A Behavioral Modeling Concept and Practice of CNN-UM VLSI Implementations

A BEHAVIORAL MODELING CONCEPT AND PRACTICE OF CNN-UM VLSI IMPLEMENTATIONS

Péter Földesy and Ang el Rodríquez-Vázquez

Abstract-- **In this paper we introduce a novel simulation time bounded behavioral modeling technique, that optimally selects the incorporated block models. The method has been specially developed for fast performance evaluation of large mixed-signal image processing arrays. The time domain accuracy is optimized under the simulation time constraint by automatic selection of various user supplied block models. A dedicated environment also has been developed for efficient numerical simulations. Utilizing the proposed methodology, a bridge has been built for the CNN-UM VLSI implementations between the device level and the high-level functionality.**

Index Terms-- **Behavioral modeling, cellular neural network universal machine, VLSI, mixed-signal, large, heuristic selection, non-linear**

I. INTRODUCTION

It is well-known that the numerical and symbolic circuit analysis are ways to connect the system behavior to the features of single components. This connection allows to perform error and tolerance analysis, model generation, hence overall numeric simulation, circuit sizing, optimization, and finally, automatic circuit synthesis [17]. Meanwhile for a wide range of standard mixed-signal circuits several analysis and synthesis tools and methods are available, for the mixed-signal Cellular Neural Network [1] implementations it is not so due to practical reasons.

From the design point of view, the integration level of the CNN Universal Machine [2] chips have almost reached [6], [11] and will reach soon the milestone of 1 million transistors working mostly in analog region. From the other hand, a complete performance evaluation definitely should incorporate optimization of some 30 different nonlinear spatio-temporal transients controlled by dozens of free parameters. Due to the high integration level with inherent mixed-signal behavior the electrical simulations (e.g. HSPICE) of such tremendous task require computing power in the range of hundreds TFlops.

Naturally, several high-level analytic and numeric methods have been published to accomplish the draft performance evaluation of a given architecture [6]-[10], but they cannot handle more than some idealized second-order device effects. The supposed and applied simplifications hold only roughly for real physical devices and shade the most significant design specific phenomena. Up to date only two projects are known to deal with CNN-UM chip behavioral level modeling [3]-[4]. The first work [3] bases on CADENCE Design Framework II using Verilog HDL (digital) and SpectreHDL (analog) description applied for a specific design. The drawback of this approach is the remaining high computational requirements and "only" months of simulation time. The second approach [4] is a general CAD taking the advances of the regularity and local connectivity of the CNN, meanwhile leaving the most important question open: the model building.

In order to thinner this deep gap between the device and the functional level, a behavioral model simplification technique and a dedicated simulator environment (called <u>B</u>ehavioral <u>L</u>evel <u>CNN</u> BLCNN simulator) have been developed. Meanwhile, the task of block model development remains in the hand of the user, their complexity – roughly the number of terms and functions - is fitted automatically. A case-study will serve as illustration of the proposed methodology applied for the ACE4k chip [11]. For which, the above mentioned optimization task running time could be reduced to hours.

This paper is organized as follows. In Section II the brief review of the CNN-UM architecture is presented. In Section III the introduction of the model generation process can be found. In Section IV we then review some properties of the BLCNN simulator. In Section V the simulation results of the case-study can be found. Finally we summarize our major findings.

II. THE CELLULAR NEURAL NETWORKS

Cellular Neural Network is defined as a multidimensional array computing architecture on continuous signals, where the nonlinear dynamic elementary processors, the cells placed in the grid points of the array, are mainly locally connected within a finite neighborhood both with feed-forward and feedback programmable weights. The CNN Universal Machine [1] was introduced as a stored programmed computer, with a CNN array embedded. The additional extensions are: local continuous (analog) and logic memory, local analog and logic units as well as a global programming circuitry. Hence, continuous valued spatio-temporal dynamics is embedded in a logic structure, both locally and globally.

The design of such a large array sized VLSI implementation is quite a sophisticated task [5]. Moreover, its the performance estimation and circuit analysis against the parameter deviation and noise is an extremely time consuming process. The only fact, which gives possibility to do the job is the well-structured architecture composed of relatively simple blocks. The difficulty rises from the unknown impact of model complexity and the remaining long simulation time even in case of a simple model.

III. THE MODELING TECHNIQUE

The main goal of our intentions was including, as much as possible, second-order physical effects into the performance analysis by means of circuit modeling. On the contrary to the output error criterion driven circuit simplification modeling techniques [17]-[18], in our case the simulation time is the main constraint (besides the fact, that the referred automatic techniques cannot be applied for such huge multiple input–multiple output circuits). In order

Manuscript received __________________

The authors are with Instituto de Microelectrónica de Sevilla - CNM-CSIC, Edificio CICA-CNM, C/Tarfia s/n, 41012 Sevilla SPAIN (e-mail: peter@imse.cnm.es)

to increase the calculable model complexity, the most powerful simulation technique has been chosen: building a costume-made dedicated simulator. This environment and a meaningful simulation time *ST* (some minutes per single operation) result an upper limit on the model complexity: an upper bound of the number of involved terms and equations (let be *N*). This limit can be estimated in advance supposing that the equation solving is more time consuming process than the function evaluation:

$$\begin{gathered} T_{sim} = \alpha_{eval} N + \alpha_{solve} N^{\beta} \leq ST \\ N \approx e^{log\left(ST/\alpha_{solve}\right)/\beta} \end{gathered}, \tag{1}$$

where α_{eval}, α_{solve}, and β are the time effort parameters of the simulator on a given system.

The distinguishing constraint motivated us to develop the novel model generating concept. The introduced technique can be concluded as a *heuristic search process in collections of different complexity block models* that build up the full-chip behavioral model.

In advance, the hierarchic architecture of the circuit is supposed to be known. Initially the model prototypes for every block are clarified at several complexity levels. These prototypes are derived both empirically and physically at multiple precision levels priory to the optimization:

- **Different complexity physical device models (e.g. MOS transistors [16]);**
- **Symbolic analysis of different error criteria for linear blocks [17] (e.g. amplifiers);**
- **Parameter extraction and macromodel generation methods for nonlinear blocks with different error tolerances [14], [15].**

Since the full parametric modeling has no sense due to the known physical technology data, design reuse, and evidently insignificant elements, basically semi-parametric models are used. Hence, the different parameters of the prototypes are selected to be numeric or to remain variable. After the numeric simplifications the model library is not modified any more, and the selection process launched.

A *Selection Optimization*

As a heuristic search and optimization process the adaptive simulated annealing (ASA) technique has been chosen [19]. Once the models of a population are created, the complexity (1) and later on the numeric comparison is carried out in order to rank the variants.

The brief properties of the implemented ASA are the followings: the number of iterations *I* is set between *N*/10..*N*/100, the temperature schedule as a function of the *i* iteration number is: $T_i = exp\left(-(i-1)^2/20I\right)$. The maximal length of a random jump from the actual selection is simply $\pm N \cdot T_i$. A selection is accepted if it works with less error, and also accepted if a uniformly distributed random number between 0 and 1 is less than the square of the actual annealing temperature: $rand[0,1] < T_i^2$. In addition, the best selections are stored in order to not loose a good solution during the search process. The fitness factor was the numeric error (that will be described in the next chapter).

If the complexity of a selection is estimated to be more than the allowed bound, the numerical comparison and the selection are automatically skipped. The numeric evaluation is performed using about a dozen of relatively complicated time domain waveforms of the most detailed form of the design (e.g. the extracted netlist of the original layout). The modeling process is illustrated in Fig. 1.

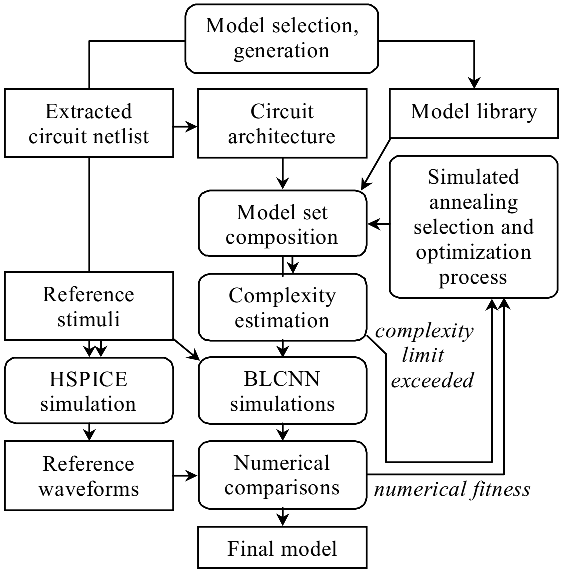


Fig. 1. The flow diagram of the modeling process.

B *Waveform Comparison and Error Metric*

The waveform difference is calculated in a special controlled manner. For final numerical fit a strict metric must be used in order to measure the precision. Meanwhile, in the selection optimization phase an error definition is needed, which allows the error space to be more "smooth" and tractable, causing faster convergence. In order to fulfill these requirements, a compact metric has been defined.

It can be concluded as a continuos transition between a "filtered" difference calculation to a strict one. It is composed of a relative difference metric and a Euclidean distance operator. The former one has been chosen to produce the following ε waveform as

$$\varepsilon\left(x(t), y(t)\right) = \frac{\left|x(t) - y(t)\right|}{\left|x(t)\right| + M}, \tag{2}$$

where *x*, *y* are the reference and the evaluated waveforms, and $M = max(x) - min(x)$ is the dynamic range of the reference. This definition allows the error calculation even for almost zero reference signal [13].

The difference waveform is than transformed by an Euclidean operator [14], which introduces the "nonlinear filtering" by means of removing small phase errors, glitches, and generally the high-frequency behavior. The operator produces a new waveform (E_z) calculating the shortest Euclidean distance for a curve (*z*) as

$$E_z^s(t) = \inf_{t \in [0,T]} \sqrt{z^2(t) - s^2(t - t_0)^2}, \tag{3}$$

where T is the simulation time and s is a scaling factor that scales the time distance to an equivalent voltage value. The resulting waveform presents the shortest distance between the point $(0, t_0)$ and an arbitrary point $(t, z(t))$ on the waveform z. Using this operator we can define the final distance metric given by

$$d_s^p(x,y) = \left\| E^s_{\varepsilon(x(t),y(t))}(t) \right\|_p , \qquad (4)$$

but we used only d_s^1. And last, supposing I waveforms, the average error can be formed by

$$\varepsilon = \frac{1}{I} \cdot \sum_{i=1}^{I} \left\{ \frac{1}{T} \cdot \int_0^T d_s^p \left(x_E^i , x_M^i \right) dx \right\}, \qquad (5)$$

where x^i_E and x^i_M are the i^{th} reference and the model simulation outputs, respectively. The numeric integration is done by trapezoid formula.

Note, that the value of the scaling factor s controls the "filtering" strength of the (3) operator. Typically, s is set by determining the time and voltage resolution. But, choosing s to be zero, the filtering effect disappears and only the linear point-to-point comparison remains. This feature enables us to control the stringent of the error calculation, thus s is to be reduce form the initial value to zero during the optimization parallel with the annealing temperature.

IV. The BLCNN simulator

As was mentioned in Section III, the more complex model could be synthesized and used under the simulation time constraint if the simulation tool is more efficient. Additionally, it must be taken into account that any CNN simulating tool should handle effectively large data arrays, algorithmic issues, and compact result evaluation. Definitely only a dedicated program could satisfy these requirements.

Motivated by these facts, a dedicated mixed-signal simulator framework has been developed (called BLCNN) in standard C code. The framework embodies a computational core of a general mixed-signal electrical simulator [12], the possibility of "hard-wiring" a circuit architecture, and an open interface for different block model descriptions. Once every block model is selected, their features are translated into subroutines and embedded into the simulation environment. After compilation the data arrays (images) and the algorithm descriptions are passed and simulated. Let us summarize some features of the general simulation core of the BLCNN simulator:

- **Trapezoid integration method, Newton-Raphson sparse-matrix nonlinear equation solving.**
- **Advanced time scheduling: Automatic multi-rate detection, latency/wake-up detection, selective route-trace variable updating, automatic computational effort/timestep trade-off. Distinguished transient specific operating modes.**
- **Mismatch introduced variation handling, sensitivity analysis.**
- **Direct data file I/O without graphical interface.**

The proper working of the simulator has been checked by benchmark circuits (such as op-amps or RC ladders).

V. Case study

The behavioral modeling of the ACE4k chip will serve as an application case study. This chip comprises almost 1 million transistors of 0.5 µm standard CMOS technology offered by Alcatel Mietec, on-chip programming and template memories, 64x64 cell array, cell-wise logic, simple arithmetic units, logic and analog memories, special extensions, and optically sensitized areas [11].

The model of the cells was trained on ten different single-cell transients, because of the practically impossible whole array electrical simulation. The signal distribution models was verified separately from the cell array using cell substitution of simple controlled sources.

Every block, including the switches, have been modeled at two up to four different levels, starting from the simple static one up to complex dynamic levels. The total number of different selections was 122,100 and the most detailed model simulation time was slightly more than six hours with the relative error of 0.55%. Then, the fitness constraint on simulation time was set to appr. 4-5 minutes on an UltraSparc 350 MHz model. The reached times speed up resulted only an increase of average relative error to 0.67%.

The measured simulation time of the BLCNN simulator, the extrapolated data the AHDL behavioral simulation [3], and the HSPICE simulation is presented in Table I. The experiment shows about 80 times less term number and 1,000 times faster simulation compared with the electrical simulator. A time window of a numerical reference comparison is shown in Fig. 2.

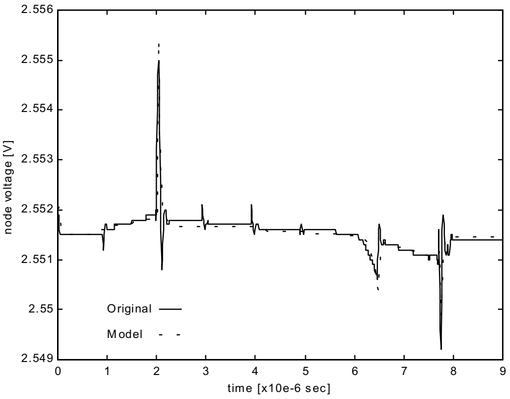


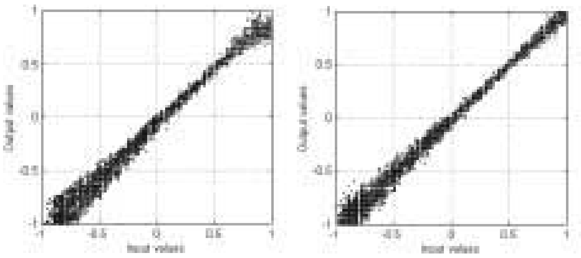
Fig. 2. A time window of one of the output comparisons can be seen. The solid curve shows the result of the HSPICE simulation of the extracted layout, the dotted curve shows the BLCNN simulations of the behavioral model, respectively.

TABLE I
SIMULATION TIMING DATA OF THE WHOLE ARRAY EXAMPLE

	HSPICE	AHDL	BLCNN
Number of terms	*>10M*	*?*	*127k*
Simulation time	*~4 months*	*~3 days*	*192s*
Physical time Timestep	*50µs* *10ps-5ns*	*50µs* *3ps-1ns*	*50µs* *10ps-50ns*

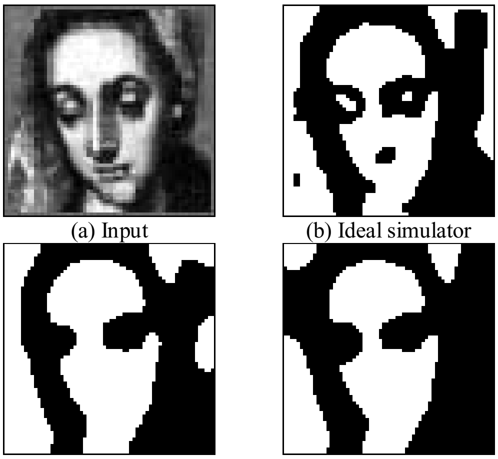
As the proof of concept, results of two entire array executions are presented in Fig. 3 and in Fig. 4. In the first example consequtively ten gray-scale inversions has been performed on a random image. Due to paramter deviation, the output of the chip is a bit "blurred". As the input-output pixel scatter-plots show in Fig. 3, the mismatch and noise models estimated this error quite precisely.

The second example contains a spatial low-pass filtering operation with binarized output. The Fig. 4 shows the different results of an "ideal" CNN simulator [20], the BLCNN simulator, and the real chip embedded in a general algorithm development environment [20]. As can be seen, the strange behavior was estimated again quite properly.



(a) Chip result (b) BLCNNS output

Fig. 3. Input-output scatter-plot of ten consequtive gray-scale inversions on the same random input image.



(a) Input (b) Ideal simulator

(c) BLCNNS output (d) Chip output

Fig. 4. Output comparison of a spatial filtering operation.

Conclusions

We introduced a running time bounded behavioral model simplification technique. Baselines also have been given about the details of the methodology: motivation, automatic block model selection and optimization process, and dedicated simulation tool. The application area of the method covers the modeling of well-structured architectures with unknown block impact on the high-level behavior.

As a case study, results was presented of a high integration level array processor chip. Through this example we demonstrated that the automatic selection of behavioral block model complexities could speed up the simulation efficiency with additional orders of magnitude without significant loss of precision and a proper qualitative behavior estimation of the real systems.

References

[1] L. O. Chua and L. Yang, "Cellular neural networks: Theory and Applications", *IEEE Transactions on Circuits and Systems,* Vol. 35, pp. 1257-1290, October 1988.

[2] T. Roska and L. O. Chua, "The CNN Universal Machine: An Analogic Array Computer", *IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing*, Vol. 40, pp. 163-173, March 1993.

[3] A. M. Arias Drake, "Desarrollo de un modelo de alto nivel del comportamiento del CNN chip-set", M.Sc. dissertation, July 2000.

[4] R. Carmona, G. Liñán, R. Domínguez-Castro, S. Espejo and A. Rodríguez-Vázquez, "SIRENA: A CAD Environment for Behavioral Modeling and Simulation of VLSI CNNs". *International Journal of Circuit Theory and Applications,* Vol. 27, No. 1, pp. 43-76, January-February 1999.

[5] _, "Design of Large-Complexity Analog I/O CNNUC", *ECCTD-99, Design and Automation Day proceedings*, (ECCTD-99-DAD), pp. 42-57, Stresa - Italy, 1999.

[6] A. Paasio, A., Kananen, A., Halonen, K., Porra, V., "A QCIF Resolution Binary I/O CNN-UM Chip", *Journal of VLSI Signal Processing*, November-December 1999, Vol. 23, No. 2/3, pp. 281-290.

[7] B.E. Shi, T. Roska, and L.O. Chua: "Random Parameter Variation in Analog VLSI Neural Networks for Linear Image Filtering", *Int. Joint Conf. on Neural Networks*, Florida, 1994.

[8] P. Kinget and M. Steyaert, "Evaluation of CNN Template Robustness towards VLSI Implementation", *Proc. of IEEE Int. Workshop on Cellular Neural Networks and Their Applications,(CNNA'94)*, pp. 381-386., Rome, 1994.

[9] I. Fajfar and F. Bratkovic, "Statistical Design Using Variable Parameter Variances and Application to CNNs", *Proc. of IEEE Int. Workshop on Cellular Neural Networks and Their Applications,(CNNA'94)*, pp. 147-152., Rome, 1994.

[10] R. Tetzlaff, R. Kunz, G. Geis, "Analysis of Cellular Neural Networks with Parameter Deviations", *Proc. of 13 European Conference on Circuit Theory and Design, (ECCTD'97)*, Vol.2. pp. 650-654, Budapest, 1997.

[11] G. Liñán, P. Foldesy, S. Espejo, R. Domínguez-Castro and A. Rodríguez-Vázquez, " A 0.5µm CMOS 10^6 Transistors Analog Programmable Array Processor for Real-Time Image Processing", *Proc. of the 25th European Solid-State Circuits Conference*, pp. 358-36, Duisburg-Germany, Sept. 1999.

[12] Resve A. Saleh, A. R. Newton, "Mixed-Mode Simulation", Kluwer Academic Press, 1990.

[13] C. Borchers, "Symbolic Behavioral Model Generation of Nonlinear Analog Circuits", *IEEE trans. on Circuits and Systems-II: Analog and Digital Signal Processing*, Vol. 45., No. 10., pp. 1362-1371., October 1998.

[14] Y.-C. Ju, V. B. Rao, and R. A. Saleh, "Consistency Check and Optimization of Macromodels", *IEEE trans. on Computer-Aided Design*, Vol. 10., No. 8., pp. 957-967., August 1991.

[15] G. Casinovi and A. S.-Vincentelli, "A macromodeling Algorithm for Analog Circuits", *IEEE trans. on Computer-Aided Design*, Vol. 10., No. 2., pp. 150-160, February 1991.

[16] J. Bastos, "Characterization of Transistor Missmatch for Analog Design", Katholieke Universiteit Leuven, ISBN. 90-5682-110-5. April 1998.

[17] F. V. Fernández, A. Rodríguez-Vázquez, J. L. Huertas, G. G. E. Gielen, "Symbolic Analyis Techniques", IEEE Press, ISBN: 0-7803-1075-6., 1998.

[18] W. Deams, at al., "Evaluation of Error-Control Strategies for the Linear Symbolic Analysis of Analog Integrated Circuits", *IEEE trans. on Circuits and Systems-II: Fundamental Theory and App.*, Vol. 96., No. 5., May 1999.

[19] A. L. Ingber, "Adaptive Simulated Annealing", Global optimization C-code, Caltech Alumni Association, Passadena CA, 1993.

[20] P. Szolgay et al., "The Computational Infrastructure for Cellular Visual Microprocessors". *Proc. of the IEEE 7th Int. Conf. on Microlectronics for Neural, Fuzzy, and Bio-Inspired Systems*, pp 54-60, Granada, Spain, April 1999.