Malleable Architectures for Adaptive Computing
NTTMIT 9904-04

Progress Report: July 1, 1999—December 31, 1999

MIT: Arvind, Larry Rudolph, 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. Column caching provides control as to where items are stored in a cache and curious caching allows the cache to fetch data that has been placed on the bus.

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.

Progress Through December 1999

Column Cache – Embedded DRAM

We developed a methodology to improve the performance of embedded processors running data-intensive applications by allowing embedded software to manage on-chip memory on an application-specific or task-specific basis. We provide this management ability with a novel hardware mechanism, *column 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.

TRS – A High-level Description and Synthesis Framework

Synthesizable forms of hardware descriptions have traditionally followed a state-centric paradigm that specifies for each state element in the system, its new state after every clock cycle. A designer must explicitly manage the concurrency in a design by scheduling the exact cycle-by-cycle interaction between the different parts of the system. Such a description, whether schematic or textual (RTL Verilog), gives detailed instructions on how to implement a digital circuit, but the same description cannot be easily correlated to the intended functionality of the circuit. Due to this wide abstraction gap between implementation and functional representations, considerable effort and digital design expertise are required to produce a state-centric description from a functional starting point**.**

Hardware design can be simplified by allowing the direct synthesis of operation-centric functional descriptions that specifies the behavior of a system on an operation-by-operation basis. Each operation is described by a triggering precondition and the operation's effect on the system's state as a whole. Semantically, these operations are interleaved atomically and sequentially during an execution. For example, a microprocessor programmer's manual is an operation-centric description that describes the behavior of a processor on an instruction-by-instruction basis. We developed an architectural description language based on the formalism of Term Rewriting Systems. This language captures complex hardware behavior concisely and precisely. A compiler has been developed that can generate a synthesizable Verilog RTL description from an operation-centric TRS description In a design example based on a 32-bit MIPS integer core, the TRS description is less than one-tenth the size of a hand-coded RTL description.

Research Plan for the Next Six Months

During the next six months, we expect to apply our ideas on column and curious caching in particular, and malleable processors in general to a variety of application domains. During the recent MIT trip to NTT, Arvind will meet with Mr. Hiroshi Sawada and members of the adaptive architecture group including Dr. K. Shimohara. This group is working on similar FPGA synthesis problems, and we expect to use our processor and architectures designed using the architecture exploration system as benchmark applications for the FPGA CAD software being developed at NTT. In addition, we expect to investigate the ability to implement our embedded DRAM and column caching ideas for NTT applications.

Curious cache technology improves the performance of stream-based applications. We expect to develop a practical implementation of this idea and to evaluate its use for several platforms, especially for NTT applications.