

Malleable Architectures for Adaptive Computing

MIT 9904-04

Progress Report: January 1, 2001—July 31, 2001

MIT: Arvind, Larry Rudolph and Srinivas Devadas
NTT: Hiroshi Sawada

Project Overview

Field Programmable Gate Arrays (FPGAs) are being used as building blocks in many different adaptive computing systems. We propose a framework for the synthesis of reconfigurable processors based on existing million-gate FPGAs such as the Xilinx XC4000XV series. The instruction sets of the processors are synthesized prior to running each application so as to significantly improve performance or the power dissipated during the execution of the application. Synthesis of instruction sets is made possible by the development of an architecture exploration system for programmable processors. Further, processor caches can be reconfigured in a dynamic manner so as to improve hit rates for multimedia streaming data. This reconfiguration is made possible by implementing several hardware mechanisms such as column and curious caching into the processor cache.

Neither general-purpose microprocessors nor digital signal processors meet all the needs of intelligent personal devices, multimedia players and recorders, and advanced communication applications. Designing special purpose chips for each application is too expensive to be a feasible solution. It is possible that a large reconfigurable device with appropriate tools and infrastructure may be the solution.

We are investigating a revolutionary technology for designing hardware and firmware from high-level specifications. The approach is to synthesize "malleable" processors, with application specific instruction sets, into million-gate FPGAs. The instruction sets of the processors are tailored to each application so as to significantly improve either the performance or the power dissipated during the execution of the application. Synthesis of instruction sets is made possible by the development of an architecture exploration system for programmable processors. This technology can dramatically reduce the time to market in sectors where the standards are changing too quickly or where functionality evolution is too rapid for traditional hardware design.

For the past six months, we have focused on the malleable cache aspect of a malleable processor. We developed a methodology to improve the performance of embedded processors running data-intensive applications by managing on-chip memory on an application-specific or task-specific basis. We provide this management ability with several novel hardware mechanism, *column*, *curious*, *TLB*, *caching*.

Column caching provides software with the ability to dynamically partition the cache. Data can be placed within a specified set of cache "columns" to avoid conflicts with other cached items. By mapping a column-sized region of memory to its own column, column caching can also provide the same functionality as a dedicated scratchpad memory including predictability for time-critical parts of a real-time application. Column caching enables the ability to dynamically change the ratio between scratchpad size and cache size for each application, or each task within an application. Thus, software has much finer software control of on-chip memory.

Progress Through June 2001

For the past six months, we have focused on reducing latency through predictive hierarchy prefetching. In a time-shared system, jobs or processes share the memory system. We have found that it is often possible to predict the next process to execute. So, a prefetch engine begins to bring in the data required by this next process even before it begins to execute thereby minimizing the "cold" misses. Our technique is applicable to all levels of the memory hierarchy and to all types of tasks – jobs, threads, and code blocks. The key is to insure that prefetching data to be used by the next process does not evict data needed by the current process. Column caching and selective evicting of data makes this possible. Preliminary investigation has shown significant performance improvements for applications such as event-driven simulation systems. We believe that many types of stream-based applications, such as speech and vision processing, will benefit due to the ability to look ahead in the input stream and prefetch the necessary data before it is needed.

A student from our group, Josh Jacobs, spent the summer at NTT investigating how to use subband processing to improve BSS performance. Experiments were performed using subband techniques to improve the performance of an independent component analysis (ICA) based blind source separation (BSS) system. The system separates the input signal into different frequency bands, feeds each frequency band into a separate BSS system with different parameters, and recombines the outputs to create the final un-mixed signals. The system allows each frequency band to be processed using different FFT block sizes and microphone configurations. By varying these parameters intelligently for each frequency band, it is possible to significantly improve the system's performance.

Research Plan for the Next Six Months

We plan to apply speculative prefetching of task, thread, and process data to a variety of stream-based applications. In particular, we will look at speech, vision, and ICA analysis applications. A complete simulation environment will be completed over the next three months and then used to perform the experiments. Modifications to the existing microprocessors will also be defined and evaluated.