MEMORY HIERARCHY PARTITIONING AND DATA MAPPING TECHNIQUES FOR SCRATCH PAD BASED ARCHITECTURES

PRASENJIT CHAKRABORTY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY DELHI
JUNE 2015

MEMORY HIERARCHY PARTITIONING AND DATA MAPPING TECHNIQUES FOR SCRATCH PAD BASED ARCHITECTURES

by

PRASENJIT CHAKRABORTY

Department of Computer Science and Engineering

Submitted
in fulfillment of the requirements of the degree of
Doctor of Philosophy

to the

Indian Institute of Technology Delhi
June 2015

Certificate

This is to certify that the thesis titled **Memory Hierarchy Partitioning and Data Mapping Techniques for Scratch Pad Based Architectures** being submitted by **Prasenjit Chakraborty** for the award of **Doctor of Philosophy** in Computer Science & Engg. is a record of bona-fide work carried out by him under my guidance and supervision at the Department of Computer Science & Engineering, Indian Institute of Technology Delhi. The work presented in this thesis has not been submitted elsewhere, either in part or full, for the award of any other degree or diploma.

Preeti Ranjan Panda
Professor
Department of Computer Science and Engineering
Indian Institute of Technology Delhi
New Delhi - 110016

Dedication

To my uncle, Late Dr. Monotosh Chakraborty.

Thank you for your love, support and encouraging me for heights!

Acknowledgments

There are several people I would like to thank, without them this thesis would not have been possible. First and foremost, I would like to thank my advisor, Prof. Preeti Ranjan Panda. My first interaction with him was during my application process for part-time PhD and he was instrumental in enabling me to get admitted to IIT Delhi and for giving me an opportunity to work under his supervision. His constant guidance, advice and technical mentorship were valuable to make this thesis a reality and I have learned a great deal from him. I thank Prof. Sandeep Sen for providing the theoretical formulation and proof of the work presented in this thesis. I would like to thank many other professors at IIT Delhi that I have interacted over the last 7 years.

I would like to express my deepest gratitude to IBM India for permitting me to purse part-time PhD studies. My sincere thanks to Sriram Vajapeyam for instigating the thought to pursue PhD and embarking me into the world of research. Special thanks to Manish Gupta and Reena Malangone for instrumenting this into a reality within the realm of office work. I also thank Intel India for all the support.

Thanks to all my PhD research colleagues - Ayesha, Vaibhav, Gayathri, Namita, Krishnaiah, Sandeep, Sharat for their support during various stages over the course of my PhD studies. Thanks to Sahu, Anant, Lava and Kameshwar for all the discussions and company over the coffee breaks.

I thank my wife, Ananya, for the understanding over the past few ardous years while I was too busy managing PhD research and office work. I thank my daughter Pratyusha and son Adhvik for their love; they make life worth looking forward to. Finally, I thank my mother for her affection and unconditional love.

Prasenjit Chakraborty

Abstract

Modern computer systems lay greater emphasis on energy-efficiency as performance ceases to be the sole criteria for processor design. Therefore, architecting systems that improve performance yet remain within the power budget becomes necessary. The memory subsystem plays a dominant role in this endeavor of system redesign. Due to the growing gap between the CPU and DRAM performance/energy, innovating on-chip memory storage becomes paramount. Scratch-Pad memory (SPM) is considered a useful component in the memory hierarchy, solely or along with caches, to meet the power and energy constraints. Although the efficiency of SPM is well-known, its usage has been restricted owing to difficulties in programmability. Real applications usually have regions that are amenable to exploitation by either SPM or Cache, and hence can improve if the two are used in conjunction. Dynamically adjusting the on-chip local memory resources to suit application demand, can significantly improve the efficiency of the overall system. In this work, we propose a compiler technique to map application data objects to SPM/cache and also partition the local memory between SPM and cache depending on the dynamic requirement of the application.

First, we design a framework to characterize application data objects and partition the on-chip memory space between the objects. This helps alleviate the manual effort required of the developers in programming SPM based systems. This profile-based solution is obtained by a quick one-time execution of an instrumented binary instead of slow and detailed simulations of all possible local memory configurations. We apply detailed compiler based analysis for further refining the allocation and partition sizes. Second, we develop a novel cost based cache aware algorithm to decide the mapping of the application

objects between SPM and cache at different regions of the program. Our data mapping strategy is able to seamlessly integrate existing SPM mapping techniques, and is structured as an enhancement to assign the cache to data objects where it is considered to be more appropriate than SPM. We demonstrate its utility by executing applications on Cell BE hardware which resulted in performance improvement over previous SWC-based strategies.

Finally, in this research work we introduce a novel graph based structure to tackle data mapping decisions in a complex application. We use this to present a data mapping heuristic to map program objects for a fixed sized SPM-Cache hybrid system that targets whole program optimization. We extend this formulation to adapt the SPM and cache sizes, as well as the data mapping as per the requirement of different application regions. We study the applicability of the technique on various workloads targeted at both SPM-Only and hardware reconfigurable memory system, and obtain the performance and energy results using simulations and compare it against existing techniques. We demonstrate that our compiler based allocation scheme is applicable for complex programs targeted at all flavors of SPM based architectures and results in significant performance efficiency improvement over existing competitive techniques.

Contents

List of Figures

List of Tables