HIGH-SPEED LINK SIMULATION USING X PARAMETERS, VOLTERRA SERIES AND THE LATENCY INSERTION METHOD (LIM)

Allocation: Illinois/50 Knh
PI: Jose Schutt-Aine[1]
Co-PI: Raj Mittra[2]

[1]University of Illinois
[2]The Pennsylvania State University

EXECUTIVE SUMMARY

In today's high-speed communication era, humongous data throughput is being routed through communication channels. Circuit design and analysis techniques have become more challenging and will remain a critical component in ensuring the success and viability in the design of short-range high-speed input/output (I/O) links. Because of the large amount of data, the simulation of such systems is a daunting task. Present systems require bit error rates (BER) in the order of 10-15. A deterministic simulation of such a large number of bits would require several thousand years using a modern-day computer. We used the Blue Waters computational platform to explore the feasibility of simulating high-speed serial links. By properly combining the computational powers of the Blue Waters system with new algorithms developed in our research group, we reduced the computational time by several orders of magnitude and made the design cycle for high-speed links more manageable.

INTRODUCTION

In today's high-speed communication era, bandwidth is increasingly becoming the major limiting factor in channel performance. In order to keep up with the humongous data throughput needed, high edge rates are routed through communication channels consisting of complex arrangements of high-density interconnects, packages, and integrated circuits. As short-range data rates keep increasing to multi gigabits per second (Gbps), interactions between channel links, integrated circuits, packages, and power distribution networks take place and become more difficult to predict and manage. Noise sources such as crosstalk and jitter limit the performance of these systems and force the data rates to be well below the Shannon limit of channel capacity. Circuit design and analysis techniques have become more challenging and will remain a critical component in ensuring the success and viability in the design of short-range high-speed input/output (I/O) links.

To circumvent the performance limitation and voltage scaling problem posed by the conventional parallel buses, the industry has migrated from the multi-drop parallel buses to point-to-point serial buses. In a serial bus, a device called SerDes (Serializer/Deserializer) is used to transmit and receive data over the serial link. The SerDes can be either a stand-alone device or, in most cases, an intellectual property (IP) core integrated into a serial bus controller or an ASIC. The timing skew problem encountered in a parallel bus is solved by embedding the clock signal into the data stream. Since there is no separate clock signal in a serial bus, timing skew between clock and data (which, together with the minimum setup and hold time, determines the maximum data transfer rate) no longer exists. Consequently, a serial bus operates at a much higher data rate than its counterpart in a parallel configuration.

The design of a multi-gigabit SerDes can be challenging because of the high-speed, mixed-signal circuitry involved, as well as stringent electrical specifications. Robust design of these systems requires access to reliable modeling and simulation tools that can help predict the performance and optimize the design of these systems. Simulation techniques for the prediction of signal propagation in these environments have become a subject of increased interest over the past few years. Behavioral and macro-modeling techniques have been among the most popular methods used to predict the performance of these systems.

METHODS & RESULTS

The standard tool for circuit simulation is SPICE. However, for very large networks, the computational time becomes prohibitively large. Our research group introduced the latency insertion method (LIM) for time-domain simulation of large networks [1]. The method is based on a finite difference time domain (FDTD) formulation of the branch and node equations and obtains the solution in a leap-frogging method similar to the Yee algorithm [2] used in electromagnetics. No matrix inversion is performed and the method has linear numerical complexity.

The advantage of the LIM method rests in the fact that it is significantly faster than SPICE. Moreover, this speedup advantage increases as the number of nodes in the network increases. Thus LIM is a time-domain formulation that leads to the generation of update algorithms for the simulation of networks. This algorithm exhibits linear computational complexity; in addition, because of the time domain nature of the formulation, it can handle nonlinearities.

The latency insertion method has proven to be a viable circuit simulation approach that is a good alternative to SPICE. However, transistor models for a LIM-based simulator have yet to be developed. Thus far, the models have neglected the nonlinear charge storage effects which may be critical in short-channel technologies. Given that the LIM algorithm has been found to be efficient in simulating large transistor circuits, it is important that the basic model used be very accurate. In our work, a dynamic model for short-channel MOSFET transistors was developed for LIM-based simulation. The model incorporates the nonlinear charge storage effects. A modular representation is adopted for the MOSFET, which leads to general branch and node update equations. A dynamic LIM-based model was developed for an NMOS transistor which led to the associated update equations. A simulation program was implemented. Results are examined and comparisons are made with existing techniques.

An approach for the transient simulation of circuits through the latency insertion model using advanced models for MOS transistors was implemented. By taking into account the dynamic charge storage effects in short-channel devices a more accurate simulation of high-speed digital and analog circuits via the latency insertion method was performed. The approach makes use of the SPICE LEVEL 3 transistor model for MOSFETs. The use of the latency insertion method allows better convergence and higher computational speed for the simulation. Computer simulations showed improvement in accuracy by using the high-level models. However, when the number of transistors becomes large, the computational simulation becomes very slow, which warrants the use of a platform such as Blue Waters.

In recent years, macromodeling has received increasing attention due to the ever-growing complexity of networks used for computer and communications applications. In particular, blackbox macromodeling has become an attractive approach since it allows us to bypass the complex topologies of networks through the use of behavioral descriptions at the ports and terminals of these systems. S-parameter blackbox macromodeling has become a convenient vehicle to exchange behavioral data pertaining to packages, connectors, channels used in high-speed links and other types of communication networks. As an example, in signal integrity applications, it is often required to perform the simulation of systems for which equivalent circuits are not available. This occurs in instances where the data from the system are obtained from measurements or from a field solver. In these cases, only the network port parameters are known as a function of frequency and are used to represent the network as a macromodel. The final objective is to use the exchanged behavioral model in conjunction with a circuit simulator such as SPICE. In order for such a simulation to become possible, two key steps must be performed.

First, a model-order reduction process must be carried out to extract the multiport poles and residues of the network. The poles and residue contain the "character" of the system and may describe its behavior over a wide range of frequencies. Today, the most commonly used pole/residue extraction method is the vector fitting method which has proven to be robust and reliable. Once the poles and residues of a system are obtained, a simulation of the network can be performed using time-domain simulation techniques such as recursive convolution. In many cases however, a circuit description is often desired in order to be combined with external circuitry that may contain nonlinear elements. The synthesis of such circuits has been explored in the literature using various techniques. In particular, the pole-residue representation can be used to perform a circuit synthesis of the macromodel in which a SPICE-compatible netlist is generated and can be used for simulation. In our effort, we developed a synthesis technique for the implementation of equivalent circuits based on scattering parameter representation of linear broadband networks. Several circuit topologies for complex pole residue pairs were defined and tested and found to accurately describe the behavior of the blackbox networks. When the number of ports becomes large, the computational efficiency of the method is very limited, which warrants the use of a platform such as Blue Waters.

WHY BLUE WATERS

We plan to use the Blue Waters computational platform to explore the feasibility in simulating high-speed serial links that would necessitate several thousand years using standard algorithms. By properly combining the computational powers of the Blue Waters system with new simulation algorithms developed in our research group, we plan to reduce the computational time by several orders of magnitude. More specifically, our research will include the following tasks: use LIM on Blue Waters to make transistor-level simulations involving a large numbers of devices (this would not be feasible without Blue Waters); develop the circuit synthesis of blackbox macromodels with large numbers of ports; develop techniques to parse, analyze, and process the big data generated by measurement or calculation of X parameters; and process the generated data to develop deterministic and stochastic simulation methods for analog blocks using Volterra series and the latency insertion method (LIM) on the Blue Waters platform.

NEXT GENERATION WORK

In the 2019-2020 time frame, we plan to demonstrate the simulation of integrated circuits with large numbers of transistors as well as high-speed links.

HARDWARE ACCELERATION OF DEEP LEARNING

Allocation: Illinois/50.0 Knh
PI: Tao Xie[1]
Co-PI: Yuan Xie[2]

[1]University of Illinois at Urbana-Champaign
[2]University of California at Santa Barbara

EXECUTIVE SUMMARY

Our project aims to use Blue Waters for hardware acceleration of deep learning for big data image analytics. To achieve near real-time learning, efforts are required for hardware scaling out (increasing the number of compute nodes in a cluster) and scaling up (improving the throughput of a single node by inserting hardware accelerators). We evaluated the performance of scaling up using the graphics processing unit (GPU) enabled node XK7 for training convolutional neural networks. Our key observation thus far is that implicit data synchronization across different nodes severely slows down the training process. We propose a data manager that explicitly covers the data transfer overhead with computation. Our first step was to test the proposed strategy on a single XK7 node of Blue Waters, and results show a speedup of 1.6 times that of the implicit data transfer implementation.

INTRODUCTION

Deep learning has been used in applications such as image classification, speech processing, and object recognition. A very large amount of training data is necessary for deep neural networks, therefore this work requires computing power capable of matching the advanced state-of-the-art accuracy of these tasks. Mainstream deep learning facilities are central processing unit (CPU)-based clusters, which usually consist of thousands of compute nodes. As the major computation of deep learning is convolution and matrix multiplication, which are suitable for GPUs to process, most modern deep learning facilities are equipped with GPUs as hardware accelerators. However, straightforward implementation of deep neural networks on GPU-enabled compute nodes will lead to under-utilization of computing resources. In this work, we evaluated the performance of a convolution neural network on a GPU-enabled cluster (XK7 nodes on Blue Water). We observed that the performance bottleneck of the convolutional neural network training is the implicit data synchronization across different nodes, which is used to confirm the correctness of the output of each layer in the neural network that is distributed in different nodes. To alleviate the performance degradation caused by the synchronization, we propose a method to hide the data transfer explicitly by computation. This method dramatically increases the utilization of the compute units on each GPU chip and thus improves the overall training performance.

METHODS & RESULTS

Traditionally distributed convolutional neural networks allocate one compute node for each replica as part of the training model. The training process of each layer in each replica consists of three phases: 1) receiving output data from the previous layer of other replicas and feeding them into the device memory; 2) executing the device kernel to compute the output data of current layer; and 3) sending the output data of the current layer to other replicas.

The performance bottleneck of this straightforward implementation is that steps 1 and 3 introduce very long latency, which is determined by the longest unpredictable network latency between different compute nodes. The reason for the performance loss is under-utilization of hardware resources of the GPU accelerators. In Steps 1 and 3, compute units are completely idle while waiting for data synchronization. In Step 2, the direct memory access units are inactive while the compute units are executing the kernel functions. Therefore, overlapping the data synchronization and the kernel execution will reduce the total running time of the training process.

To combat performance loss due to the under-utilization of hardware resources, we break down the replicas into finer grained replicas with the same number of compute nodes used. This way, the kernel execution of one replica can run simultaneously with data synchronization of other replicas. Ideally, if the replica break-down does not introduce any communication overhead, there will be more replicas in each node and better performance will be achieved. However, the communication overhead is not negligible, so we have to find the best number of replicas per node. We find the best ratio is two replicas per node, and the speedup of this configuration is 1.6 times over that of straightforward implementation.

WHY BLUE WATERS

Blue Waters offers XK7 nodes which consist of one AMD 8 floating point core CPU and one NVIDIA K20X GPU. As GPUs are more suitable than CPUs for convolution and matrix multiplications, state-of-the-art deep learning facilities widely employ GPUs as their hardware accelerators. Blue Waters offers an opportunity to perform research on equipment optimized for deep learning cluster with GPUs. Also, Blue Waters provides a CUDA API programming environment, which enables us to customize the specific functions to be executed on GPUs.

NEXT GENERATION WORK

A field-programmable gate array (FPGA) is an alternative hardware accelerator to GPUs. It can be quickly used to prototype new hardware designs with very high power efficiency. In the next generation work plan could employ a FPGA to accelerate deep learning algorithms.