PCNN Neurocomputers – Event Driven and Parallel Architectures

Cyprian Grassmann[1], Tim Schoenauer[2], Carsten Wolff[3]

Infineon Technologies AG, CPR ST[1], WS TI[3]
{cyprian.grassmann, carsten.wolff}@infineon.com
Multilink Technology GmbH[2]
tim.schoenauer@multilink.de

Abstract : The simulation of large spiking neural networks (PCNN) especially for vision purposes is limited by the computing power of general purpose computer systems [5,9,10]. Therefore, the simulation of real world scenarios requires dedicated simulator systems. This article presents architectures of software and hardware implementations for PCNN simulator systems. The implementations are based on a common event driven approach using spike events for communication and processing flow. Furthermore, parallel approaches utilizing the spike event computing are introduced for simulation acceleration. Implementations of software simulators on workstation clusters and parallel computers and hardware accelerators based on FPGAs, ASICs and DSPs are described. The presented results demonstrate the capability to simulate large vision networks close to real world/real time requirements.

1. Introduction

The examination of large vision PCNNs is of major interest for both the better understanding of brain function and the inspiration for computer vision. The simulation performance of standard simulation systems is far below the requirements [5,9,10]. Thus special approaches are required. The basic paradigm for all presented algorithms is the event driven simulation which benefits from the utilization of rather seldom spike events. Chapter 2 describes the basic event driven algorithms and their implementation within the software simulator of SPIKELAB [3]. Results of the simulator are compared to GENESIS [1] runs of similar networks. In Chapter 3 parallel simulation techniques and a parallel PVM simulator are described. The speedup of simulation is shown by typical example networks of large vision PCNNs [10,13] on SUN workstation clusters and a massively parallel Pentium III cluster. Chapter 4 introduces neurocomputer architectures based on both the event driven and the parallel algorithms. SPIKE128k [1] is a FPGA based single processor architecture, ParSPIKE [14] is the extension to a parallel DSP design. NESPINN[5] and the successor system MASPINN [10] are ASIC architectures. As part of MASPINN, a neuroprocessor ASIC, the NeuroPipe chip [11] has been fabricated. Chapter 5 summarizes the results.

2. Event Driven PCNN Simulation

Spikes in a pulsed coupled neural network are rare and the connections between the neurons of a network are sparse. Therefore the activity in these kind of networks is very low. Compared to the number of neurons in the network, just a fraction of them is usually active. Hence modeling the spikes as events is straight forward but the corresponding neuron models are getting more complex. This is due to the fact, that each neuron model must predict its future behavior, with respect to the incoming and outgoing spikes. Each time an incoming spike arrives at its inputs it must predict if it will emit an outgoing spike or not. Contrary in time slice simulations the model may be kept much simpler, because only the changes from one time slice to the other have to be calculated and serve as basis for the next time slices. Therefore these models are often easier to implement – with the sake of performance degradation. The same is especially true for learning algorithms, which are often described by an iterative algorithm. Spending the effort in event-driven models returns faster simulations or enables one to simulate larger networks in an acceptable amount of time. As an example we compared the event driven simulation of SPIKELAB to the time sliced simulation of GENESIS. We used a rather simple model, which is related to the spike response model [7]. The postsynaptic potential (PSP) follows the equation

$$PSP = \frac{t - t_S}{t} \cdot e^{-\frac{t-t_S}{t}}$$

and is weighted by the individual weight of the corresponding synapse. Choosing 2.7ms for τ results in a PSP, which is close to measured PSPs. An outgoing spike is emitted, when the sum of all incoming PSPs crosses a fixed threshold. A network with five fully connected layers of these neurons was built up and fed by a spike train neuron, which generates spikes in a predefined interval. The delay between the neurons was randomly chosen between one and three milliseconds. Figure 1 shows the simulation time relative to the size of a time slice (left side) and relative to the network size for different levels of activity (right side). Although the PSP is implemented as a lookup table in GENESIS, the Performance of SPIKELAB is much higher. Moreover the performance of GENESIS degrades much faster for higher resolutions. As one can expect the simulation time of GENESIS is nearly independent from the network activity, but SPIKELAB clearly profits from lower activities.

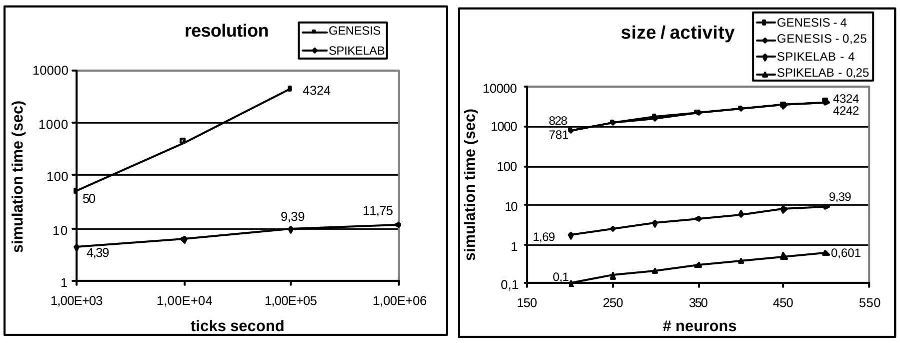

Fig. 1. Comparison of SPIKELAB and GENESIS

Therefore the event driven simulation of SPIKELAB is especially well suited for the fast and precise analysis of the temporal aspects, like synchronization in pulsed coupled neural networks. Larger networks may be accelerated by a parallel simulation like described in the following chapter and in [2].

3. Parallel PCNN Simulation

Common ANNs show very poor speedup for simulation on parallel computers because of their high communication volume [8]. PCNNs consist of complex model neurons and need lower communication effort due to the information exchange via rather seldom spike events. Due to the inherent dynamic structure of the networks the load and communication balancing is a very complex task. As a solution we use the event driven approch considering only active neurons and additionally explore the specific structure of the vision networks for appropriate network partitioning. The presented parallel simulation algorithm is based on a farmer-worker simulation. The neurons are placed on the workers by distribution of the data sets describing their behavior. For each presynaptic neuron all connections to its successors together with the connection weights are provided in an adjacency list called stimulation information block (SIB). Spikes emitted by a neuron are called PreSpikes, received spikes (distributed via SIB) are called PostSpikes. For our simulations we use two algorithms. In the *distributed* version the SIBs are split and distributed over all workers. On the other hand, the *centralized* approach dispenses only the neurons and all SIBs reside completely in the local memory of the farmer. The distributed simulation is more suitable for a message passing machine while the centralized simulation is more appropriate for a shared memory architecture.

Distributed Simulation The distributed algorithm collects the processed PreSpikes in local spike lists which are exchanged between workers. The SIBs are split and distributed. Originally, the SIBs are stored in a sender oriented way such that for all presynaptic neurons the connections to postsynaptic successor neurons are in one SIB. In a parallel simulation, this would lead to heavy traffic because the PreSpike of this neuron has to be transformed into many PostSpikes which must be send to other workers. Because one worker host has many postsynaptic neurons, the usage of PreSpikes for communication reduces traffic substantially. Hence, SIBs are split and distributed to the workers with postsynaptic targets once before simulation start.

Centralized Simulation The centralized algorithm stores all the SIBs in the local memory of the farmer. All workers transmit their PreSpike to the farmer. The farmer determines the SIBs and propagates the connections to the corresponding workers. Hence, the farmer splits the SIBs and collects the processed PostSpikes in packets for the corresponding targets. The worker uses a SIB cache with a simple cache strategy which benefits from two simulation characteristics. Firstly, the activity in vision PCNNs is quite local and slowly moving leading to a high reuse of cached SIBs. Secondly, many neurons are stored on one worker and the neurons are mainly connected to their neighbors leading to many local connections.

Both the distributed and the centralized simulator are implemented using the PVM (Parallel Virtual Machine) environment with Sun Solaris. The neurons are distributed once at the start of simulation, partitioning of the network is done in before within a special compiler [8]. For simulation we used a Sun Sparc 5 Workstationcluster with up to 12 nodes and a Fujitsu Siemens hpcline Pentium III (450 MHz) cluster with up to 24 nodes (details in [8], simulation time in simulated ms per virtual 1 ms timeslot).

Number of nodes	**1**	**2**	**4**	**8**	**12**	**16**	**24**
Weitzel net [13], sparc 5, distributed	370	190	110	85			
Weitzel net [13], sparc 5, centralized	490	270	160	90			
Brause net [10], sparc 5, distributed			960	510	370		
Weitzel net [13], hpcline, distributed		19	9.5	6.5		7	
Brause net [10], hpcline, distributed		263	129	63	41		28

Tab. 1: Processing time (ms) per virtual timeslot (1ms) on different simulator systems

4. Neurocomputer Architectures

4.1. SPIKE128k and ParSPIKE

SPIKE128k [1] is a neurocomputer based on spike event processing with one dedicated hardware pipeline and a single sender oriented SIB memory. The system is composed of several VME type boards with programmable logic (FPGAs, CPLDs), memories and dedicated arithmetic components. The system can be connected to other SPIKE128k systems and to host computers via transputer links. The SPIKE128k hosts a dedicated learning module [9]. Processing speed for the Weitzel net can be found in [4].

ParSPIKE [14] is a concept for a successor system based on a parallel DSP approach. Analog Devices SHARC DSPs with large on chip memory are used as complete processing nodes. These nodes are composed with a spike switching circuitry and a dedicated SIB memory controller on VME boards forming a system either for the centralized or the distributed simulation algorithm. Estimations with a 2-DSP prototype and simulation results from the PVM simulator can be found in [4].

4.2. NESPINN and MASPINN

Both PCNN-accelerators, NESPINN [5] and MASPINN [10], make use of ASICs to combine high processing speed with a customized architecture. The systems are based on accelerator-boards connected to a host computer. Like the SPIKE128K, NESPINN- and MASPINN-boards consist of a spike event list with a sender oriented SIB memory and a processor-ASIC which computes a configurable neuron model. In contrast to the SPIKE128K, the NESPINN and MASPINN are not limited to a fixed neuron model, but allow the configuration of neuron models with up to 16 dendrite potentials (DPs) with different functionality (e. g. inhibitory, excitatory, multiplicative). To increase the processing speed, in both systems active DPs are tagged and only the active ones are processed by the neuroprocessor-chip.
A block diagram of a MASPINN-board is depicted in Fig2 (left). In Fig2 (upper right) the architecture of the neuroprocessor-ASIC of the MASPINN-system, the NeuroPipe-Chip, is shown. The MASPINN-system – as the successor of the NESPINN-system – includes some additional architectural concepts in order to gain about another order of magnitude in speed. On board-level, weight caches have been introduced. They allow a further parallelization of the processing steps necessary for the simulation of a PCNN. Furthermore, a compressed DP-Memory, relaxes the bandwidth requirements of the neuron-processor-chip.

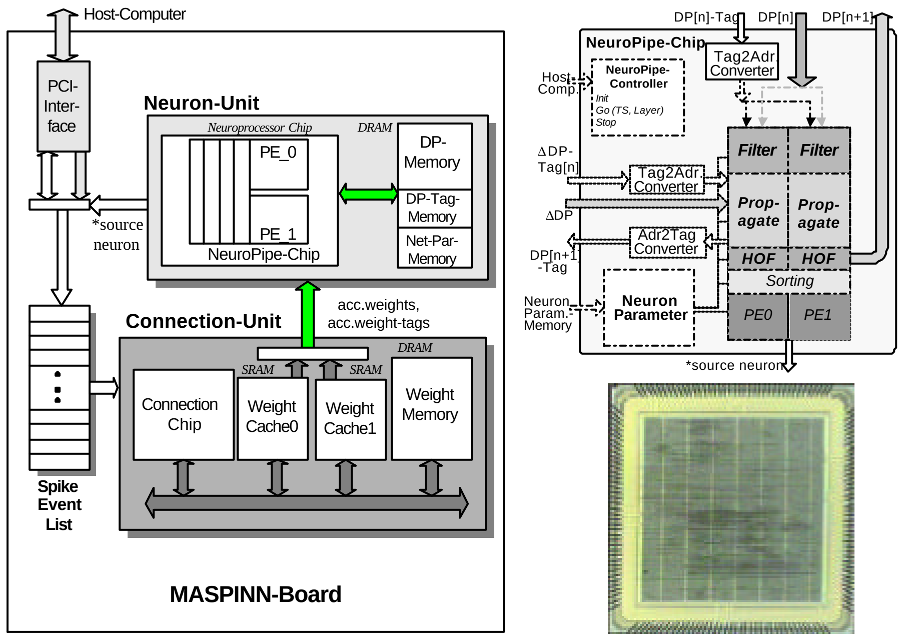

Fig. 2. MASPINN with the NeuroPipe-Chip

Pre-analysis tests the relevance of each DP in the context of the other DPs of the corresponding neuron. This reduces the computational load during the computation of the membrane potential in the pipeline of the NeuroPipe-Chip. Also on chip-level, an on-chip inhibition unit in the NeuroPipe-Chip may emulate the inhibition of the entire network or large parts of it. Since all parameters related to inhibition and its computation are hosted on-chip, the bandwidth and computational requirements of the entire system are also reduced.
A prototype of the NeuroPipe-Chip with two parallel processing elements (PEs) has been fabricated in a 0.35um-digital-CMOS technology (see chip photo in Fig2-lower right).

5. Results and Conclusion

The main conclusion of this article is that event driven algorithms are the key to high speed PCNN processing. Based on these algorithms parallel approaches and dedicated hardware architectures can deliver almost any required simulation performance. The presented implementations and concepts pursue different goals and therefore are not comparable solely on the basis of simulation speed. Neurocomputers based on dedicated neuroprocessor-ASICs like NESPINN and MASPINN certainly offer maximum simulation speed. Concerning flexibility, handling and the ease of implementation a software solution like the SPIKELAB simulator is superior to dedicated hardware. However the simulation performance for real scenarios is still insufficient. Even a parallel implementation, e.g. on the basis of PVM, achieves the required performance only on large high performance computers like hpcline. The ParSPIKE concept combines commercial hardware with dedicated communication hardware to overcome the typical bottleneck in parallel processing. Between the extremes of the ParSPIKE and the MASPINN system there is the SPIKE128k. The application of the presented systems and architectures pursues two aspects. On the one hand PCNNs should be developed and examined and on the other hand PCNNs should be applied to real-world tasks. For the development of PCNNs the required simulation performance is less crucial while there is a strong demand for flexibility and handling. Suitable simulators for such a task are software simulators. For applications of real world tasks on the other hand, systems like NESPINN and MASPINN present a superior platform for the simulation of PCNNs.

References

1. Bower J.M., Beeman D. - The Book of GENESIS. Springer Verlag, (1994)
2. Grassmann C., Anlauf J.K. - Distributed, Event Driven Simulation of Spiking Neural Networks. Neural Computation 98: 100-105, (1998)
3. Grassmann C., Anlauf J.K. - Fast Digital Simulation Of Spiking Neural Networks And Neuromorphic Integration With Spikelab. International Journal of Neural Systems, Vol. 9, No. 5 (1999)
4. G. Hartmann, G. Frank, M. Schäfer, C. Wolff: SPIKE128K - An Accelerator for Dynamic Simulation of Large Pulse-Coded Networks. MicroNeuro 97, (1997)
5. A. Jahnke, U. Roth, H. Klar, “A SIMD/Dataflow Architecture for a Neurocomputer for Spike-Processing Neural Networks (NESPINN)”, MicroNeuro'96, p. 232-237, (1996).
6. A. Jahnke, T. Schönauer, U. Roth, K. Mohraz, H. Klar: Simulation of Spiking Neural Networks on Different Hardware Platforms. ICANN97. Springer Verlag, Berlin, pp.1187-1192 (1997)
7. Maass, Wolfgang, Bishop, Christopher M. - Pulsed Neural Networks. (1998)
8. R. Preis, K. Salzwedel, C. Wolff, G. Hartmann: Efficient Parallel Simulation of Pulse-Coded Neural Networks (PCNN). PDPTA2001, Las Vegas (2001)
9. M. Schäfer, T. Schoenauer, C. Wolff, G. Hartmann, H. Klar, U. Rückert: Simulation of Spiking Neural Networks - Architectures and Implementations. to appear in NeuroComputing, Elsevier Science BV
10. T. Schoenauer, N. Mehrtash, A. Jahnke, H. Klar, “MASPINN: Novel Concepts for a Neuro-Accelerator for Spiking Neural Networks”, VIDYNN'98, (1998).
11. T. Schoenauer, S. Atasoy, N. Mehrtash, H. Klar, “Simulation of a Digital Neuro-Chip for Spiking Neural Networks”, Proceedings of the IEEE International Joint Conference on Neural Networks, IJCNN2000, Como, Italy, July (2000).
12. J. Thiem, C. Wolff, G. Hartmann: Biology-Inspired Early Vision System for a Spike Processing Neurocomputer. BMCV2000, Seoul (Korea) 2000, Lecture Notes in Computer Science 1811, Springer, pp. 387-396
13. L. Weitzel, K. Kopecz, C. Spengler, R. Eckhorn, H. J. Reitböck: Contour Segmentation with Recurrent Neural Networks of Pulse-Coding Neurons. CAIP'97, Kiel (1997)
14. C. Wolff, G. Hartmann, U. Rückert: ParSPIKE - A Parallel DSP-Accelerator for Dynamic Simulation of Large Spiking Neural Networks. MicroNeuro 99, Granada, pp. 324-331 (1999)