

Accelerate Image Applications with Poseidon ESL Tools

Triton system-level design tools simplify the creation of high-speed solutions for imaging applications.

by Stephen Simon
Sr. Director of Marketing
Poseidon Design Systems
ssimon@poseidon-systems.com

Bill Salefski
VP of Technical Marketing
Poseidon Design Systems
bills@poseidon-systems.com

The number of embedded processor architectures implemented in today's image-processing systems is growing. Designers value the ability of processors to perform complex decision and combinational processes more efficiently than hardware logic. As a result, these designers are moving to a processor-based design that combines the advantages of processors with the inherently fast processing rates of hardware accelerators.

Image-processing applications are rich with signal-processing requirements. FFTs, FIR filtering, discrete cosine transforms (DCT), and edge detection are all common functions found in image-processing algorithms. These functions are computationally

intensive and can severely impact the performance of processor-based solutions. You can design these functions in hardware to elevate performance, but this requires a large amount of manpower and expertise, which may not be available in today's shrinking product design cycles.

Designers seem to be caught between the poor performance of processor-based solutions and the long design times of discrete hardware implementations. What they need is a way to bridge the gap and deliver a system solution that meets performance requirements without having to design discrete solutions with RTL.

Triton System-Level Tools

Poseidon Design Systems offers a way out of this dilemma. The Triton Tool Suite greatly simplifies the process of creating high-performance systems without the time-consuming task of designing hardware with RTL. The suite comprises two tools, Tuner and Builder. Tuner is a SystemC simulation, analysis, and verification environ-

ment; Builder is a hardware accelerator generation tool. Triton tools support PowerPC™ and Xilinx® MicroBlaze™ processor-based platforms.

Triton Tuner

Performing proper system analysis and design is critical when assembling any high-performance imaging system. For example, you must evaluate many architectural issues that can critically impact the performance of the entire system. Because these systems require complicated software and hardware architectures, you must also evaluate the performance of proposed architectures early in the design process rather than waiting for the working system to meet your desired performance goals.

A number of factors determine system performance beyond just the available processor MIPS. Some of these factors are proper cache management, shared-bus congestion, and system topology. With Poseidon's Tuner product (Figure 1), you can co-simulate your hardware and software

systems in the SystemC environment using transaction-level models (TLMs). You can then evaluate system operation and performance and make critical design decisions to develop a robust, efficient solution.

Tuner uses transaction-level modeling to provide the proper level of abstraction and fast simulation speeds. This environment also provides data visualization capability and event coherency, giving you a powerful tool in analyzing and verifying the performance of your target architecture. Tuner outputs data not just on processor performance, but also on cache performance, traffic congestion on shared buses, and efficiency of the memory hierarchy. Tuner links hardware events such as cache misses to the line of code and data accessed at the time; you can gain insight into the software profile and how effectively the system architecture supports the software.

Triton Builder

Hardware/software partitioning also impacts architectural decisions. Determining what portions of the algorithms need to be in hardware to achieve your desired results is not a simple task. In a traditional design flow, a long process follows the partitioning decision to build the hardware and test the resultant system. Changes at this point in the partition are very costly and will greatly impact the delivery schedule.

With Poseidon's Builder tool (Figure 2), you can quickly create a hardware/software partition and then evaluate the performance of the proposed architecture. Builder allows you to easily move critical portions of the selected algorithm to a custom hardware accelerator. Builder inputs algorithms written in ANSI C and creates a complete system solution. No special programming methodology or custom language is required. The tool generates the RTL of the accelerator, memory compiler scripts, test bench, modified C algo-

rithm, software drivers, and TLM of the resultant accelerator hardware. Builder generates all of the parts required to implement the accelerated solution.

To complete the design loop, you can import the accelerator TLM that Builder

created into Tuner and quickly evaluate the performance of the selected partition and resultant hardware. This provides you with almost instantaneous feedback about the effect of your partitioning decisions. The Tuner simulation environment displays system details that evaluation board platforms cannot. This tight loop enables you to quickly test different approaches and configurations, making key trade-offs in performance, power, and cost.

Builder supports different architectures to support different application data requirements. For large data requirements, Builder instantiates a bus-based solution (Figure 3) (such as the PowerPC PLB based on direct memory access (DMA) engines to rapidly move data into and out of the hardware. For smaller data sets or lower latency applications, Builder provides a tightly coupled solution that interfaces to the processor through the auxiliary processor unit (PowerPC) or Fast Simplex Link (MicroBlaze processor) interfaces. For very large data requirements, Builder supports the Xilinx multi-port memory controller (MPMC). This provides for large external fast memories to hold the data and overcome bandwidth limitations on existing processor buses.

With all of these options, Builder automates the process of programming the interfaces, communicating with the accelerator, and scheduling the data. With these two tools you can make critical design trade-offs and quickly realize a high-performance imaging system that meets your design requirements.

Radix 4 FFT Example

Let's look at a typical image-processing application using a fast Fourier transform. We chose an FFT for its popularity and because most designers are familiar with it. With Triton tools, you can not only quickly implement common functions such as FFTs; you can also implement the proprietary algorithms companies develop to differentiate their designs.

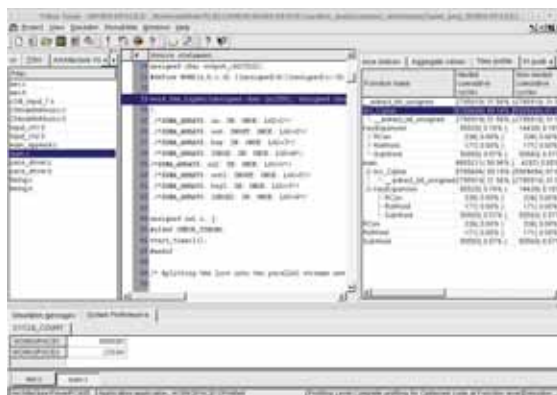


Figure 1 – The Tuner main screen displays software profiling data.

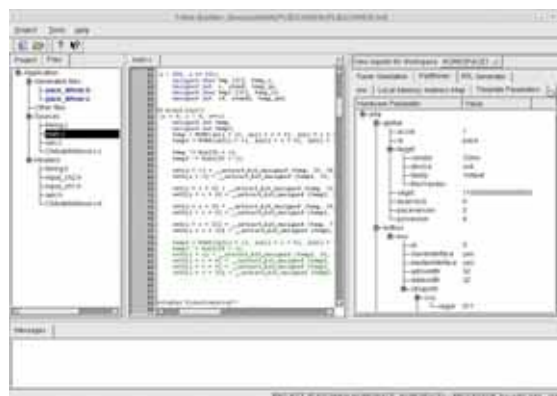


Figure 2 – The Builder main screen displays accelerator parameters.

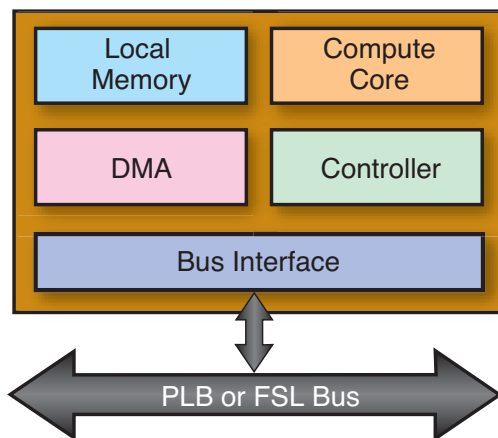


Figure 3 – Bus-based accelerator

Along with the ability to partition loops into hardware, Builder gives you the flexibility to easily move functions into hardware as well.

The specific FFT algorithm in our example is a Cooley-Tukey Radix 4 decimation in frequency written in standard ANSI C. The algorithm runs on a 300 MHz PowerPC on a Virtex™-II platform. The algorithm executes in 1.3 ms. This gives a maximum frame rate of 769 frames per second when 100% of the host processor is dedicated to the FFT task. Processing the selected algorithm in place makes efficient use of the memory resources. The software comprises five main processing for loops, one for each pass of the FFT algorithm.

The first task is to profile the FFT algorithm to determine the best candidates for moving to FPGA fabric. Tuner reads the architectural description for the Xilinx Platform Studio (XPS) and simulates the algorithm. The five loops performing the Radix 4 butterflies consume between 15% and 23% of the cycles used in the entire algorithm. These are all good candidates for acceleration and C-to-RTL conversion into dedicated FPGA logic.

Using Builder, you can determine how much of the algorithm to move to hardware to accelerate the application. If the design needs a 50% improvement in processing speed, it would be very inefficient to implement an entire FFT in hardware. In this example, by selecting all five loops for partitioning, more than 99% of the processor cycles were moved into the hardware accelerator. It would be just as easy if you required less performance to select any number of loops to match the desired acceleration of the system, thus saving space in the fabric for other uses.

Along with the ability to partition loops into hardware, Builder gives you the flexibility to easily move functions into hardware as well. With this flexibility, you have the freedom to match the partition to the requirements and structure of the specific application.

Builder also provides a C-lint tool that can help you determine which existing code is synthesizable. If any non-synthesizable structures exist within the algorithm, the

tool flags that section of code and reports the condition that makes it unrealizable into hardware. The tool also suggests enhancements and modifications that might be useful in implementing more efficient algorithms in hardware.

The partitioning process is as simple as selecting the loops or function that you would like moved into the accelerator – the tool performs the rest. If you wish, you can control the optimizations and settings to optimize the solution, making trade-offs to best match the desired system.

Builder also takes care of integrating the partitioned code back into software by generating a software driver. The tool determines the scheduling of the variables for the resultant hardware and ensures that the data will be available when the hardware requires it. Builder determines memory sizes and memory banking requirements to efficiently process the data in the central compute core. Builder then creates the memory compilation script for the LogiCORE™ system to build the optimized memory structure.

The bus-based accelerator is appropriate in this example because of the moderate amount of data required. This solution has its own DMA, which enables the accelerator to access the data in system memory and run autonomously to the processor. Builder automatically schedules and programs the accelerator to transfer data into it using the DMA, and similarly transfer the results back out into system memory.

Because of the tight integration with the Tuner environment, you do not have to guess the effect of the partitioning decisions and architecture options created using Builder. With each new option you can quickly generate the TLM and simulate the proposed solution using Tuner. This gives you instant feedback as to how fast the resultant solution is as well as how it integrates into the existing system architecture. When the software and hardware meet the desired performance, you can automatically generate the pcores for the hardware accel-

erator to import back into XPS. All of the required files and scripts are generated, allowing you to continue on with the existing design flow.

By using this automated technology, you can easily create and verify speedups of as much as 16x. This pushes the maximum frame rate of the FFT application to 12,000 frames per second for each channel of FFT implemented, with the processor load decreasing to less than 5%.

You can see with our FFT example the ease of use of Triton ESL tools and how with little effort you can generate high-performance systems. This example took only two man weeks of effort and without any manual modifications to the resultant system. If you require additional processing capability, it is a straightforward process to modify the software to support a streaming architecture. Creating concurrency between passes or creating multiple channels with separate accelerators, you can quickly achieve an overall acceleration of 48x to 80x.

Conclusion

The Triton Tool Suite is a powerful tool for the development of image-processing architectures. The Triton Tool Suite enables you to efficiently perform and verify architectural development, hardware/software partitioning, and the automated hardware generation of key digital signal processing algorithms. Tuner and Builder not only provide large savings in your design efforts; they allow you to create more efficient high-performance and scalable solutions. The tools are highly automated, yet give you the flexibility to optimize your solution and meet your system design requirements.

The system also automates the interface between the Triton Tool Suite and the Xilinx XPS system. This solution is optimized for computationally intensive algorithms found in audio, video, VoIP, imaging, wireless, and security applications. For more information, please visit our website at www.poseidon-systems.com and request a free 30-day evaluation. 