



Accelerating Processor-based Systems

by Farzad Zarrinfar (VP of Worldwide Sales & Marketing), Bill Salefski (VP of Engineering), and Stephen Simon (Director of Sales & Business Development), Poseidon Design Systems

Designing an efficient processor-based system architecture with overall system performance optimized for a specific application is not trivial, requiring skills and technology similar to those employed by supercomputer designers. Accomplishing this feat requires new tools and methodologies to augment the EDA flow for both traditional ASIC design and the new class of programmable SoCs, e.g., FPGAs. Architectures and performance must be verified early in the design cycle; the designer cannot wait until RTL development to discover their architecture does not support their system requirements.

Problems to Solve

Current processor tools have not kept up with the challenges of the new high-performance systems. Designers need tools to explore and exploit performance, power, and cost opportunities in their processor-based designs. Designers need tools to help them address and overcome:

- Inadequate processor performance to support an ever-increasing functional demand;
- Memory hierarchy and system architecture bottlenecks;
- Increasing design scalability, i.e., enhancing the capability of the design without radical redesign work.

Tools that solve these problems will enable designers to quickly model the system architecture and explore tradeoffs between performance, power consumption, and cost.

application can be a daunting task without a tool to help objectively measure and compare them.

Furthermore, although various IP now come ready to connect to popular busses to simplify assembling them, the designer must properly set up their bus capacity and topology to weave them into an efficiently functioning system. How well a design team handles these challenges may well determine the success of the entire design program.

Design Scalability

As a product evolves, performance, power, and/or cost pressures often require the designer to scale the original design in one of these dimensions. Frequently, the solution is to move software functions into hardware, so as to:

- Increase performance - A dedicated hardware block will greatly accelerate the software function;
- Lower power consumption - A hardware block can use a much lower clock rate than an equivalent software-implemented function to get the same level of performance, thereby reducing power consumption;
- Lower system cost - A hardware block can often eliminate the need for one or more processors while still meeting product performance requirements;
- Protect IP - A hardware block is much more difficult to reverse engineer than software, which is subject to straightforward code disassembly and reverse engineering.

Traditional methods for moving functions to hardware required major effort on the part of the system designer. These require architecting new data flows through the system and developing a new software flow with an interface to the hardware. These programs require an expertise in processor design as well as the time to develop and test the additional hardware. Designers need a tool to provide scalability in their designs. This tool must provide a quick and easy flow to evaluate the system design and then generate additional hardware to support the new requirements.

Poseidon's Triton Tool suite enables the designer to quickly develop an optimal architecture. The tool suite was created specifically for processor-based systems that require efficient robust architectures with the need to

optimize performance, power and cost. Triton consists of two main tools:

- Triton Tuner - a system and software analysis tool;
- Triton Builder - a hardware accelerator synthesis tool.

Triton Tuner is a simulation and analysis environment based on SystemC. Tuner co-simulates the hardware and software system, identifying inefficiencies in the design. Triton Builder synthesizes algorithm-aware accelerator blocks to allow the designer to move algorithms from software to hardware. Together, these tools provide a system design and optimization environment that unobtrusively plugs into an existing design flow, enabling designers to quickly make dramatic improvements in the performance and power consumption of their application.

Design Flow

The Triton Tools allow the designer to use either tool independently, or both together as an integrated suite. A typical system design flow is usually an iterative process where the user analyzes the system performance, determines inefficiencies, modifies the system then checks the resultant performance. The Triton tools aids in this process by providing tools that accelerate the process of identifying the problem areas, and an integrated flow that allows the user to move between the tools to develop the optimal architecture. The typical flow (Figure 1) consists of:

- Triton Tuner profiles the ANSI C code, reveals bottlenecks in the code or architecture and eliminates the inefficiencies;
- Triton Builder partitions processor-intensive algorithms in ANSI C into hardware and generates a hardware accelerator;
- Triton Tuner verifies that the new system performs to the desired level.

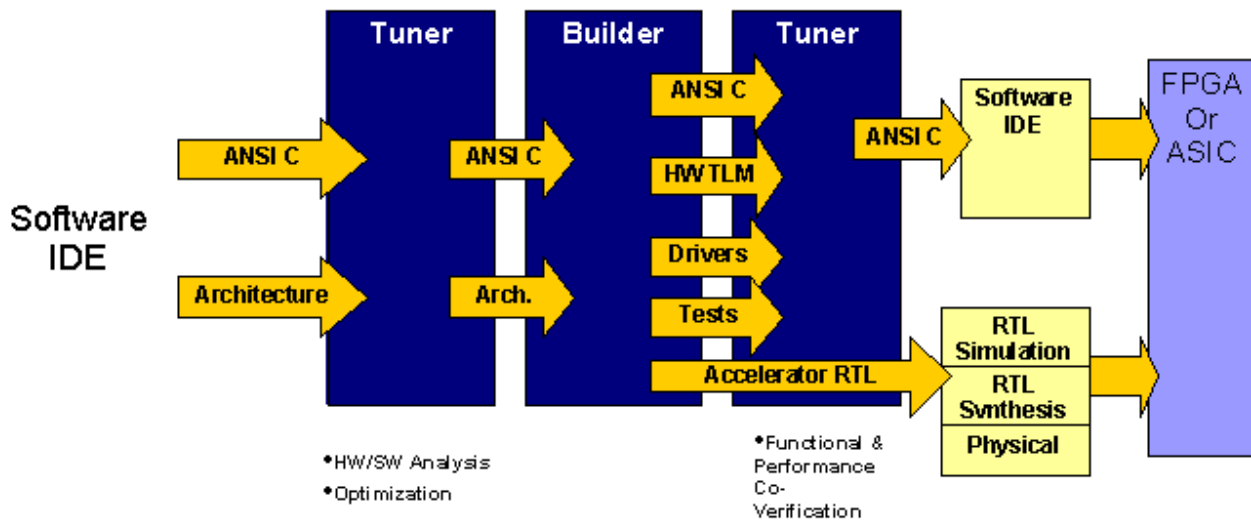


Figure 1 - Poseidon Tools Flow

Tuner takes in the application source code in ANSI C and an architectural description. The user then performs system analysis and optimization on the simulated system. If additional performance gains are required, the designer uses Builder to further analyze and partition the system. The architecture and source code are then transferred into Builder, which generates a hardware accelerator moving selected loops and functions into hardware. Builder generates: modified source, transaction-level models of new hardware, drivers, test vectors and RTL for the accelerator. The designer then uses Tuner to verify the new design meets the system requirements.

Interface to Existing Tools

Triton Tools augment existing design tools through standard interfaces, including:

--ANSI C - Tuner and Builder accept and generate ANSI C for software descriptions.

--Compilers - Tuner and Builder use the existing compilers and build environment; Triton does not force the designer to adopt a new compiler. All Builder-created drivers follow standard ANSI C and pass through the existing flow.

--Synthesizable VHDL and Verilog RTL - All Builder-generated RTL is synthesizable VHDL or Verilog RTL that common synthesis tools such as Synplicity will accept.

--VHDL and Verilog Testbench - The Builder-generated testbench is compatible with standard VHDL and Verilog simulators.

Driver Generation and Integration

Builder simplifies the task of modifying the code to utilize the new hardware accelerator. The tool generates all of the drivers necessary to utilize the new hardware. The application code is also modified to reflect the new partition. Builder takes the original source code and creates a new copy of the application code with the algorithms that were moved to hardware extracted and the drivers inserted to invoke and control the accelerator. The tool provides the designer with executable code that runs on the new hardware and matches the functionality of the old architecture. The generated drivers contain:

--Downloads of the code for the accelerator control unit. This is commented machine code so the designer can update it if they wish.

--Storage of the accelerator parameters for the particular execution of the accelerator, including pointers to data location, coefficient updates, etc;

--Trigger for the accelerator execution;

--Polling loop, if the designer selects polling as the method to monitor progress;

--Loading from the accelerator parameters and status from execution.

The driver makes use of the accelerator transparent to the rest of the software. The designer can use either polling or interrupts for notification of accelerator completion.

Testbench and Verification

Builder creates the testbench to verify the functionality of the accelerator. The design can then be evaluated and verified in Tuner. Builder creates the

necessary files and models to execute the updated system with the accelerators using the high-speed transaction-level simulation in Tuner. This enables the designer to verify the performance of the new architecture and verify the functionality.

Summary

In short, Poseidon's Triton Tool suite enables the designer to add sophisticated hardware acceleration to their processor-based system. This enables designers to achieve higher system throughput, reduced power consumption and cost as well as shorten design cycle.

by Farzad Zarrinfar (VP of Worldwide Sales & Marketing), Bill Salefski (VP of Engineering), and Stephen Simon (Director of Sales & Business Development), Poseidon Design Systems

January 18, 2005