**Parallel Computing of Graph-based Functions in Re-RAM**

Introduction

* CMOS approaches its physical boundaries due to the continuous shrinking feature size
* Search for promising emerging technologies beyond scaling limit has sped up
* Advancement of future low power, leakage free and nanoscale electronic systems
* Resistive Random Access Memory (ReRAM) is a non-volatile memory technology
* Attracted a high amount of attention among the emerging post-CMOS device technologies
* ReRam Features
  + Low power consumption
  + Inherent computing capabilities
  + Efficiency of logic synthesis
* Maximization of parallel computation
* Limiting number of memory accesses allowed
* Binary Decision Diagram (BDD)
* And-Inverter Graph (AIG)
* M-And-Inverter Graph (M-AIG)

BDD

* BDD-based synthesis approach for ReRAM crossbars
* Utilizes Multiply-Accumulate (MAC) operation instead of logic primitives
  + Implication
  + Majority function
* Proposed approach first optimizes BDDs using an evolutionary algorithm
* Directly maps the BDD nodes represented by multiplexers onto parallel MAC operations realized in crossbar columns.

AIG

* Automated compiling procedure
* An in-memory computer architecture based on regular ReRAM crossbar arrays
* Translates arbitrary Boolean functions
* Executable by applying appropriate voltage levels
* Execution of bit-level parallel computational instructions

Graph-based Representation

* Graph-based representations of Boolean functions
  + BDDs
  + AIGs
* Plays an important role in synthesis.
* AIGs are representations of Boolean networks.
* The edges represent wires between two-input AND gates that correspond to the nodes
* The edges can be complemented to represent inverters between the nodes

ReRAM MAC Computation

* Applications such as neural networks and neuromorphic computing
* Perform a lot of matrix multiplications
* An efficient implementation of the MAC operation is needed
* ReRAM provides the basis for such an efficient implementation,
* Compute multiple MAC operations in parallel
* Consider the ReRAM crossbar in matrix, computes I = Ax

Graph Based Computation in ReRAM

* Existing approaches have tried to find efficient ways of mapping
* MIGs need the smallest number of operations and devices
* They have become the state-of-the art graph structure for ReRAM-based synthesis.
* Existing approaches do neither focus on parallel computations, nor do they use the MAC operation

Types of Parallelism

* Parallel computations perform on a ReRAM array
* Addressing multiple wordlines or bitlines simultaneously
* A computation is wordline parallel if it uses one wordline and multiple bitlines;
* A computation is bitline parallel if it uses one bitline and multiple wordlines;
* A computation is mixed parallel if it uses multiple wordlines and multiple bitlines.
* Row and column drivers are represented by triangles
* Active devices are shown in red and slightly enlarged
* The ReRAM devices in which a computation takes place are highlighted in red
* Mixed parallel computation has to deal with data distortion
* Even the small computation cannot be performed if only one of the six activated devices contains a value that must not be overwritten
* Problem becomes more severe as more lines are activated
* It is unfeasible to make efficient use of this parallelism in general when performing RM3-based computations
* Bitline parallel computation cannot deal with inversions efficiently, which are central to most logic representations
* Bitline is naturally inverted but the wordline and the internal resistance state are not

BDD-based Parallel Computation

* To compute a BDD on a ReRAM crossbar, every node has to be realized as a 2x1 multiplexer designating Boolean relation
* The nodes at ith The AIG levels computes successively
* Place and compute the inverted values necessary for this level
* Place the nodes of the level and initialize the devices with the corresponding host operands.
* Compute the nodes themselves by scheduling the computations that have host devices in the same word in parallel.
* BDD level use an identical input variable of the target function as select line
* Representing a function with N input variables using an initial ascending variable ordering

AIG-based Parallel Computation

* Computing two MIG-nodes in parallel within a wordline we have to respect four constraints
  + All children of both nodes must be computed. In particular, there must not be any data dependencies between the nodes.
  + The nodes must share a wordline operand.
  + The host devices of the nodes must be placed in the same wordline.
  + The content of the host devices must not be necessary for any other computations.
* Keep the number of required ReRAM devices as low as efficient parallelism allows
* Keep track of whether the content of a device is still needed
* Call the set of free devices in a word a *hole*
* The devices in a hole can be reused instead of allocating new ones
* Filling a small hole in a word is computationally inefficient
* Primary Inputs (PIs) are present in the memory array

M-AIG-based Parallel Computation

* Each node represents an m-Input And Gate or a PI
* Each input edge can either be connected to the constant 1 or to a child node
* If the input edge is connected to a node, then the edge may be complemented to indicate inversion
* AIGs are a subset of m-AIGs, m-AIGs are not canonical
* Add m-2 inputs to each node that is not a PI of the AIG
* Each additional input is connected to the constant 1.
* Add the inputs to each node
* Simplify using a bottom-up approach in the second step by merging nodes

Conclusion

* Proposed approach significantly reduces the number of devices
* Reduces operations needed
* It needs almost the same number of devices but reduces the number of operations by about 66% on average
* BDD and AIG outperform in both area and operation
* Efficiency of m-AIG has shown
* For small values of m, m-AIG has outperform AIG

**Power Aware Computing**

ABSTRACT

* Power consumption is important in in design of data center.
* Power efficiency is the major constraint in system designs.
* Power and energy are the challenges poses by power efficiency.
* In this analysis scientific benchmarks and set of representatives kernels are used.
* Result and conclusion will help in achieving energy efficiency in computing algorithm.

INTRODUCTION

* Processor designs are getting effected due to Power, energy and temperature.
* Stagnation of CPU clock frequencies and reliance on parallelism increase energy efficiencies of future.
* Along with advancements in hardware, software advancements for energy efficiency is important too.
* Software design can bring huge improvements in addition to the hardware.
* Ability of measuring the power and energy consumption is mandatory step.
* PAPI library was used in this analysis that give generic and portable interface to hardware counters attached to CPU and other components.
* Experiments were executed on Intel Xeon Phi Knights Landing (KNL) architecture.
* Analysis was performed using Dense Linear Algebra (DLA) Kernels.
* Specifically BLAS kernels.

HARDWARE

* Intel Xeon Phi Knights Landing processors offers 16 GB of High bandwidth Memory based on 425GB/s of bandwidth of MCDRAM, along with DDR4 memory of bandwidth 90 GB/s.
* The MCDRAM can be configured in BIOS in one of three different usage modes. One is flat, second is cache and other is hybrid mode.
* Experiment was conducted on three different configuration of MCDRAM on the kernel.

REPRESENTATIVE KERNELS

* Kernels that can be found in high performance computing application were chose to study and analyze the effect of application of power consumption and energy requirements.
* For low level linear algebra operations BLAS is specific for it.
* BLAS operations are categorized in three levels  
  **Level 1:** It addresses scalar and vector operations.  
  **Level 2:** It addresses matrix-vector operation  
  **Level 3:** It addresses matrix-matrix operations.
* BLAS routine is ideal for examining the power and performance characteristics.
* Level 1 and Level 2 belongs to **Memory Bond Class**.
* Whereas Level 3 belongs to **Compute Intensive Class.**
* Presented the study and to analyze the compute intensive routine dgemm and memory bound class dgemv.
* As it provides wide range of computational intensities.

PAPI: The PERFORMANCE API

* PAPI performance library provides a coherent methodology to performance counter information varied for different hardware and software component.
* Power Monitoring Capabilities
* PAPI offers many components which enables to monitor power usage and consumption of energy through different interfaces like Intel RAPL (Running Average Power Limit) and MicAccess API.

Study of dgemm Kernel Behaviour

Graphical user interface, diagram, application, table, Excel

Description automatically generated

* Figure **a** shows when KNL is booted in HYBRID and the data is allocated in DDR4.
* Figure **b** shows when data is allocated in MCDRAM.
* Figure **c** and **d** shows the measurements for dgemm when the KNL is booted in FLAT mode.
* The FLAT mode does not use the MCDRAM as a cache, but as physical addressable memory space

Study of dgemv Kernel Behaviour

Graphical user interface, Excel

Description automatically generated

* Figure **a** and **b** illustrates the results for the Hybrid Mode because kernel is memory bound and cache reuse benefits are limited.
* Performance drops dramatically between the two storage.
* Figure **c** and **d** illustrates the results for Flat Mode.
* Results are observed same as of Hybrid Mode except for DDR4. For which performance is 4 times slower as compared to MCDRAM.

CONCLUSION

* This study allowed us explore power management and capping strategies matching the computational intensities which leads to energy saving.
* Using high bandwidth MCDRAM on KNL is important for achieving high performances and minimal consumption of power.
* Behavior of these kernels were explored while booting configuration of MCDRAM on KNL.
* If the application is compute intensive than Hybrid mode is best, in either case memory bound option is best and allocate the data in MCDRAM.

**Temperature-Aware Computer Systems Opportunities and Challenges**

Abstract

* Effects of localized heating and non-uniform power dissipation
* Architecture-level thermal modeling
* Dynamic voltage scaling
* Migrating computation
* Temperature-tracking dynamic frequency scaling.

Introduction

* Power-aware design alone has failed to stem the tide of dealing with problems like heat density
* Localized heating occurs much faster than chip-wide heating
* Power-management techniques must directly target the spatial and temporal behavior of the operating temperature.
* Typical high-power applications still operate at 20 percent or more below the absolute worst case

Need for Architecture-Level Thermal Management

* Architecture domain of the computing system is unique
* Development of workload to control instruction level parallelism
* Design manual provides hot spots and temperature gradients for computing system
* Role of system-architecture and operating-system is important
* This research paper has focused more on microarchitecture to handle thermal problems in computing.

Need for Architecture-Level Thermal Modeling

The need for architecture-level thermal modeling is there to avoid temporal and spatial nonuniformities in computing due to thermal effects. An effective architecture-level thermal model must be

* Simple enough to allow architects to reason about thermal effects and tradeoffs
* Detailed enough to model runtime changes in temperature within different functional units
* Yet computationally efficient and portable for use in a variety of architecture simulators.

Thermal Modeling at Architecture Level

In the research paper, properties about the compact model of a parametric microarchitecture for a computing system are presented.

It must track temperatures at the granularity of individual microarchitectural units

It must be parameterized in so that a new compact model for different microarchitectures.

It must be able to solve the RC circuit’s differential equations quickly

Finally, it must be boundary and initial-condition independent

IC package with heat sink: physical structure(Left) Simple compact thermal model(Right)

Diagram

Description automatically generated

Temperature-tracking Dynamic Frequency Scaling

Several microarchitecture techniques target runtime temperature regulation. In this paper, a sensor is modeled with the techniques mentioned previously. One sensor per architectural block with random noise of 10C. Temperature versus average power density for gcc with a power averaging interval of 0.033 seconds is presented in the paper.

Temperature-tracking Dynamic Frequency Scaling

Independently of the relationship between frequency and voltage, the temperature dependence of carrier mobility in CMOS means that frequency is also linearly dependent on the operating temperature. This suggests that the standard practice of designing the nominal operating frequency for the maximum-allowed operating temperature is too conservative. When applications exceed the temperature specification, they can simply scale frequency down in response to the rising temperature. Because this temperature dependence is mild within the operating region of interest, the performance penalty for doing so is almost negligible.

Dynamic Voltage Scaling

* Designers have long regarded DVS as a solution for reducing energy consumption
* Researchers have recently proposed it as one solution for thermal management
* Transmeta’s Crusoe processors use it for this purpose
* When changing the processor voltage, a processor must reduce frequency in conjunction with voltage, because circuits switch more slowly as the operating voltage approaches the threshold voltage.

Migrating Computation

From the results, MC is the best DTM technique at 0.8 K/W

* It works well for three reasons.
  + The floorplan used by itself is enough to reduce the operating temperature of the primary integer register file.
  + MC can exploit ILP to hide the extra latency of the spare register file
  + The complete elimination of activity in the primary register file allows it to cool quickly, minimizing the use of the slower secondary register file.

Conclusion

* In terms of modeling, the most important area for future work is the inclusion of heating from the clock grid and other interconnect.
* This paper currently approximates the effects of wires by including their power dissipation in the dynamic, per block power density values that drive the RC model.
* Different techniques such as Temperature-tracking Dynamic Frequency Scaling, Dynamic Voltage Scaling, and Migrating Computation are presented in detail to overcome the power and thermal issues in computing