based on the power/performance/quality requirements, it dynamically adjusts the robustness of each core depending on the significance of tasks that are scheduled to run. We believe that such scheme can emerge as a promising candidate for energy efficient and reliable operation on next generation platforms.

Acknowledgement: We would like to thank Anand Raghunathan and Andreas Burg for their comments and suggestions.

REFERENCES

[1] S. Borkar et. al., "Designing reliable systems from unreliable components: The challenges of transistor variability and degradation," *IEEE Micro*, 2005, pp. 10-16.

[2] Dan Ernst, et al., "Razor: Circuit-Level Correction of Timing Errors for Low- Power Operation," *IEEE Micro*, 2004, pp.10-20.

[3] B. Shim, et al., "Reliable Low-Power Digital Signal Processing via Reduced Precision Redundancy," *IEEE TVLSI,* 2004, pp. 497-510.

[4] L. Leem, et al., "ERSA: Error-Resilient System Architecture for Probabilistic Applications," *IEEE DATE,* 2010.

[5] G. Karakonstantis, et al., "System Level DSP Synthesis Using Voltage Overscaling, Unequal Error Protection & Adaptive Quality Tuning," IEEE SiPS, 2009.