Reconfigurable Computing-an Innovative Solution for Multimedia and Telecommunication Networks Simulation*

Juanjo Noguera[1], Rosa M. Badia[2], Jordi Domingo[2], Josep Solé-Pareta[2]

[1]Dept. Computer Science, Universitat Autònoma de Barcelona
Edifici C, Campus UAB, E08193 Bellaterra, SPAIN
juanjo@cnm.es

[2]Dept. Computer Architecture, Universitat Politècnica de Catalunya
Campus Nord - Mòdul D6, C/ Jordi Girona, 1-3 E08034 – Barcelona, SPAIN
{rosab, jordid, pareta}@ac.upc.es

Abstract

Sequential network simulation is a high time-consuming application, and with the emergence of global multihop networks and gigabit-per-second links is becoming a non-affordable problem with traditional simulations. New techniques for the acceleration of these simulations based on other hardware architectures are required. Previous approaches to simulation acceleration are based on parallel computing and reconfigurable computing. A short review of most outstanding approaches showing its benefits and problems is presented in the paper.

A new approach based on mapping network simulations on reconfigurable hardware is presented. Most important features of this system are: the acceleration of the simulation by hardware, and the use of a high level network modeling language which allows a transparent use of the hardware by telecommunication engineers. The core of the proposed environment is an automatic tool that compiles the high-level network model and maps the simulator behaviour into the hardware.

1. Introduction

In order to provide multimedia services, such as interactive television, video conferencing and access to digital libraries, new control and management systems for telecommunication networks have to be developed.

While the behaviour of single network elements and small systems have been extensively studied by means of mathematical analysis, testbed prototyping and simulation techniques, the behaviour of entire network control systems, where network elements of different types interact is far understood today.

A number of network testbeds have been set up under various research initiatives [2][8]. These testbeds have demonstrated the feasibility of providing high-speed connectivity and multimedia communication capabilities among small groups of users. Unfortunately, the results produced on such a testbed are generally restricted to a specific system configuration and cannot be used to predict the behaviour of the system when the number of users, services and network nodes is increased.

Network engineers and researchers routinely use simulations in their daily network design, analysis and evaluation tasks. Simulation provides a practical methodology for understanding system behaviours that are either too complex for mathematical analysis or too expensive to investigate by testbed prototyping.

During the network design phase, and before running the simulation, a description (model) of the network architecture is produced, which specifies the network on a conceptual level, comprising its basic structure, functions and characteristics. For such purposes, normally network-modeling languages are used (i.e. OPTNET [23]). This type of languages hide details of how the simulation technique is implemented, so networks engineers only have to concentrate in network modeling. A more accurate study of these languages can be found in [3].

With the emergence of global multihop packet networks and gigabit-per-second links, the network simulation community is faced with new and significant challenges. First, actual packet traffic is dominated by long-range correlations, which means that realistic models

* This work has been founded by CICYT projects TIC98-0410-C02-01 and TEL97-1054-C03-03.

have to be simulated for very long timescales to avoid misinterpreting transient behaviours. Second, network configurations of really large size have to be simulated to study issues such as scalable routing or packet loss correlations in multihop networks. Such features just cannot be captured in small network models. The immediate need for such extensive modeling capabilities for planning, growth management and network management can be found in [1].

The capabilities of conventional sequential simulation techniques are inadequate to address such simulation requirements. Telecommunication networks simulation using sequential techniques is a highly time-consuming application. This affirmation is easier to understand with a practical example. Suppose an ATM connection for a MPEG-2 video flow with a bandwidth of 2.2 Mbits/sec over a link of 155 Mbits/sec. Approximately, this connection will send 5000 cells[1] per second. If a CLR (Cell Loss Rate) of 10^{-6} is desired, the simulation must be able to transmit 10^7-10^8 cells to the connection to ensure a satisfactory interval of confidence. This corresponds to a simulation time of at least 2000 seconds. During this time, the number of cells that will be sent over the output link is near 700 millions. A sequential simulation of this model can take several days [12].

The exposed example demonstrates the existence of a practical necessity to accelerate the simulation of telecommunication networks, given the inefficiency of several days long simulations. Nowadays, the modeling phase of a network using a modeling language takes several days and the simulation phase takes the same order of magnitude. The possibility that this loop has to be repeated to obtain an accurate model of the network being specified, implies a long time to develop and evaluate a functionally correct model.

Clearly, the final objective is reaching the point of implementing a simulation environment or framework, where the simulation phase takes the order of minutes, and the modeling phase remains as the more time-consuming one.

From all explained above, the final conclusion is that an acceleration technique is required to reduce the simulation time, and this possibility relies on the use of other hardware architectures. See Figure 1.

The rest of this paper is structured in the following manner: in section 2, parallel computing based simulation is explained and special emphasis is done in TeD, a modeling framework for telecommunication network simulation based on parallel computers. Reconfigurable computing based simulation is explained in section 3. Basics concepts on reconfigurable computing and a concrete simulation framework are also explained in this section. Section 4 summarises the conclusions from the study of the commented frameworks. Benefits and problems of such frameworks are explained too, and our new approach based on reconfigurable computing to multimedia and telecommunication network simulation is explained in section 5. This new approximation tries to combine the better features of explained frameworks and to overcome the problems observed in these simulation environments. Finally, the conclusions of this paper can be found in section 6.

[1] An ATM cell is composed of 53 bytes, where 5 bytes are used as header and 48 as payload.

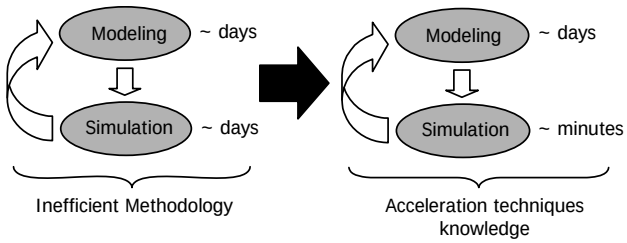

Figure 1 Acceleration Techniques necessity

2. Parallel Computing based Simulation

Parallel simulation is the process of using multiple processors simultaneously for executing a single simulation, with the goal of reducing the total execution time. Several general-purpose parallel simulation environments can be found in the literature [18][10].

Recently, parallel simulation techniques have been applied to the new challenges presented by telecommunication networks simulation. Examples of this type of frameworks can be found in [20][21][22]. Because of its interesting features, a concrete parallel simulation framework will be explained next.

2.1. TeD: Telecommunication Description language

Telecommunication networks specialists are not usually familiar with algorithm parallelization, which is the base for parallel computing based simulation. A simulation environment where the use of the parallel computer (and so, the algorithm parallelization) was transparent to telecommunication network specialists would be really useful.

A concrete implementation of this concept is the environment developed at the Georgia Institute of Technology. A modeling language has been developed to achieve transparency in the use of the parallel computer to telecommunication network specialists: TeD (Telecommunication Description Language). See Figure 2.

TeD is a language specially designed for telecommunication networks modeling. TeD specification (1996/1997) is divided in two parts: MetaTeD [20] and an “external language” [21]. MetaTeD defines a set of concepts to modelate the interactions between entities

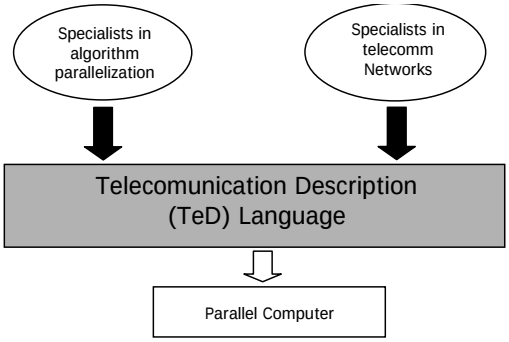

Figure 2 TeD as an multidisciplinar research activity

(network elements) and the structure of these entities. Being MetaTeD an incomplete language, when it is combined with an external general purpose programming language, a complete network modeling language is obtained. Actually, C++ is used for such purposes.

The TeD environment has been developed on top of a general purpose parallel simulation kernel, the GTW which runs over the parallel computer. Any event-driven simulation can be performed using the GTW kernel, using C++ as input language. Networks engineers and researchers use TeD as the modeling language, which afterwards can be compiled to finally obtain C++ code that is the input to GTW. See Figure 3.

Some of the main TeD features are:

- It is a language sufficiently generic and easy of use that can be used to model actual and future telecommunication networks.
- It is a modular language, by which the final user is able to create and use libraries, for TeD code reutilization.
- TeD hides details of algorithm parallelization to networks specialists, as commented before.

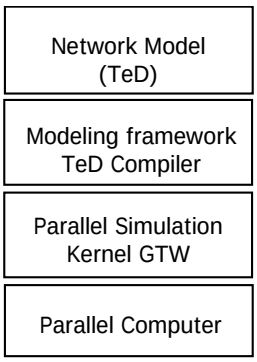

Figure 3 TeD Layered Approach

Actually, TeD is widely extended over the research community, where it's possible to find a lot of people and institutions of recognised prestige developing models in TeD. Examples are: Bellcore (Bhatt), PNNI protocols; Dartmouth (Nicol), IP protocols; Georgia Tech (Zegura), ATM switches; Rutgers (Suvaine), wireless protocols; UMass (Towsley & Kurose), Multicast protocols; MIT (Zhang & Andrews), call admission protocols; Georgia Tech (Srinivas), video traffic [11][29][5].

3. Reconfigurable Computing based Simulation

3.1. Basic Concepts in Reconfigurable Computing

The principal feature of a general-purpose hardware system is that it has a basic architecture where the desired functionality is fixed via software. With this approximation, it is possible to implement a great number of applications at a low cost. The main inconvenient of these systems is that, they can not give the necessary throughput depending on the application requirements. The solution to this problem consists in the use of hardware specific systems where the architecture has been designed to implement a concrete algorithm or application. Clearly, such a solution has a higher cost.

The appearance of new programmable devices, as an evolution of old programmable logic for digital systems design (PAL, PLA and PLD), has made possible the development of a new research field, Reconfigurable Computing [34], which offers an intermediate solution between general-purpose systems and application specific systems. See Figure 4.

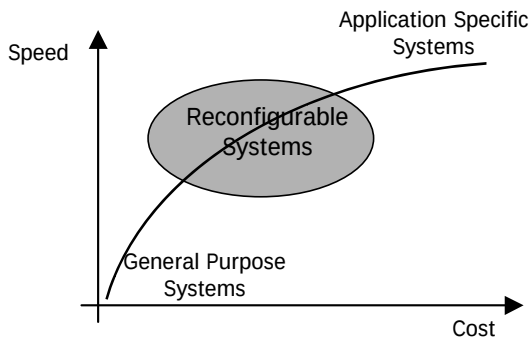

Figure 4 Reconfigurable Systems Features

Reconfigurable Computing means the possibility of repeatedly configure (reconfigure) a hardware to implement different applications and algorithms, thus appearing the reconfigurable systems [17][27][31].

As said before, reconfigurable systems are based on new programmable devices, known as FPGA and CPLD, which thanks to technology improvements has increased

the capacity and processing speed of these devices [30].

Finally to conclude this basic concepts on Reconfigurable Computing, comment the advances in high-level hardware description languages (specially VHDL [9]) and automatic synthesis tools that have significantly reduced the time for hardware system prototyping [15][24]. Commercial hardware synthesis tools are available for mapping VHDL models to FPGAs.

3.2. FAST: FPGA-based ATM Simulation Testbed

A concrete example of a reconfigurable system applied to telecommunication networks simulation, is found in the FAST system (FPGA-based ATM Simulation Testbed), developed at California University at Santa Cruz [12].

The FAST system can be used to implement key elements in the simulation of an ATM network, such as traffic generators, scheduling algorithms, switch architectures and congestion control mechanisms. This system is based on four printed circuit boards, each one based on programmable devices from Altera (programmable devices manufacturer). A single board can be used to simulate an ATM switch, with its associated algorithms. Several boards can be connected to simulate an ATM network (up to four switches).

It is important to say that the FAST system is not a prototyping system. It is a functional emulation system, which can be used to modelate/evaluate the ATM layer, as well as higher layers. Details from the physical layer are not implemented, although delays of this layer can be incorporated to the model simulation. The great flexibility of programmable devices has made possible modeling different parts of the system to be simulated at different detail levels. For example, size and format of an ATM cell can be different from the ones defined in the standards, or it is possible to incorporate only the ATM fields which are interesting for the simulation.

The FAST system has, besides the four circuit printed boards, a host computer which is responsible for configuring the different programmable devices to simulate the desired model. A second functionality of the host computer is to execute the software which controls the simulation and save its results.

The generic architecture of one of these four printed circuit boards is found in Figure 5. It is composed of four traffic generator modules, four input modules and four output modules. Each of these modules is based on a FPGA and static RAM memory (SRAM). The four input modules are connected to the four output modules, as well as each traffic generator module is connected to its corresponding input and output module. With this architecture it is possible to emulate/simulate any architecture of a 4x4 ATM switch.

Moreover, in this board it is possible to find a distributed memory module, which is connected to all input and output modules through a shared bus. This module is clearly present in the board for distributed memory switches simulation, although it can also be used to save intermediate results from the simulation.

Finally, the last module that can be found in the board is the interface module, which has principally two functions: implementing the communication between the board and the host computer and interconnecting different boards when a simulation of several switches is desired. To implement this function, the interface modules of two boards are connected via a simple handshaking protocol.

A short commentary about the number of ports in the switch to simulate. In a first though, a 4x4 switch can be seen as a not appropriated switch to study the scalability of scheduling algorithms, but in the literature there are some papers demonstrating that few number of links is not a limitation to study scheduling algorithms scalability, given that the important thing to study scalability is that a great number of virtual circuits can be simulated [13].

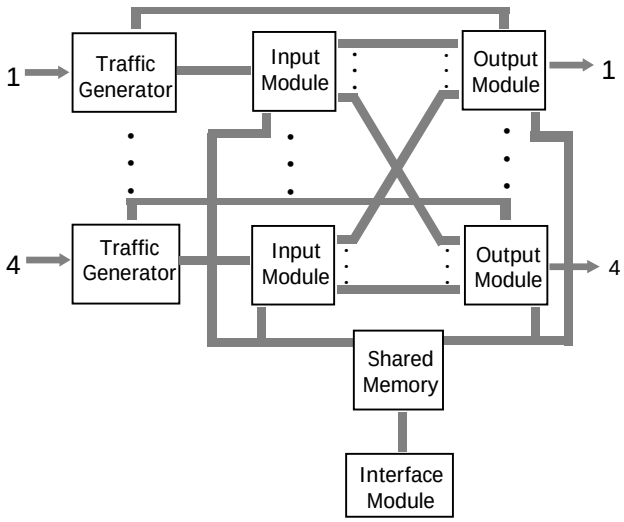

Figure 5 FAST generic board architecture

The FAST architecture is designed to facilitate the simulation of ATM switches with a broad range of internal architectures. The simulation process of the FAST system follows a classic digital design methodology [15]. VHDL is used as the hardware description language for development of the simulation models for FAST. A library of hardware models of several important simulation components such as FIFO memory controllers, priority list controllers, etc. So, to finally specify the simulation model, the user enters the switch design using VHDL and components of this library. All this VHDL code is first functionally simulated to verify the correct implementation of the algorithms in the hardware model. Finally, functionally correct VHDL code is synthesised into the target FPGA technology, using CAD tools.

To evaluate the utility and flexibility of the FAST system, two different ATM switches architectures (and the

associated scheduling algorithms) have been modelled, implemented in the FAST system and simulated under different traffic models. The first design is an ATM switch with output buffering, which uses a weighted round-robin scheduling algorithm. The second design is a distributed memory switch, implementing sorted-priority type scheduling algorithm.

The results obtained in the simulation of these two designs are explained below. To obtain the speedup offered by the FAST system over conventional sequential simulation techniques, the execution time (seconds) for both designs has been measured in the FAST system and in a simulator written in CSIM, running on a DEC Alpha 3000/400 workstation with 92 Mbytes.

The following table summarises the results obtained for the first design. Firstly, the great difference in time (seconds) between both simulation methods can be observed. It can be seen also, how the execution time increases with the number of virtual circuits with the simulation written in CSIM, and how this does not occur in the FAST system. For the second design, the results are in the same order of magnitude.

Number of VC	CSIM	FAST
4	410	3
8	434	3
16	470	3
32	540	3

Table 1 FAST System Results (seconds)

Finally we can conclude that the results that can be obtained corresponding to a network simulation using the FAST system could be even better compared to sequential simulation techniques, given that in sequential simulation techniques the simulation time will augment linearly with the number of switches. This affirmation is not true in the FAST system, where the interconnection topology is not important given that each switch is simulated in a different printed circuit board. Extrapolating these results a sequential simulation of 100 million cells in a network consisting of 5 switches in series with 32 VC would take more than 3 days, in comparison to 5 minutes on FAST.

4. Discussion

The use of sequential simulation techniques to execute the telecommunication networks simulation is highly inefficient, due to a large simulation time. It exists a practical necessity in accelerating these type of telecommunication networks simulation.

Parallel computing and reconfigurable computing can be used as acceleration techniques. However, parallel computing and reconfigurable computing based simulation techniques are not yet commonplace in network simulation. The main drawback of these acceleration techniques is that telecommunication engineers are not usually familiarised with them. It will be really helpful a simulation environment where acceleration techniques are transparent to telecommunication engineers.

The TeD language has been presented as a first practical approximation to this concept. TeD is an easy to use language and it uses few concepts in the modeling phase of telecommunication networks. It is also a language that is being used in several research institutions of recognised prestige where TeD models for telecommunication networks research are being developed.

Parallel computing based event-driven simulation has two main problems, the obtained speedup and the high cost of a parallel computer. The speedup obtained by parallelizing event-driven simulation may be small because their inherently serial nature [28]. Communication and synchronisation bottlenecks also limit the achievable speedup. High cost of a parallel computer also limits the use of parallel computing as acceleration technique in industrial simulation frameworks.

The FAST system has been presented as an example of a reconfigurable system applied to telecommunication networks simulation. This system has demonstrated with good results (speedup of 140 over sequential software simulation) the viability of this research line. Reconfigurable systems have been widely used in other applications (i.e. real-time vision) giving the same, or even better, results than parallel computers [27].

The FAST system has several problems, too. It is mainly designed for ATM switch and scheduling algorithms simulation. Only ATM networks of up to four switches can be simulated. A consequence of previous statement, is that other type of telecommunication networks cannot be simulated using the FAST system. The second main drawback of the FAST system is that the use of reconfigurable hardware is not transparent to telecommunication engineers, so knowledge of digital design methodology is needed. Finally, conclude that the FAST system is not really a simulation framework, it is a functional emulation system.

5. Our New Approach

Our approach to multimedia and telecommunication network simulations tries to overcome problems presented by existing simulation frameworks, and that have been summarised in the previous section. Taking, or try to combine, the benefits of explained environments (TeD and FAST) is an objective of our approach too.

In this section, a new simulation framework will be

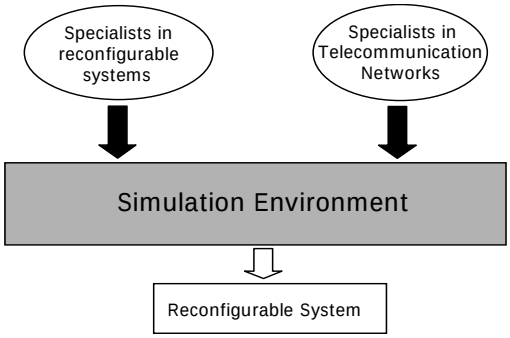

Figure 6 Reconfigurable Computing based Simulation Environment

proposed. Algorithms, methodologies and tools that we are planning to use will be explained, as a new approach to telecommunication network simulation.

Desired features in our new simulation framework are:

- Efficient simulation. Our simulation environment must be able to simulate long timescales and network configurations of really large size in an acceptable execution time (minutes or hours), as explained in the introduction.
- General purpose simulation framework. It must be capable to simulate actual telecommunication networks, as well as, future networks.
- Ease utilisation and a well defined working methodology must be desired, too. That is, the acceleration technique must transparent to the telecommunication user.
- Low cost. The cost of the system that we want to implement will be lower than the cost of a parallel computer. The cost of such system must be affordable by all telecommunication research community.

To achieve the desired features of low cost and efficient simulation we will use a reconfigurable computing based system. A network modeling language (i.e TeD or OPNET) will be used to obtain the feature of easy use. Finally, an automatic software tool will be the responsible for mapping the specified network to the hardware platform, achieving the desired transparency in the use of the reconfiguirable system. See Figure 6.

5.1. Simulation Algorithm

Our simulation algorithm is based on event-driven simulation paradigm. Basically, an event-driven simulator works on a producer-consumer manner. There are events to process which are saved in an ordered list (event memory), and simulation processes (P_i) which are responsible for the event processing. A simulation controller is required for assigning events to processes, save new events generated by processes and ordering the event list. See Figure 7.

The key point when planning to design and develop a network simulation framework is how to implement the mapping between network elements and simulation processes.

Mainly, two approaches are possible:

- One network element is mapped on a simulation process.
- Several network elements with the same behaviour (or of the same type) are mapped on a single simulation process.

Supposing the network shown in Figure 7, the one-to-one mapping option will require five simulation processes. The multiple-to-one option will require of three simulation processes (traffic generator, switch type A and switch type B).

We have chosen the second possibility (multiple-to-one) as the mapping strategy between networks elements and simulation processes, because this way will be possible to implement in the reconfigurable hardware a larger number of network elements, as is explained in the following subsection.

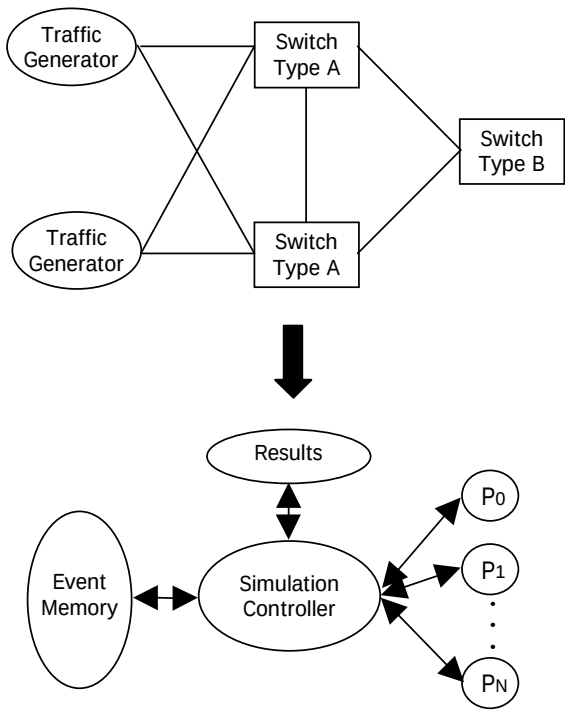

Figure 7 Network Elements to Simulation Processes Mapping

5.2. A HW/SW Codesign-based Approach

Recently, a new digital design methodology, known as Hardware/Software codesign [14][26], is subject of continuos research. Basically, this methodology works as follows: A specification of the application is done using a high level language, such specification enters to the HW/SW codesign tool which decides the parts of the specification must be mapped to HW[2] and the parts of the system that must be mapped to SW. This partitioning is the most important task in a HW/SW codesign methodology, and it is done on base to the requirements of the application (i.e. processing speed). Finally, conclude that this partitioning can be done manually or automatic.

An event-driven simulation algorithm, as explained in the previous subsection, is a perfect application for HW/SW codesign methodology. Our new network simulation framework will be based on a HW/SW codesign tool, as shown in Figure 8.

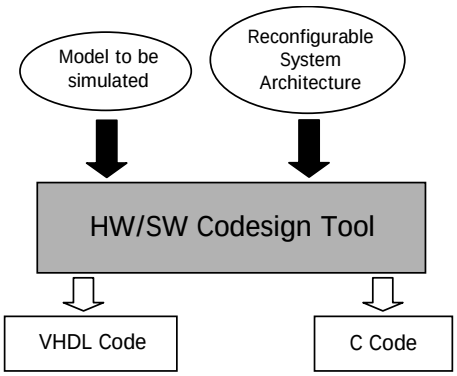

Figure 8 HW/SW Co-design based approach

The main tasks or functions of this HW/SW codesign tool are:

- Obtaining a list of all simulation processes from the network model which is specified using a high-level network modeling language.

- Realise the partition between processes that have to be mapped into the reconfigurable hardware and the processes that have to be mapped to software. This partition will be done is base to the reconfigurable system architecture and the execution time of each simulation process. The most time consuming simulation processes will be mapped into the reconfigurable hardware to achieve a better speedup in the simulation. This partitioning will be done automatically.

- VHDL code generation for the simulation processes that have to be implemented in hardware, which could be synthesised afterwards to the reconfigurable system using commercial tools. The idea is to map one simulation process into a single reconfigurable device.

- C code generation for the simulation processes that have to be implemented in software, which could be compiled and executed by any commercial microprocessor.

Finally, the reconfigurable system architecture will be implemented as a co-processor PCI board attached to a commercial PC. This PCI board will have a set of reconfigurable devices where to implement simulation processes. The simulation processes that have to be implemented in SW will be mapped into the PC microprocessor.

6. Conclusions

In this paper, a review of acceleration techniques for network simulation has been done. Such acceleration is required given that sequential simulation techniques are inadequate to address new telecommunication network simulation challenges. Two different types of techniques are possible: Parallel computing based simulation and Reconfigurable computing based simulation. Benefits and problems of such techniques has also been explained.

A new simulation framework has been proposed, which is focused on network simulation challenges which are not solved nowadays by existing simulation frameworks.

A HW/SW codesign approach and reconfigurable devices will be used, achieving an efficient simulation framework where reconfigurable hardware is transparent to telecommunications engineers. A mapping strategy between network elements and simulation processes has been explained. The main tasks or functions of the automatic HW/SW codesign tools has also been presented.

The architecture system architecture is being specified. A more accurate design of the codesign tool is subject of actual work.

The relation between reconfigurable systems and telecommunication networks has been explained, as a method to implement the “Networked Multimedia" concept. Thus, the objective is to use reconfigurable systems to simulate, prototype and evaluate telecommunication networks, which have to give the necessary QoS to the applications.

Figure 9 Use of Reconfigurable Systems as a way to achieve the Networked Multimedia Concept

[2] The hardware to be used in a HW/SW codesign methodology can be based on reconfigurable devices, but this possibility is not compulsory.

7. References

[1] "Next Generation Internet Implementation Plan". http://www.ngi.gov. Feb 1998.

[2] A. G. Fraser, "XUNET2: A Nationwide Testbed in High Speed Networking", Proc. of the INFOCOM. Florence, Italy. May 1992.

[3] A. M. Law, M. McComas, "Simulation Software for communications networks, the state of the art". IEEE communications Magazine, vol. 32, pp. 44-50. March 1994.

[4] A. Sangiovanni-Vincentelli, A. El Gamal, J. Rose, "Synthesis Methods for Field Programable Gate Arrays". Proc. of the IEEE, vol. 87, no. 7., pp. 1057-1083. July 1993.

[5] B. J. Premore, D. M. Nicol, "Parallel Simulation of TCP/IP using TeD", Proc. 1997 Winter Simulation Conference. December 1997.

[6] B. L. hutchings, M. J. Wirthlin, "Implementation approaches for reconfigurable logic applications", International Workshop on Field Programable Logic and Applications, pp. 293-302, Oxford, 1995.

[7] C. Papadopoulos et al., "A real-time testbed for prototyping Cell-based Comunication Networks" Proceedings of IEEE Workshop on Rapid System Prototyping, Grenoble. June 1994.

[8] D. D. Clark et al., "An Overview of the AURORA Gigabit Testbed", Proc. of the INFOCOM. Florence, Italy. May 1992.

[9] D. L. Perry, "VHDL". McGraw-Hill, 1994.

[10] D. R. Jefferson, "Virtual Time", ACM Transactions on Programming Languages and Systems, vol. 7, no. 3, pp. 404-425. July 1985.

[11] D. Rubenstein, J. Kurose, D. Towsley, "Optimistic Parallel Simulation of Reliable Multicast Protocols", ACM SIGMETRICS Perf. Eval. Rev., vol. 25, no. 4. March 1998.

[12] D. Stiliadis, A. Varma, "FAST: an FPGA-based simulation testbed for ATM networks", Proceedings of ICC'96, June 1996.

[13] D. Stiliadis, A. Varma, "Latency-rate servers: A general model for analysis of traffic scheduling algorithms", Proceedings of IEEE INFOCOM'96, pp. 111-119.

[14] G. Micheli, "Computer-Aided Hardware-Software Codesign", IEEE Micro, pp. 10-16, August 1994.

[15] G. Micheli, "Synthesis and Optimization of Digital Circuits". McGraw-Hill Series in Electrical and Computer Engineering, 1994.

[16] H. Zhang, "Service disciplines for guaranteed performance service in packet-switching networks", Proceedings of the IEEE, vol. 83, pp. 1374-96, October 1995.

[17] J. M. Arnold, D. A. Buell, E. G. Davis, "Splash 2", Proceedings of the 4'th Annual ACM Symposium on Parallel Algorithms and Architectures, pp. 316-324, June 1992.

[18] J. S. Steinman, "SPEEDES: Synchronous parallel environment for emulation and discrete event simulation". Advances in Parallel and Distributed Simulation", vol. 23, pp 95-103. January 1991.

[19] K. Hayashi et al., "A novel approach to real-time verification of transport system design using FPGA based emulator". Proceedings of IEEE Workshop on Rapind System Prototyping, June 1996.

[20] K. S. Perumalla, A. T. Ogielski, R. M. Fujimoto, "MetaTeD – A Meta Language for Modeling Telecommunication Networks". Technical Report GIT-CC-96-33, College of Computing, Georgia Institute of Technology, 1996.

[21] K. S. Perumalla, R. M. Fujimoto, "A C++ Instcance of TeD". Technical Report GIT-CC-96-33, College of Computing, Georgia Institute of Technology, 1996.

[22] M. C. Chan, G. Pacifici, R. Stadler, "Prototyping Network Architectures on a Supercomputer". IBM Research report RC-20438. New Jersey, 1996.

[23] MIL3, Inc., "OPNET Tutorial Manual" Version 2. Software Release: 3.1. 1989-1997

[24] N. Wirth, "Hardware Compilation: Translating Programs into Circuits", IEEE Computer, vol. 31, no. 6, pp 25-31. June 1998.

[25] O. T. Albaharma, P. Y. K. Cheung, T. J. Clarke, "On the Viability of FPGA based Integrated Coprocessors", Proc. of the 1996 IEEE Symposium on FPGA's for Custom Computing Machines.

[26] P. Athanas, H. F. Silverman, "Processor Reconfiguration Through Instruction-Set Metamorphosis" IEEE Computer, 26(3):11-18. March 1993.

[27] P. Bertin, D. Roncin, J. Vuillemin, "Programable Active Memories: A performance Assesment". Prl report, DEC Paris Research Laboratory, 85, Av. Victor Hugo, 92563 Rueil-Malmaison, Cedex (France). June 1992.

[28] R. Fujimoto, "Parallel discrete-event simulation", Comunications of the ACM vol 33, no. 10, pp. 30-53, October 1990.

[29] S. Bhatt, R. Fujimoto, A. Ogielski, K. Perumalla, "Parallel Simulation Techniques for Large-Scale Networks". IEEE Comunications Magazine, pp. 42-47. August 1998.

[30] S. Brown, J. Rose, "FPGA and CPLD Architectures: A tutorial", Proceedings of the IEEE Design & Test of Computers, vol. 13, no. 2, pp. 42-57, 1996.

[31] S. Guccione, List of FPGA-based Computing Machines (on-line). Web page, URL: http://www.io.com/~guccione/HW_list.html.

[32] S. Monaghan, P. Noakes, "Reconfigurable special-purpose hardware for scientific computation and simulation". Computer & Control Engineering Journal 3, 5. pp 225-234. September 1992.

[33] T. Matsumura et al. "Real-time emulation for ATM switching systems in Broadband ISDN". Proceedings of IEEE Workshop on Rapid System Prototyping, June 1996.

[34] W. H. Mangione-Smith, B. L. Hutchings, "Configurable Computing: The Road Ahead", Reconfigurable Architectures Workshop (RAW'97). Geneva, 1997.