Automatic Software Hardware Co-Design for Reconfigurable Computing Systems

Proshanta Saha
NSF Center for High-Performance Reconfigurable Computing (CHREC)
ECE Department, The George Washington University
sahap@gwu.edu

1. INTRODUCTION

A formal methodology for automatic hardware-software partitioning and co-scheduling between the µP and the field programmable gate array (FPGA) has not yet been established. Current work in automatic task partitioning and scheduling for the reconfigurable systems strictly addresses the FPGA hardware, and does not take advantage of the synergy between the microprocessor and the FPGA [1][2]. In this research, we consider the problem of formalizing a co-scheduling methodology and develop a set of intuitive tools to assist users in realizing the full potential of an RC architecture.

Scheduling is critical for efficient resource utilization and achieving speedup in high performance reconfigurable computers (HPRC). The primary targets of this research are reconfigurable computing (RC) systems that have both microprocessors and FPGAs.

2. SCHEDULING REQUIREMENTS

Scheduling algorithms for RC systems have to consider the resource constraints on the reconfigurable hardware such as the number of FFs, LUTs, MULTs, CLBs, the reconfiguration overhead, communication overhead, routing overhead, size constraints, throughput, and power constraints before selecting a task to map on to the FPGA. In addition, the scheduling algorithms have to consider multiple implementations of a task depending on the objective function. Unlike reconfigurable hardware only scheduling algorithms, RC systems need to schedule tasks between the µP and the FPGA to take full advantage of the architecture.

3. RECOS ALGORITHM

In our work we investigated scheduling algorithms from related fields such as Embedded Computing (EC), Heterogeneous Computing (HC), and Reconfigurable Hardware (RH) to adapt and leverage existing scheduling algorithms and techniques. The result from our investigation showed that simply adapting existing scheduling algorithms would not be sufficient [8][10]. The Reconfigurable-computing Co-Scheduling (ReCoS) algorithm was proposed to address the concerns [9][11].

Figure 1 shows the various objective functions serve as input to the ReCoS algorithm including communication overhead, resource constraints, reconfiguration overhead, routing overhead, and application task graph. The objective functions are used to find a suitable schedule for the tasks. The output is then further optimized to ensure that all resources are utilized to its full potential. The result is a task graph assigned to its corresponding processing element (PE).

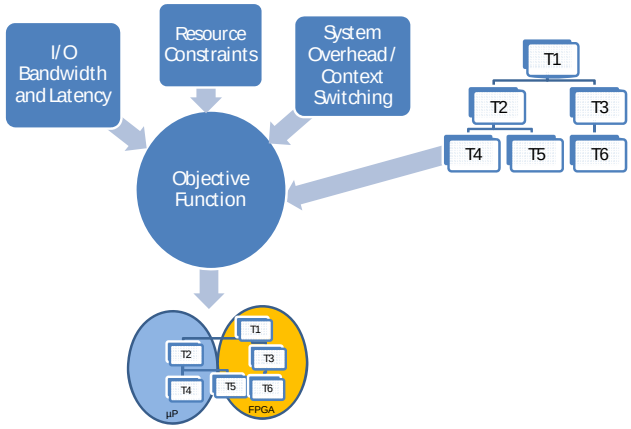

Figure 1: Objective Functions of the ReCoS Algorithm

The proposed algorithm combines the strengths of the HC and RH scheduling algorithms namely in speed and scheduling capabilities and pays special attention to load balancing issues and lack of co-design constraint satisfaction.

4. RESULTS

The ReCoS algorithm is compared against three EC algorithms, Hou [4], Yen [5], and Oh [6] algorithms on an SRC6 platform. These scheduling algorithms were chosen as they are representative of the typical algorithms available for embedded design, ie. critical path reduction, exhaustive search, and penalty reduction. All subtasks targeting the µP are written in C (ISO-C99), while all subtasks targeting the FPGA are written in Verilog (Verilog 2001) along with the necessary wrappers to interface with the SRC6 system written in Carte's MAP-C language [7]. The scheduling algorithms assume a 2 PE model (µP and FPGA). The test bed includes four applications JPEG IDCT, MPEG IDCT, DES encryption, and IDEA encryption.

In [11] the benefit of using the ReCoS algorithm is shown. Despite then enhanced Yen, Hou, and Oh algorithms' [8] effort to reduce the number of PEs, in this case resulting in the best case scenario of a single PE and thus similar execution times as shown in figure 2, they exploit neither the inherent parallelism of FPGAs nor the

I/O capabilities of the µP. The ReCoS algorithm is able to exploit the space remaining on a FPGA to add more implementations of the tasks onto the chip without violating the area constraint $\sum A_R \leq .75$.

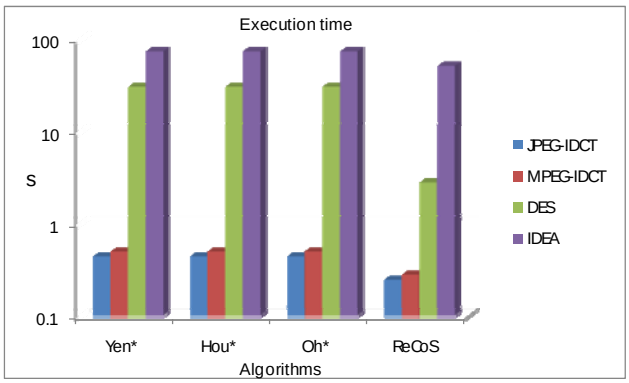

Figure 2: ReCoS algorithm execution time compared against enhanced reconfigurable aware Yen, Hou, and Oh algorithms on 1M blocks of data.

5. AUTOMATING CO-DESIGN

Automation of the co-design process is underway, along with work to present users with an intuitive user interface (UI) that will assist users in examining their application and the possible areas in which hardware acceleration can be beneficial. The UI consists of an integrated development environment (IDE) that provides a visualization of the user's application and the various results from the tools which includes an intermediate compiler that will tag user source code for parallelism analysis; A software profiler that can identify the hot zones, loops, and bottlenecks in the application; A hardware profiler that provides an estimate of the resource requirements; The ReCoS co-scheduling algorithm that provides a suitable task graph; and finally a dynamic load balancing analysis tool to provide feedback in regards to the actual load and reconsider PE assignments if necessary.

6. CONCLUSIONS

To the best of our knowledge there are no automatic hardware/software co-schedulers for reconfigurable computing systems that take advantage of the synergy between the µP and the FGPA available today. Related work either focuses solely on the reconfigurable hardware or exists in related fields such as HC and EC. Manual partitioning efforts are currently the only means of co-scheduling on an RC system, and are often tedious for large applications. In our work we investigated scheduling algorithms that can aid developers in identifying an optimal co-schedule that takes advantage of the synergy between the µP and the FPGA quickly and efficiently [8][9][10][11]. This dissertation [12] aims to provide compiler writers, tool developers, and application scientists a methodology for efficient and automated partitioning between software and hardware.

7. ACKNOWLEDGEMENTS

This work was supported in part by the I/UCRC Program of the National Science Foundation under the NSF Center for High-Performance Reconfigurable Computing (CHREC).

8. REFERENCES

[1] Danne, K.; Platzner, M.; A heuristic approach to schedule periodic real-time tasks on reconfigurable hardware; International Conference on Field Programmable Logic and Applications, 2005; 24-26 Aug. 2005 Page(s):568 – 573

[2] Katherine Compton, Zhiyuan Li, James Cooley Stephen Knol, Scott Hauck; Configuration relocation and defragmentation for run-time reconfigurable computing; IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 10, No. 3, June 2002

[3] Howard Jay Siegel, Shoukat Ali; Techniques for mapping tasks to machines in heterogeneous computing systems; Journal of Systems Architecture Volume 46 Page(s): 627-639.

[4] Junwei Hou; Wolf, W.; Process partitioning for distributed embedded systems; Fourth International Workshop on Hardware/Software Co-Design, 1996. (Codes/CASHE '96), Proceedings., 18-20 March 1996 Page(s):70 - 76

[5] Yen, T.-Y.; Wolf, W., Sensitivity-Driven Co-Synthesis of Distributed Embedded Systems, Proceedings of the Eighth International Symposium on System Synthesis, 1995., 13-15 Sept. 1995 Page(s):4 – 9

[6] Hyunok Oh, Soonhoi Ha; A Hardware-Software Cosynthesis Technique Based on Heterogeneous Multiprocessor Scheduling; Proceedings of the Seventh International Workshop on Hardware/Software Codesign, 1999. (CODES '99), 3-5 May 1999 Page(s):183 – 187

[7] Carte-C Programming Environment, http://www.srccomputers.com/CarteProgEnv.htm

[8] Saha, P., El-Ghazawi, T.; "Extending Embedded Computing Scheduling Algorithms for Reconfigurable Computing Systems"; 3rd Annual IEEE Southern Conference on Programmable Logic 2007; February 26-28, 2007

[9] Saha, P., El-Ghazawi, T.; "Software/Hardware Co-Scheduling for Reconfigurable Computing Systems"; International Symposium on Field-Programmable Custom Computing Machines(FCCM2007); 23-25 April 2007

[10] Saha, P., El-Ghazawi, T.; "Applications of Heterogeneous Computing in Hardware/Software Co-Scheduling "; ACS/IEEE International Conference on Computer Systems and Applications 2007; May 13-16, 2007

[11] Saha, P., El-Ghazawi, T.; "A Methodology for Automating Co-Scheduling for Reconfigurable Computing Systems"; Fifth ACM/IEEE International Conference on Formal Methods and Models for Codesign 2007, May 30-June 1 2007

[12] Saha, P.; "Application Hardware Software Co-Design for Reconfigurable Computing Systems"; The George Washington University Dissertation Proposal; November 8 2006 (Electrical and Computer Engineering Department Internal Document)