Architecting High-Speed Routing Systems

3G/IMT2000 UDP/TCP Blade Processor System

Introduction

VisualSim is a modeling and simulation software package used for the performance and architecture analysis of a design that is to be implemented. Using VisualSim, Architects can evaluate metrics such as bottlenecks, throughput, latency and utilizations. The protocols, traffic and hardware can be described in the environment. Moreover the Architects can describe the functional tasks and map them to hardware and software without writing the actual software code.

This document describes the modeling of a Universal processor blade server that is used in IMT2000, 3G and other wireless data application for transferring UDP/TCP data. The primary aim of this model is to evaluate resource consumption of various packet processing tasks, input and output throughput, hardware utilization, predict bottlenecks and system latency.

The system has been modeled in VisualSim as a combination of dedicated hardware, shared hardware, software resources, traffic generation and packet processing tasks.

The VisualSim model provides a very powerful but simple way to analyze and architect new and derivative products. A trained VisualSim modeler can construct the model discussed below in about 3 weeks. Constructing the model in a code-based simulator with no pre-built modules would take 2-3 thousand lines of code and 3 month effort. The total time savings using VisualSim would be about 10 weeks for this project or 75% of the total time required. Moreover the flexibility of the model to make quick changes adds powerful optimization power. The model was simulated for 30 milli-seconds of operation and this took 18 seconds to execute on a 2.8 GHz machine.

A traffic description in Excel would have provided some deterministic information about the system. The impact of contention at resources, sharing of resources, describing the data flow and incorporating analysis windows are impossible.

Goals of this experiment

Models such as the one constructed below are mainly used for two purposes- to explore possible architectures for new products and to validate current products for a large variety of network conditions. The results can be compared against network traces, standard industry benchmarks and results from previous generation products.

There can be three types of analysis conducted in VisualSim- performance, behavior or functional and architecture.

1. The behavior analysis would be protocol accuracy, effectiveness of algorithms against network errors, retransmissions and out-of-order arrivals.

2. The performance analysis is to test the end-to-end latency, total throughput for a variety of traffic loads and bottlenecks.
3. The architecture exploration is to determine the distribution of functional tasks on various processors on the board/ system, custom hardware requirements and processing cycles required for various tasks on the network traffic- cycles required for the header processing and routing table lookups.

The model explained below is a sub-set of a processor blade that is used in high-performance routing systems including 3G and Network Routers. The purpose of this experiment is to demonstrate the use of VisualSim to evaluate the performance and validate the architecture design. The model has been built for easy extension to add all the other details that have not been specified.

Advantages of this approach

System-level modeling using VisualSim enables organizations to reduce risk, maximize product quality and predict project schedule. The experiments are conducted during the specification stage. So architecture errors and limitations in performance can be identified before scheduling implementation. Also, test and diagnostics engineers can start developing verification environment earlier, thus sub-systems can tested sooner. The VisualSim model enables any changes to the specification during implementation to be validated against this golden reference.

The biggest advantage of modeling with VisualSim is significantly reduced modeling time. VisualSim enables architects and systems engineers to very quickly construct models of their proposed system. These models are constructed using the standard performance block libraries and scripting language in VisualSim.

All parts of the proposed system can be described and simulated within VisualSim. This includes queuing, protocols, board architecture, embedded software and traffic. Moreover the basic library is sufficient to define all of these entities.

This current model has been built with less than 20 unique blocks instantiated a total of 100 times. The hardware has been modeled using the scheduler block. Each software task has been modeled by one computation block and a behavior task issue block. In addition, complex traffic generators have been constructed to model the exact profile of TCP/UDP and ATM traffic.

The use of parameters enables the modeler and implementation engineers to experiment with multiple hardware characteristics, input traffic rates and number of users without modifying the model.

The hierarchical and graphical models provide easy understanding of the system and can be used as the executable specification document.

The experiments conducted with VisualSim in the early design stage provide all the corner cases that need to be verified. The test vectors generated at this stage can be reused for the verification of the board and system. The results can be compared against the output of the testers.

Board Specifications

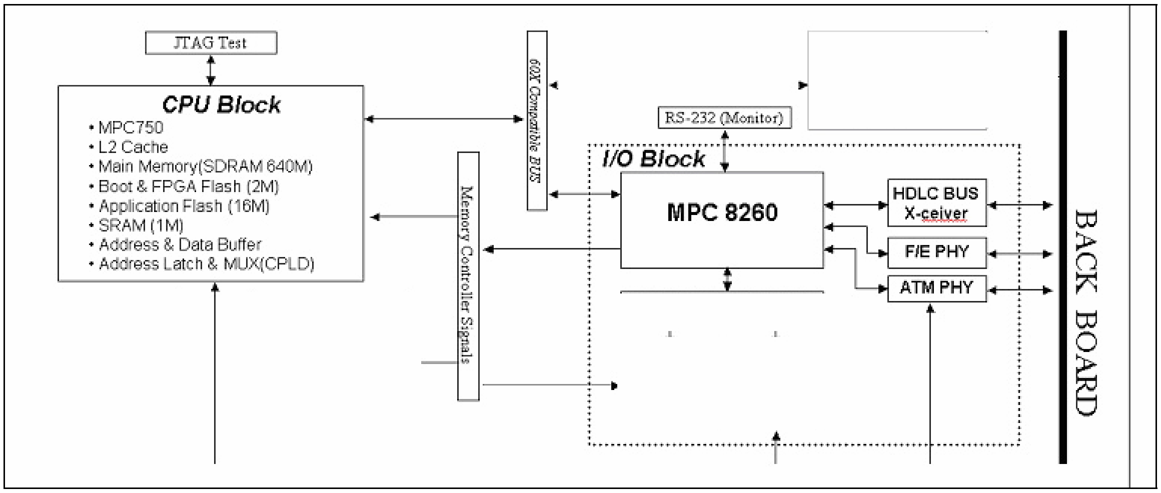


Figure 1 Universal Common Processor board Assembly – Generex2000 used in IMT2000 Systems

Num	field	spec
1	Processor	✓ CPU: XPC755(733MIPS, 400Mhz) ✓ I/O : MPC8260(280 MIPS, 200Mhz)
2	Main memory	384 MByte
3	L2 Cache	1MB
4	Local memory	128MB
5	Processor to Processor Interface	60X Controller Bus
6	Real Time Operating System	VxWorks 5.4
7	IP Packet Size	200-1500 bytes

Table 1 Hardware, Software and Traffic Characteristics

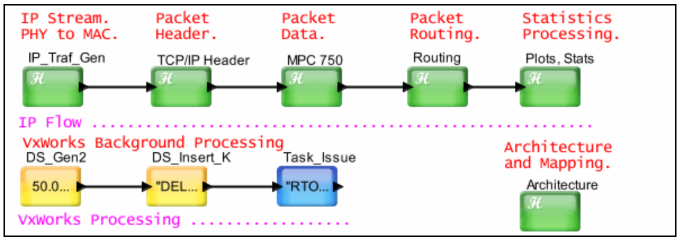
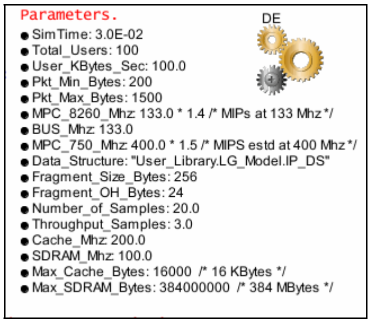


Figure 2 Top-Level VisualSim Block Diagram of the Processor Blade

Model Description

The VisualSim models a part of the hardware, accurate details of the incoming traffic and 5 tasks performed on the data packets. Most traffic, hardware, software processing cycles, users and other characteristics have been listed as parameters. These parameters can be modified to create different scenarios. For example, modifying the User_KByers_Sec from 100.0 to 193.5 will give ATM packets. Also, changing the Fragment_OH_Bytes from 24 to 8 bytes, 12 bytes (IPv4) or 16 bytes (IPv6) will make the TCP system into UDP.



Parameters.

DE

- SimTime: 3.0E-02
- Total_Users: 100
- User_KBytes_Sec: 100.0
- Pkt_Min_Bytes: 200
- Pkt_Max_Bytes: 1500
- MPC_8260_Mhz: 133.0 * 1.4 /* MIPs at 133 Mhz */
- BUS_Mhz: 133.0
- MPC_750_Mhz: 400.0 * 1.5 /* MIPS estd at 400 Mhz */
- Data_Structure: "User_Library.LG_Model.IP_DS"
- Fragment_Size_Bytes: 256
- Fragment_OH_Bytes: 24
- Number_of_Samples: 20.0
- Throughput_Samples: 3.0
- Cache_Mhz: 200.0
- SDRAM_Mhz: 100.0
- Max_Cache_Bytes: 16000 /* 16 KBytes */
- Max_SDRAM_Bytes: 384000000 /* 384 MBytes */

Table 2 List of Parameters used for exploration in VisualSim

The top-level of the VisualSim model shows 17 parameters relevant to the hardware, traffic and users. More can be added to the model as it is extended to include other tasks and hardware elements such as CAM, policy lookup and security. In addition, every block has its own set of parameters that are relevant to the operations described in that block. For example, the Routing hierarchical block at the top-level has the upper and lower range of the number of processing cycles.

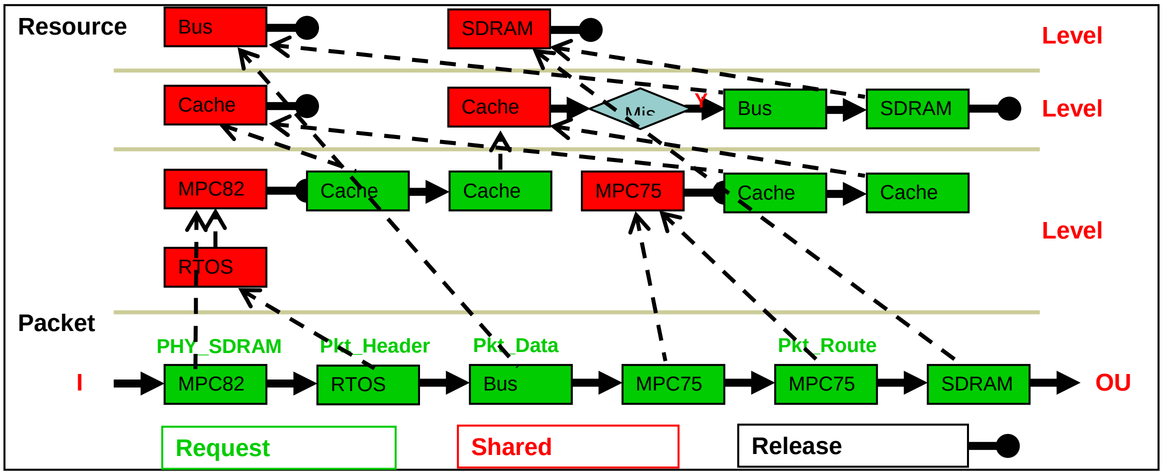
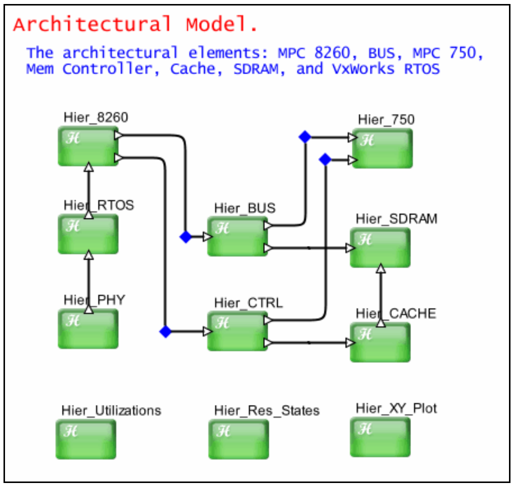


Figure 3 VisualSim Flow diagram and listing of Resources in the VisualSim model.

The model has been split into three parts- hardware architecture (Architecture and Mapping block), VxWorks scheduling overhead (yellow blocks at the top-level) and the 5 data flow hierarchical blocks describing the traffic and packet processing operations. To extend this model, the additional packet processing functions must be added to this sequence. The Architecture block can be extended to add the additional blocks not covered in this model.



Architectural Model.

The architectural elements: MPC 8260, BUS, MPC 750, Mem Controller, Cache, SDRAM, and VxWorks RTOS

Figure 4 VisualSim Block Diagram definition of the Hardware and RTOS

The Architecture block describes the performance characteristics of the MPC8260, MPC750, SDRAM, Cache, 60X Controller Bus and the memory Controller Signals. The important parameters of this block can be seen by double-clicking of the Architecture hierarchical block from the top-level of the model. These components have been described using the Scheduler and Expression processing blocks. More details for each hardware component can be added easily by extending the scheduler with other VisualSim blocks and the SmartMachine scripting language. Examples of hardware extension include the CAM, local SDRAM for the 8260 and, I- and D- cache. The connectivity between the blocks in the hierarchical level is simply the graphical representation of the data flow. The command to process the packets are initiated in the data flow blocks and are virtually communicated to these Architecture blocks. The Architecture blocks simply do the necessary processing and return the results back to the data flow blocks.

The first block in the data flow describes the traffic generation, input rate limiter, fragmentation and PHY operation. In this model, the PHY operation has been limited to writing data to the SDRAM. The SAR operations can be easily added using the basic blocks and the Finite State Machine. The IP message blocks models the incoming traffic and the ATM data rate limiter. The incoming traffic has been modeled as ATM traffic that is generated by 100 users with 53 bytes each. The incoming traffic is restricted to the input port rate. The AAL5 has been modeled with an overhead header of 15 bytes. The behavior of the AAL5 can be added easily into this model using the Finite State Machine Editor available in VisualSim. This block also provides a fragmentation function using a Queue + while loop blocks. This fragmentation function is used to segment the packets into smaller packets that are directly processed by the hardware. The PHY function in this model is limited to storing data in the SDRAM. This can be extended to include the SAR function by adding a few additional blocks.

The Packet Header, Packet Data and Packet Routing blocks contain two types of blocks inside the hierarchical level- one computes the number of processing cycles on the hardware resource and the second schedules the activity on the hardware resource. The hardware resources are centralized in the Architecture and Mapping block. The Packet Routing block has some additional details related to reading from the SDRAM. This is done when the routing tasks have been completed and the data needs to be transferred from the SDRAM to the output port. The data is also removed from the SDRAM.

The last block in the list simply consolidates the entire statistics gathered from the various data flow and prepares the data for presentation in the results windows.

Results

A number of graphs and analysis plots can be generated from a single model and a single simulation run. The particular analysis graphs discussed below were generated to match these results against the expected results from a Tester. The first set of graphs show the input and output throughput. They match very closely, indicating that the hardware resources can support the packet processing for the particular arrival. Modify the User_Kbytes_Sec and you will see the throughput get adjusted.

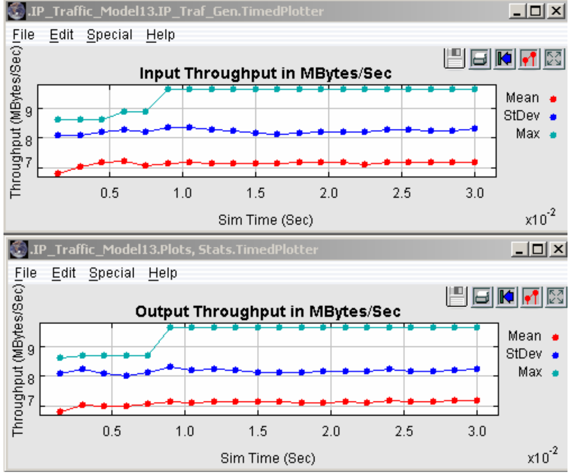


Figure 5 Input and Output Throughput Graphs

The next two graphs plot the resource consumption of the hardware and the RTOS. The Related Resource plot shows the exact times at which the resource was active. The Resource Mbytes/sec plot shows the average processing at each hardware and RTOS resources over the period of the simulation. The processing Mbytes/sec is actually greater than the input and output throughput indicating that each packet is processed multiple times by some of the resources. If the Related Resource plot is zoomed to a small region, you will see the contention for resources. This model does not consider impact of priority but this could be added by adding a parameter.

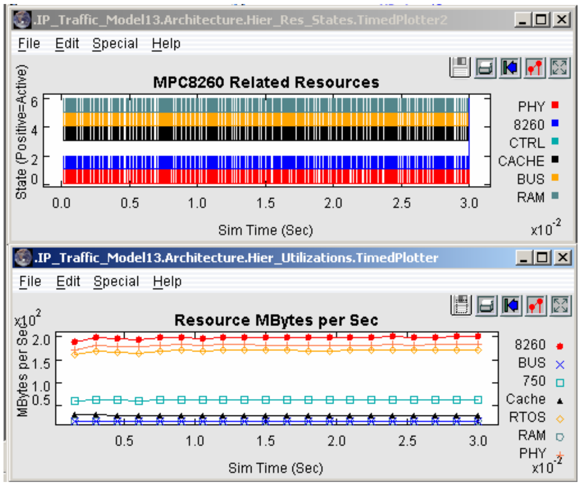


Figure 6 Resource Consumption Graph

This graph simply shows the cumulative Mbytes through each resource. As all the data is first stored in the Cache before being processed and then returned to the Cache after processing, the consumption is the highest there.

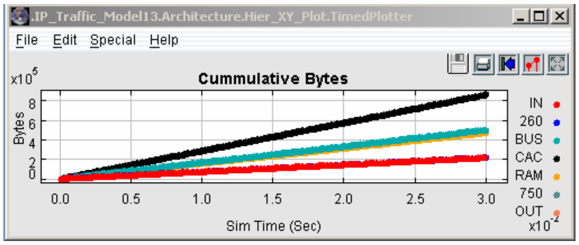


Figure 7 Cumulative MBytes at each Hardware resource

Conclusion

The model indicates that the system has sufficient capacity to handle the input traffic and has room for expansion. This is a working model and can be expanded very easily to add more processing details. Also, adding processor traces or the processor core with the software loaded can add to the accuracy of the model.

The model was simulated for 20 msec and it took 18 seconds to execute. The model can be accelerated by adding more detail in the VisualSim SmartMachine scripting language.

The VisualSim model provides a very powerful but simple way to analyze and architect new and derivative products. A trained VisualSim modeler can construct the model discussed below in about 3 weeks. Constructing the model in a code-based simulator with no pre-built modules would take 2-3 thousand lines of code and 3 month effort. The total time savings using VisualSim would be about 10 weeks for this project or 75% of the total time required. Moreover the flexibility of the model to make quick changes adds powerful optimization power.

A traffic description in Excel would have provided some deterministic information about the system. The impact of contention at resources, sharing of resources, describing the data flow and incorporating analysis windows are impossible.