Algorithm Development for Future High Performance Systems
Jigsaw: An Early Case Study

Imran Khan
SoftServ Intl. Corp. 41 Eliot Hill Road, Natick MA 01760
Ph: (508) 655-0097, i.khan@softserv-intl.com

A High Performance System in the future will have a combination of multi-core processors, RISC, DSP, FPGA and ASIC nodes to run increasingly complex algorithm in real time. The need to succed the first time will increase the pressure to get the complete design cycle automated.

Most sophisticated algorithms use C, C++ or MATLAB. While these languages have the capabilities of implementing the computation with relative ease, the result is usually not something that can be readily used in real world systems. These languages lack the semantics and constructs necessary to map the algorithm to the computing architectures that are eventually required in the final systems. To efficiently map an algorithm to the new architectures the language should be able to handle concurrency, fixed point data types of varying lengths, model communication channels along with computation, allow extensibility of data types, support design process at all levels etc.. The only language that can handle all these challenges is SystemC. SystemC has been chosen by IEEE to be the design standard (IEEE 1666) for designing increasingly sophisticated system on chips that require concurrent modeling of system, hardware and software architecture. Also relevant is the ability to encapsulate and share IP without the need for full disclosure. SystemC enforces a hierarchical and modular design with well defined interfaces. Early modularization not only eases the management of concurrent development throughout the design cycle but also makes the configuration management of the deployed system easier. SystemC provides support for both un-timed and timed simulation models. The algorithm designer can incrementally add sophistication to both the transactional and computational model. In parallel the designer can explore the performance characteristics of mapping the algorithm to different combination of hardware architectures by using the timed models.

Jigsaw Ladar

DARPA's Jigsaw program required that the final airborne system be accurately modeled so that concurrent algorithm development could take place among the collaborating groups. This provided an opportunity to take the new development paradigm through all the design and development phases of a real system that needed a variety of compute nodes to achieve the performance goals. The original C++ model was restructured using systemC.

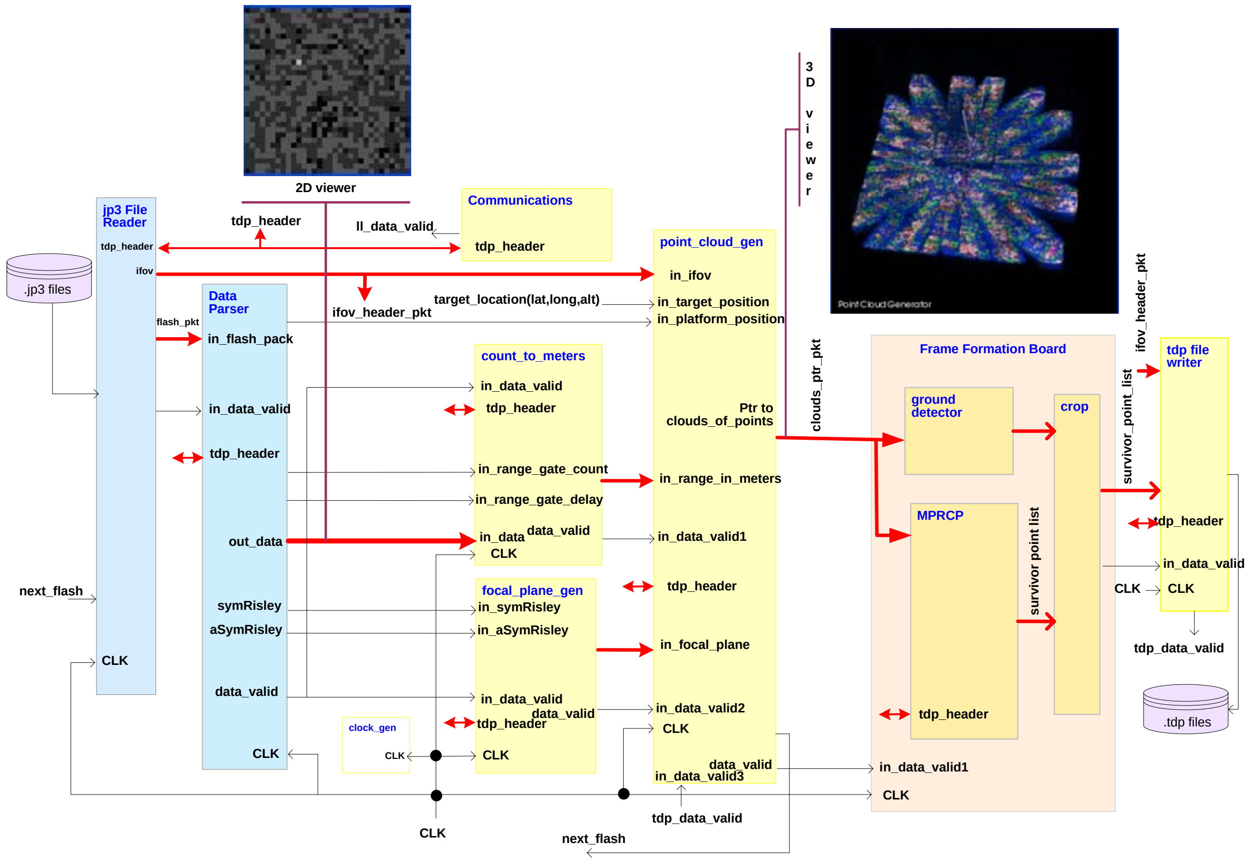


Figure 1 Block Diagram of Jigsaw systemC model

The color of the modules indicates the different physical architecture for implementing them. Yellow modules run on PowerPCs and the orange modules on FPGAs. Both control and computation were modeled. This was an un-timed model

Custom Data Types

Custom data types were added to handle both the 2D and 3D data that flows through. A custom control data type "tdp_header" helped in modeling the control stream. When the data size grew to a point where it started impacting modeling performance, a custom pointer packet data type was defined that contained a pointer to the allocated data to the down stream modules.

Data Visualization

Data Visualization probes can be inserted both in modeling and simulation. Visualization Tool Kit based visualization window assisted in debugging both 2-D count data as well as 3-D point cloud data.

Mixed Mode Simulation

The modules targeted for FPGA implementation were coded in VHDL. The systemC based C model provided the test input data as well as a reference output data. ModelSim simulator provided the needed mixed mode simulation to implement the test bench. Both pre and post synthesis simulation over a prolonged data run validated the implementation. This approach provided sophisticated unit and system level test bench data generation that helped in robust validation of the machinery.

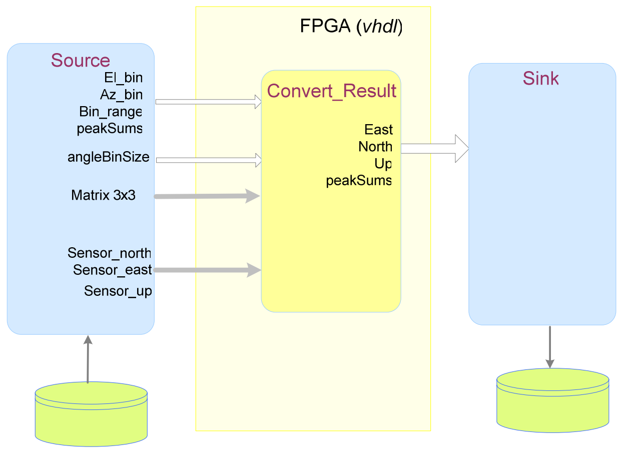
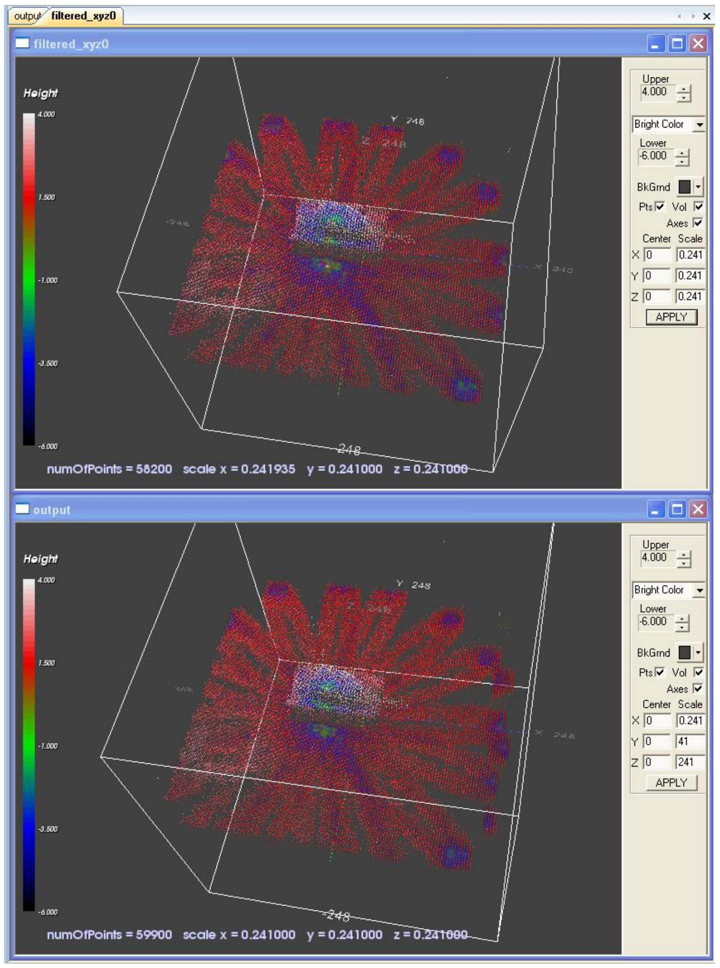


Figure 2 SystemC based Test Bench for a VHDL module

Visual Validation

Simple vector based validation becomes too simplistic when complex data streams are all interacting in real time with each other to generate the output. The size and duration of simulation is long, resulting in very large data sets that can only be viewed using multi dimensional visualization techniques. 3D rendered images of tank in the clear are shown below that provide visual comparison of ideal C model and synthesized VHDL implementations of the same algorithm.



numOfPoints = 58200 scale x = 0.241935 y = 0.241000 z = 0.241000

numOfPoints = 59900 scale x = 0.241000 y = 0.241000 z = 0.241000

Figure 3 SystemC (top) and Synthesized VHDL

While some of the basic tools are available, there is considerable room for EDA vendors to automate the design and development process. Jigsaw experience has shown that small effort in adapting the algorithm development process can have huge dividends in time to deployment.

References

[1] Imran Khan, "Learn to Manage All Kinds of Complexity with SystemC", *Electronics Design*, Sept. 2005