

An IP-based library for the design of analog Hodgkin-Huxley neurons

T. Levi⁽¹⁾, N. Lewis⁽¹⁾, Y. Bornat⁽¹⁾, A. Destexhe⁽²⁾, J. Tomas⁽¹⁾

⁽¹⁾University Bordeaux 1, IMS Laboratory (CNRS/ENSEIRB), Talence, France

⁽²⁾UNIC, CNRS UPR2191, Gif-sur-Yvette, France

tlevi@ixl.fr

Abstract— This paper presents the neuron-level integration of a complete system that emulates spiking neural networks. Analog ASICs are used for the computation of neurons activity while the connectivity between these neurons is digitally controlled. One ASIC integrates several biologically realistic neurons, following the Hodgkin-Huxley formalism, where mathematical functions are emulated by typical or specific analog blocks. As it is often the case, the development of the analog part of the system requires the largest amount of time, due to the lack of formalism and automation in that domain. One solution to accelerate the analog design cycle is to re-use already designed blocks and accumulated design knowledge, which could be illustrated by the IP (Intellectual Property) concept. Indeed, an experience of about ten years and 19 designed ASICs allow now to have an accurate idea of the system hierarchy and the recurrent analog blocks, which is the basis of IP-based design. We will describe the IP-based library which has been developed for that specific application domain and show how it can be used to accelerate the design cycle of the next ASIC generation.

I. INTRODUCTION

Engineering of neuromorphic integrated systems is a research field where microelectronics encounters biology. The link between both is realized by computational neurosciences which try to understand a part of brain activity, by modeling and simulating spiking neurons or spiking neural networks. The spiking neurons differ completely from formal neurons because their models are biologically realistic. Different levels of modeling exist from the neuron physiology to the plasticity of large neurons networks. One issue is to have the adequate simulation system that implements those models; that is the role of neuromorphic engineering [1], [2]. Important features of such systems are re-configurability, observability and also real-time running, when application field is oriented to physiology with experiments. Hardware implemented spiking neural networks has been developed since the first “silicon

neuron” [3]; this hardware-based simulation presents the advantage to have a competitive running time compared to software-based simulation.

From the microelectronic point of view, one solution is to design analog ASICs for the real-time computation of neurons activity and to digitally control the connectivity between these neurons [4]. As it is often the case, the development of the analog part requires the largest amount of time, due to the lack of formalism and automation in that domain. One solution to accelerate the analog design cycle is to re-use already designed blocks and accumulated design knowledge, which could be illustrated by the IP (Intellectual Property) concept. Indeed, an experience of about ten years and 19 designed ASICs allow now to have an accurate idea of the system hierarchy and the recurrent analog blocks, which are the basis of IP-based design.

In this paper, we focus on the design of those ASICs, especially about the modular implementation and the IP-based design.

II. FROM THE NEURON MODEL TO ANALOG FUNCTIONS

To design neuromimetic IC (in contrast to bio-inspired IC), we chose the Hodgkin-Huxley formalism. The main advantage of this formalism is that it relies on parameters, which are biophysically realistic, by the way of a conductance-based expression of the neural activity (see detailed description of the Hodgkin-Huxley formalism in [5]). We will express here the conductance-based principle and find out generic expressions for these conductance phenomena.

The Hodgkin-Huxley formalism provides a set of equations and an electrical equivalent circuit (Fig. 1) that describe the behavior of separate conductances. Each conductance represents the dynamics of an ionic specie (sodium, potassium or calcium) flowing through the neural membrane. All these ionic currents are integrated on the membrane capacitance following the electrical equation (1),

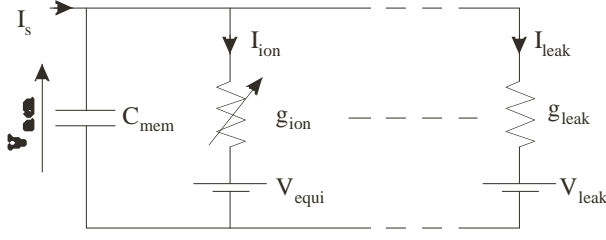


Figure 1: Neuron electrical equivalent circuit

$$C_{mem} \frac{dV_{mem}}{dt} = -\sum I_{ion} + I_s \quad (1)$$

where V_{mem} is the membrane potential, C_{mem} the membrane capacitance and I_s an eventual stimulation or synaptic current. The generic mathematical expression (2) for each ionic current is:

$$I_{ion} = g_{max} m^p h^q (V_{mem} - V_{equi}) \quad (2)$$

$$m_{\infty}(V_{mem}) = \frac{1}{1 + \exp(\pm \frac{V_{mem} - V_{offset}}{V_{slope}})} \quad (3)$$

$$\tau(V_{mem}) \cdot \frac{dm}{dt} = m_{\infty}(V_{mem}) - m \quad (4)$$

where g_{max} is the maximal conductance value, m and h are respectively the activation and inactivation function and V_{equi} the ion-specific reverse potential. For our implementation we will integrate current generators that follow expression (2). Synaptic current are based on similar conductance-based model.

All model equations can be implemented using classical analog building blocks like differential amplifiers, OTAs, current mirrors and current-mode multipliers. For example OTA are used to implement the transconductance (g_{max}). The parameters m and h are described by equations (3) and (4) and require one sigmoidal circuit and kinetic blocks. Neuromimetic ASICs have been designed by our team [6] from 1993. Now, the trend is to integrate more and more complex neural networks, on the basis on recurrent primitive analog blocks.

III. GALWAY : THE LAST ASIC GENERATION

The circuit is organized to provide to user a large variety of configurations for the simulated neural network. As the neural activity is generated by a sum of ionic and synaptic currents on a membrane capacitance, we decided to integrate a set of generic blocks, each able to compute a conductance-based model of ionic or synaptic current. Parameters of the model are stored on

analog memory cells, which values are programmed during the configuration phase of the simulation. During that first phase, the user will also set the topology of the network, i.e. define the blocks connectivity. A set of connected blocks will form an artificial neuron, with their respective currents summed on an external capacitance.

The *Galway* chip we present here comprises (see fig. 2 and 3):

- a set of conductance modules, each able to generate an ionic or synaptic current following the conductance-based model;
- spike-detection modules, to code on 1-bit the neuron membrane voltage;
- a set of synaptic input modules, that activate synaptic conductance modules with a digitally-controlled weight;
- an analog memory cells array, to store the conductances parameters;
- a matrix of switches, to control the neurons topology (i.e. the arrangement of the conductance and synaptic modules that form the artificial neuron);
- digital functions to control data transfer from and to external devices.

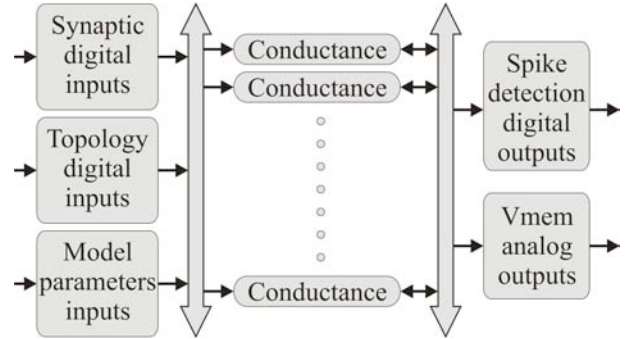


Figure 2: Galway chip architecture and data I/O

The I/O pins are used to:

- output the neuron analog membrane voltage (V_{mem});
 - output the spike information (1-bit coding of V_{mem});
 - input the synaptic weight digital control;
 - connect the external passive elements necessary for computing the kinetics in the model equations;
 - convey the control data;
 - input the analog values for the memory cells;
- convey the digital and analog power supplies.

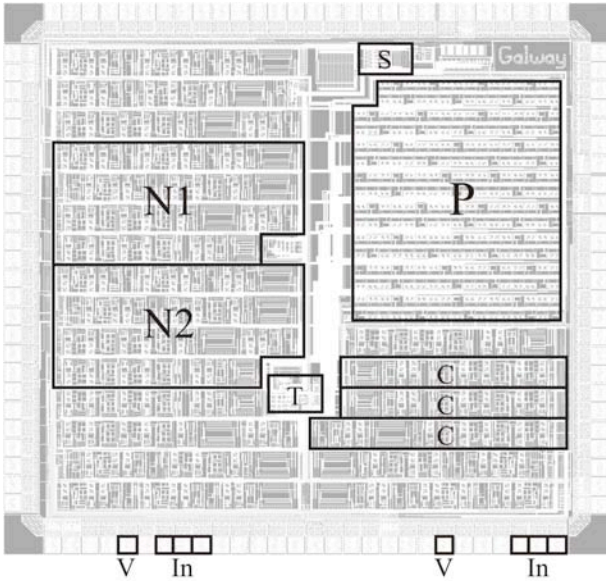


Figure 3: Layout of the Galway chip. C are conductance blocks. P is the analog memory cells array, T the topology control block, S a spike detection block, N1 and N2 represent 2 “typical” artificial neurons, each formed by 4 ionic conductances and 3 synaptic conductances; the V pads convey N1 and N2 respective analog membrane potential, In receive the N1 and N2 respective 3 digital synaptic control inputs. The chip area is 10.5 mm², the number of transistors of the core is 47000 (93% for the analog part of the circuit).

Each chip includes 21 ionic conductance blocks and 15 synaptic conductance blocks. 205 analog parameters are stored in the memory cells. A typical arrangement of the modules is the building of 5 artificial neurons comprising from 3 to 5 ionic conductances each receiving 3 synaptic inputs. Such a structure allows us to address standard configuration of neural networks. The analog signals (membrane voltage, ionic current) are exploited in specific experimental configurations, such as dynamic clamp experiments; in that case, dedicated on-chip conductances are used as artificial synapses, which communicate in real-time with in vitro biological neurons through an *Axoclamp* amplifier (*Axon Instruments*®) [7].

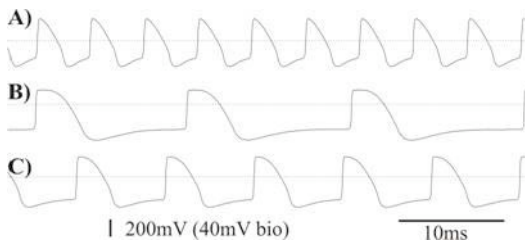


Figure 4: A), B) and C) represent the activity of the membrane voltage of 3 different neurons measured on a Galway chip.

IV. IP-BASED LIBRARY

To accelerate the design cycle for the next ASICs generations, we have developed an IP-based library.

As explained previously, successive generations of ASICs were designed, which integrate the conductance-based neuron models [8], [9]. These circuits were exploited to build a library of electronics function. Each mathematical function appearing in the neuron model corresponds to a generic analog module in the library. Conductance blocks that compute the ionic currents models are built using these generic modules. The model parameters are stored in the analog memory cells. The ionic and synaptic conductance blocks are interconnected to form a neuron following the topology control. Figure 5 describes the well-defined hierarchy of the neuromorphic ASICs.

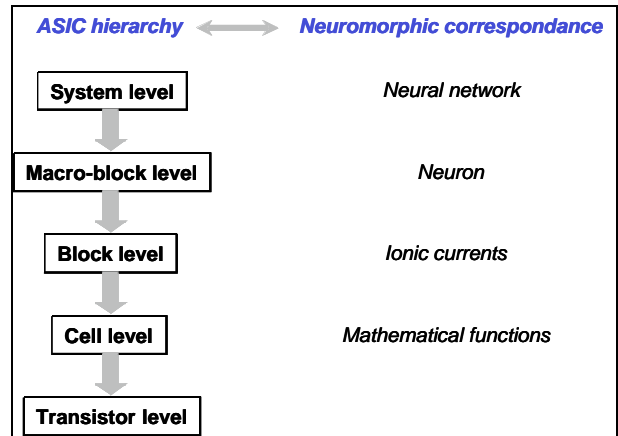


Figure 5: Typical ASIC hierarchy

A. Definition of an IP

Defining the IP content for analog applications is the first important task. Some works already pointed out this concept as the basis of optimization of analog design flow [10] [11] [12].

What should be the main properties of an IP block? First, it should give a precise characterization of already designed block; second, it has to be described with adequate representations or models, consistent with the design levels, to enable the easy re-use of the block, along the complete design flow. Table 1 shows the different descriptions (or *views*) that are embedded in our IP blocks. All views have the same terminals and the same symbol.

Table 1: IP hierarchical description levels

View name	Description / Role
Symbol	visualizes the function
Connectical	verifies connections between blocks (VerilogA)
Functional	models ideal electrical behavior (VerilogA)
Behavioural	models non-ideal electrical behavior, extracted from schematic (VerilogA)
Schematic	transistor-level schematic
Layout	chip masks generation
Characterization	See below ...

For the *connectical*, *functional* and *behavioral* views, we use VerilogA language. These views are useful for multilevel simulations especially in the verification phase of the design process (Bottom-Up design flow). The *functional* view describes the ideal equations of the function to be implemented. The *behavioral* view is more detailed, composed of refined equations which fit the schematic behavior.

One important point is to have a fluent and coherent design flow. That is why logical and mathematical links have been established between the different views. For example the parameters involved in behavioral description are related to the parameters of the transistor-level model; functional parameters are linked to behavioral ones.

The *characterization* view is the most important view for automatic exploration of IP-based architecture (Top-Down design flow). This view includes the main criteria of the re-use methodology. It contains information about the design and the re-use ability of the IP-block. The technology, the supply voltage, the terminals and their validity domains, the links between the functional model and the behavioral model, the area of the layout and the cost of the design (which may be estimated from the area) are defined in this file.

The most important criterium in our application is the validity domain which gives the range of input signals, where the circuit functionality is conserved. This information comes from systematic transistor-level simulation of the primitive *cells* (*cell level*, figure 5); then it is propagated to the *blocks* and *macro-blocks*.

B. IP-Based LIBRARY

All IP-blocks are collected in a data-base. This data-base must implement:

- The hierarchy of the system (figure 5)
- The IPs and their content (Table 1)

At the different levels of hierarchy, IPs are implemented and links between them are defined. Consequently the re-use of blocks of the top-level is possible.

The data-base is implemented with MySQL [13] formalism. Six tables have been created:

- one is for the link father-son between the different entities of the hierarchy
- one table defines each IP
- one table gives the number of IPs available for each function
- three other tables define the set of model parameters for each hierarchical level (*Macro-block level*, *Block level* and *Cell level*).

This library is integrated in the design cycle in two phases: the Top-Down phase and the Bottom-Up one.

C. Top-Down Phase

The Top-Down phase is the automatic design space exploration. The objective is to find one ASIC solution according to the initial specifications. These specifications initially come from biologists and concern the values of model parameters.

Thus a Top-Down exploration is performed, from the *macro-block level* to the *cell level* using the validity domain as a selection criterium. If we have different corresponding IPs, we should choose the one with the largest validity's domain.

The exploration requests are made in Php [14]. These requests are created from the system specifications.

At the final step, a diagnostic on re-usable blocks is returned which quantifies the possibility to re-use IPs for a new ASIC project. In a case of re-use, IPs hierarchical netlists are automatically created for the designer.

The data-base automatic exploration process has been tested for many plausible specifications set. An example of a diagnostic is given in Table 2 and Table 3. It describes the IPs we can re-use.

Table 2: Validation of the ionic currents

Ionic currents	Neuron 1	Neuron 2
Na	Right	Right
K	Wrong	Right
Ca	Right	
Leak	Right	Right

Table 3: Final diagnostic

Neurons	Numbers of IPs we can re-use	Percentage of re-use for this circuit
Neuron 1	49/63	78 %
Neuron 2	51/51	100 %

In case of no solution, the designer has an help in re-design task, to create an another IP which corresponds to the specifications.

D. Bottom-Up Phase

The Bottom-Up phase verifies if the proposed architecture really meets the specifications. Indeed the behavioral and functional models are used to perform a multi-level simulation of the entire ASIC.

All the blocks have been validated using the analog simulator *Spectre* under *Cadence* environment. Due to the huge number of components that compose the chip, it is not possible to perform transistor-level simulation of the final chip. Furthermore, the neural activity is a low frequency activity, and the neural activity has to be simulated for at least tens of ms as shown in figure 5. Let us precise that electrical potential are 5 times greater than biological potential and the biological reference potential (0 V) corresponds to 2.5 V in our simulation. The power supply of the analog part of *Galway* is 5 V.

We decided as a consequence to mix transistor level, behavioral, functional and connectical descriptions of the different blocks to validate their connectivity and ensure the functionality of the whole design.

As an example, a simulation for one neuron of type FS (Fast Spiking [15]) with null synapses inputs lasts 2'17" using *VerilogA* description and 14'56" using transistor level description. Both simulations give the same neural activity as depicted in figure 6.

To perform the validation of the whole circuit, we used the *Hierarchical Editor* of *Cadence* environment. For each simulation, only one block was described at the transistor level while the others were described using behavioral description.

We ran as many simulations as high-level blocks. Figure 7 illustrates one of these simulations, where only one neuron of FS type is not simulated using *VerilogA* view. This simulation lasted 3h13' while an identical simulation using behavioral description for all blocks lasted 44'37".

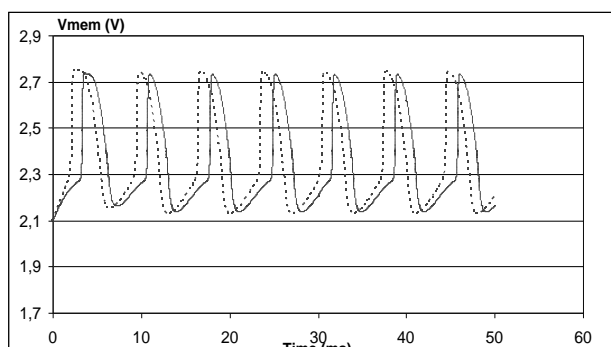


Figure 6: electrical activity of a FS spiking neuron (dashed line corresponds to VerilogA description and continuous line to transistor level description).

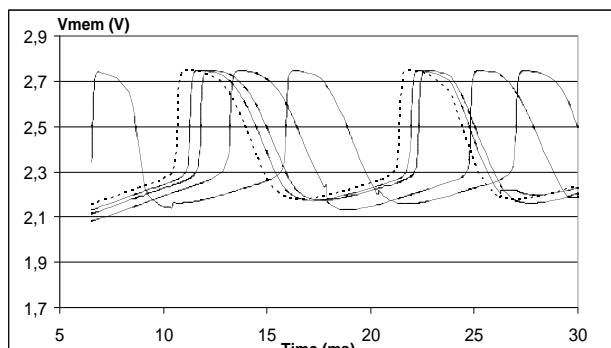


Figure 7: electrical activity of 5 neurons configured in the Galway chip. The dashed line corresponds to one of the neurons (FS type) simulated using its transistor-level description.

V. CONCLUSION

This paper presented the neuron-level integration of a complete system that emulates spiking neural networks. From the neuron model we choose, to the analog functions we implemented, the design cycle is described. To accelerate the time of design of our system and improve the design re-use, we define an IP-based library of analog elementary functions and a specific database of already designed blocks. This

database allows us to be able to build more rapidly the next generations of ASICs (Top-Down Phase), with varying numbers of ionic and synaptic conductances. The library associates behavioral and schematic views of all blocks at all hierarchical level, to be able to perform *Spectre* simulations of the whole chip (Bottom-Up Phase) in a reasonable CPU time. Simulation time is a key issue in our case, considering the complexity of the Kirchoff network of the circuit.

REFERENCES

- [1] Neuromimetic ICs and System for Parameters Extraction in Biological Neuron Models, S. Saïghi, Y. Bornat, J. Tomas, S. Renaud, Proceedings ISCAS 2006, pp.4207-4210, Island of Kos, Greece, May 2006.
- [2] BiCMOS Analog Integrated Circuits for Embedded Spiking Neural Networks, Y. Bornat, J. Tomas, S. Saïghi, S. Renaud, Proceedings DCIS 2005, Lisbon, Portugal, November 2005.
- [3] A silicon neuron, M. Mahowald, R. Douglas, Nature, vol. 354, pp. 515-518, 1991.
- [4] An Analog/Digital Simulation System for Biomimetic Neural Networks, Y. Bornat, S. Renaud, J. Tomas, S. Saïghi, Q. Zou, A. Destexhe, EPFL Latsis Symposium 2006, Dynamical principles for neuroscience and intelligent biomimetic devices, Lausanne (Switzerland), 2006.
- [5] Ionic currents underlying activity in the giant axon of the squid, A. L. Hodgkin, A. F. Huxley, B. Katz, Arch. Sci. Physiology, Vol. 3, pp. 129-150, 1949.
- [6] <http://neuromorphic.ims-bordeaux.fr>
- [7] Analog electronic system for simulating biological neurons, V. Douence, A. Laflaquiere, S. Le Masson, T. Bal, G. Le Masson, IWANN'99, Alicante, Spain, June 1999.
- [8] Hardware computation of conductance-based neuron models, L. Alvado, J. Tomas, S. Saïghi, S. Renaud, T. Bal, A. Destexhe, G. Le Masson, Neurocomputing, Vol. 58-60, pp. 109-115, 2004.
- [9] BiCMOS Analog Integrated Circuits for Embedded Spiking Neural Networks, Y. Bornat, J. Tomas, S. Saïghi, S. Renaud, Conference on Design of Circuits and Integrated Systems (DCIS), Lisbon, Portugal, 2005.
- [10] A Study on Analog IP Blocks for Mixed-Signal SoC, Zheyang Li, Li Luo, Jiren Yuan, Proceedings ASIC, pp.564-567, Beijing, China, October 2003.
- [11] UML/XML-based approach to hierarchical AMS synthesis, I. O'Connor, F. Tissafi-Drissi, G. Révy, F. Gaffiot, Proceedings FDL 2005, Lausanne, Switzerland, September 2005.
- [12] Analog/Mixed-Signal IP modelling for design reuse, N. Martínez Madrid, E. Peralías, A. Acosta, A. Rueda, DATE Conference, Munich, Germany, 2001
- [13] MySQL, P. Dubois, Sams Developer's Library, 2005.
- [14] Programming Php, R. Lerdorf, K. Tatroe, O'Reilly publisher, 2002.
- [15] Intrinsic firing patterns of diverse neocortical neurons, B. Connors, M. Gutnick, Trends Neuroscience, vol.13, pp. 99-104, 1990.

Acknowledgment: This work is supported by the European grant FACETS (FP6-IST-FETPI-2004-15879).