Exploiting Hardware Transactional Memory for Error-Resilient and Energy-Efficient Execution

Dimitra Papagiannopoulou\textsuperscript{1}, Andrea Marongiu\textsuperscript{1}, Tali Moreshet\textsuperscript{3}, Maurice Herlihy\textsuperscript{2}, Luca Benini\textsuperscript{4}, R. I. Bahar\textsuperscript{1}

\textsuperscript{1}School of Engineering, Brown University, Providence, RI 02912
\textsuperscript{2}Dept. of Computer Science, Brown University, Providence, RI 02912
\textsuperscript{3}Dept. of Electrical and Computer Engineering, Boston University, Boston, MA 02215
\textsuperscript{4}Dept. of Information Technology and Electrical Engineering, ETH, Zürich, Switzerland CH-8092

Abstract—As semiconductor circuit sizes continue to shrink, execution errors are becoming an increasingly concerning issue. To avoid such errors, designers often turn to “guardband” restrictions on the operating frequency and voltage. If guardbands are too conservative, they limit performance and waste energy, but less conservative guardbands risk moving the system closer to its Critical Operating Point (COP), a frequency-voltage pair that, if surpassed, causes massive instruction fails.

In this paper, we propose a novel scheme that allows to dynamically adjust to an evolving COP and operate at highly reduced margins, while guaranteeing forward progress. More specifically, our scheme dynamically monitors the platform and adaptively adjusts to the COP among multiple cores, using lightweight checkpointing and roll-back mechanisms adapted from Hardware Transactional Memory (HTM) for error recovery. The overall effect is a system that offers safe execution guarantees at significantly reduced energy levels.

I. INTRODUCTION

Scaling of physical dimensions in semiconductor circuits has opened the way to many-cores [4], but at the price of ever-increasing static and dynamic hardware variability [1]. Spatial die-to-die and within-die static variations ultimately induce performance and power mismatches between the cores in a many-core array, introducing heterogeneity in a nominally homogeneous system (formally identical processing resources). Dynamic variations depend on the operating conditions of the chip, and include aging, supply voltage drops and temperature fluctuations.

The most common consequence of variations is path delay uncertainty. Circuit designers typically use conservative guardbands on the operating frequency or voltage to ensure safe system operation, with the obvious consequent loss of operational efficiency. When the guardbands are reduced, or when the system is optimistically operated far from the safe point, the delay uncertainty manifests itself either as an intermittent timing error [5] [3] or a critical operating point (COP) [10]. Timing errors may ultimately cause erroneous instructions with wrong outputs being stored or, worse, incorrect control flow. COP defines a voltage and frequency pair at which a core is error-free. If the voltage is decreased below (or the frequency is increased beyond) the COP, the core will face a massive number of errors [10]. The COP effect is highly pronounced in well-optimized designs [7] [8] due to so-called “path walls”.

Circuit level error detection and correction (EDAC) techniques [5] [3] can transparently detect and correct timing errors, with the side-effect of increased execution time and energy. While EDAC techniques are suitable in the presence of sporadic errors, they are obviously not a good solution for the “all-or-nothing” effect of the COP. In principle the COP can be determined for a particular chip after its production, and the most efficient yet safe voltage/frequency pair for the chip could be configured at that time. However, due to static and dynamic variations, the COP may actually change over space and time. As a result, the “safe” operating point may i) differ from one core to another (necessitating the entire chip to be conservatively tuned to meet the requirements of the slowest core) and ii) suddenly become unsafe due to aging, temperature fluctuations or voltage drops.

In our approach, we propose an integrated HW/SW scheme that addresses both types of COP variation phenomena by dynamically adjusting to an evolving COP, thus enabling the system to operate at highly reduced margins without sacrificing performance, while at the same time guaranteeing forward progress at significantly reduce energy levels. More specifically, our approach dynamically monitors the platform and adaptively adjusts to the COP among multiple cores, using lightweight checkpointing and roll-back mechanisms adapted from Hardware Transactional Memory (HTM) for error recovery.

We support two distinct types of recovery mechanisms: non-critical and critical. Non-critical recovery is required whenever a non-systematic error takes place in the datapath. In this case the consequence of an error is that incorrect data may be stored in memory. When this situation arises we employ a lazy-error detection mechanism: upon transaction commit we check if an error has occurred and if so we use lightweight rollback to restore memory and processor state and restart the transaction.

Critical recovery is required when an error takes place in the control part of the processor pipeline (e.g., instruction fetch/decode). This type of error breaks the original control flow of the program and prevents any software-based solution from taking control. When this situation arises we need to react promptly: An eager error detection mechanism is activated in HW, and the previous safe operating point is restored on the target core before transaction rollback and restart.
We assume the platform is initially configured to operate at a safe, nominal operating voltage. Every time a new transaction is started, our technique optimistically lowers the voltage in small steps (0.02 V), individually on each core. If sporadic errors take place, the described techniques intervene and ensure correct program behavior and progress. If systematic or critical errors take place, a new COP is found and the system reverts to the previous stable operating point. If, over time the COP changes, the technique is re-activated and the system is re-calibrated.

II. TARGET ARCHITECTURE & IMPLEMENTATION

Our HW/SW design is driven to a large extent by the target architecture. In particular, our work focuses on a programmable many-core accelerator (PMCA) that leverages a multi-cluster design to overcome scalability limitations [4] [6] [11]. In this multi-cluster configuration, simple processing elements (PE) are grouped into clusters sharing high-performance local interconnect and memory. Clusters are replicated and interconnected through a scalable medium such as a network-on-chip (NoC). Within a cluster, up to 8 32-bit in-order RISC processors feature private L1 instruction caches, and share a L1 tightly-coupled data memory (TCDM). The TCDM is configured as a shared multi-banked scratchpad memory that enables concurrent access to different memory locations. Simultaneous accesses to the same bank are serialized in a round-robin fashion. In the special case of simultaneous read requests to the very same location, a broadcast operation responds to all readers within the same cycle. Accesses to memory external to the cluster go through a peripheral interconnection. The basic synchronization mechanism is through standard read/write operations to a dedicated memory space which provides test-and-set semantics. However, we note that the TCDM itself does not employ a cache-coherence protocol.

The cores in the cluster are equipped with the HW monitoring mechanism for error detection described in [5]. On top of this baseline cluster we design our HTM extensions for error-tolerance. More specifically, we revisit existing checkpointing and rollback mechanisms that have been employed for HTM, to now be used as a lightweight mechanism for fast and efficient error recovery.

Our hardware transactional memory design incorporates a data versioning mechanism for keeping track of speculative and non-speculative versions of data in case it is necessary to rollback and recover from errors induced by operating below the COP. Similar to [9], our design uses the TCDM memory to hold both speculative and non-speculative data. In particular, data are distributed across the TCDM memory banks, such that each TCDM bank will be responsible for handling recovery only for data associated with its own bank. Data versioning is accomplished using distributed per-address logs, which has the advantage of being simple, fast, and space efficient. When a restore is needed, the process is triggered internally by each bank without interacting with other banks in the TCDM.

Our proposed algorithm works as follows. We start with all platform components set at the safe reference voltage level (1.0 V). Each time a core encounters a new transaction it saves its internal state and current stack and checks whether the self-calibration procedure was previously completed and the COP for this core is known. If the COP is still unknown, the executing core optimistically lowers its voltage level by a pre-defined step (0.02V), but if the COP has already been reached, then no voltage adjustment is made. If the transaction end is reached without errors being detected, the transaction commits and the logs of the committing core are cleaned. If errors are detected, then the transaction aborts. The logs and the internal state of the aborted core are restored and its voltage is adjusted back to the previous safe level. The core is then ready to retry the transaction. From this point on, the voltage level is no longer reduced when starting a new transaction.

Our proposed architecture has been modeled in Virtual SoC [2], a SystemC-based cycle-accurate virtual platform for the acceleration of massively parallel heterogeneous System-On-Chips, with back-annotated energy numbers for every single system component. We compare our transactional memory inspired technique to a conservative steady-voltage technique, which uses voltage margins (guardbands) to absorb the effects of static and dynamic variations. Our experiments indicate that our TM technique achieves significant energy savings compared to conservative execution at a steady reference voltage, for various temperature corners. To the best of our knowledge, this is the first full-fledged implementation of HTM for error resilient execution that targets reducing energy consumption. For the workshop, we will present performance and energy results over a range of benchmarks, considering both non-catastrophic and catastrophic recovery scenarios.

REFERENCES