Abstract
Runtime Environment for Dynamically Reconfigurable Embedded Systems

Sandeep Neema, Ted Bapty, Jason Scott
Institute for Software Integrated Systems, Vanderbilt University

Emerging applications for embedded systems impose strict demands on system implementation technology. High performance requires application-specific architectures, but flexibility and system agility require a programmable approach. Reconfigurable hardware offers a potential solution, allowing efficient, hard-wired designs to be implemented in FPGA's, while permitting dynamic reprogramming to achieve system agility with multiple operational modes.

While FPGA's have proven to allow efficient processing of some algorithms, others tasks, such as floating point arithmetic, are not easily implemented on current architectures. For this reason, a heterogeneous approach results in a more effective design. The target technologies must include software-centric components (DSP, RISC), application specific processors (FFT/Vector Chips), and fixed-function ASICs where performance is paramount. In addition, designs must be able evolve with advancing technology. For effective design across this wide span of technologies, especially considering the dynamic nature of the target applications, significant design tools are required.

Real-time operating systems, such as VxWorks, have enabled the development of software-based real-time systems. While sufficient for a limited number of RISC/DSP processors, these software-only kernels are of limited utility for heterogeneous, dynamically reconfigurable designs. This paper describes a runtime environment that supports programmable hardware as well as the software-programmable components.

The target systems are built on a heterogeneous computing platform including configurable hardware, ASIC and general-purpose processors, and DSP's. An underlying execution environment supports system execution with a common virtual environment. The same virtual execution environment exists across all programmable components, both software and hardware. The runtime supports the execution of a Dataflow specified computation, where the computational elements are distributed over the heterogeneous architecture. Computational elements are designed independently, to operate on input data streams and to generate output data streams. Communications is supplied by the runtime environment, in the form of asynchronous queues. On-device communications occur with efficient pointer-transfer operations (no extra memory copies). Cross-device communications are handled by the runtime environment, and are transparent to the computational nodes. Likewise, communication between dissimilar implementation technologies (software-to-FPGA, FPGA-to-ASIC, etc) are transparent, with required handshaking and data translation operations.

The runtime environment enables seamless integration of the different implementation technologies. Since each component can talk to any other implementation technology, the designer can rapidly mix-and-match components and implementations to achieve desired performance. This also contributes to ease of technology migration. Adding a new type of DSP or FPGA requires only a port of the runtime system and access to software libraries or FPGA IP components. The rest of the system remains unchanged.

The runtime environment also manages the dynamic system reconfiguration, including software reconfiguration for the parallel DSP's and hardware reconfiguration for the FPGA's in the system. Dynamic reconfiguration is managed such that all communications and memory management remains consistent during reconfiguration. Project goals require timeline guarantees to be supported for a reconfiguration event.

The low-level runtime environment is not sufficient, by itself, to design reconfigurable architectures. A companion paper describes aa *model-integrated* approach that is used in the design capture and synthesis of these systems. The Model-Integrated approach defines a domain-specific graphical system design environment, customized to the needs of the reconfigurable systems designer. The tools capture system requirements, algorithm design information and alternatives, and the resources available for system

implementation. This information is represented as a set of *Multi-Aspect Models*. A model interpretation process uses these models to create a fully functional system. This process generates hardware/software architecture specifications, executable/synthesizable code, and a run-time Configuration Manger allowing dynamic adaptation to changing environments while the synthesized system is on-line. The synthesis process optimizes hardware/software architectures for user-definable cost functions such as weight, power, algorithmic accuracy and flexibility

This project is a DARPA/ITO Adaptive Computing Systems funded effort, involving close cooperation with US ARMY/AMICOM.