[1]Hardware Oriented Simulation Models of HH-Neurons

Werner Bonath[1], Bastian Wollweber[2]

[1] *University of Applied Sciences Giessen-Friedberg, Wiesenstr. 2, D-35390 Giessen, Germany, bonath@ieee.org*

[2] *Institute of Physiology, Philipps-University Marburg, Deutschhausstr. 2, 35037 Marburg, Germany*

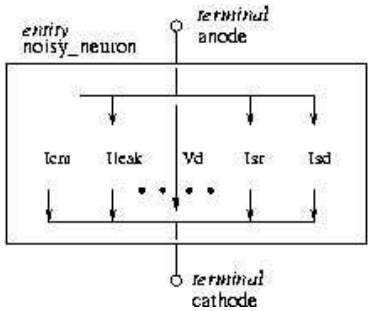
There are a lot of simulation models of the Hodgkin-Huxley model published, usually in procedural or object-oriented programming languages like C, C++ or Java. Our approach is hardware-oriented and uses a concurrent simulation methodology, which is implemented in the VHDL-AMS language standard.

The basic idea is that the slow computer simulation of the HH-equations could be substituted by fast microelectronic hardware. The first step in the design of suitable hardware architectures is to develop models for a system simulation, which serve as the start of a top-down-design methodology. For our project we use the formal language VHDL-AMS because it has three advantages compared to standard programming languages:

(1) It is possible to mix functional with structural models. Functional models of the HH-neuron can easily be imported from literature, structural models are essential for hardware-based simulation approaches.
(2) The language includes modelling of conservative natures like the electrical system with Kirchhoff-Laws.
(3) The simulation is a concurrent one which can directly applied to electrical circuits.

A non-conservative VHDL-AMS-model is already published (http://www.syssim.ecs.soton.ac.uk). For we are interested in the transformation of the software model into an electronic hardware model, we have formulated the Hodgkin-Huxley-Equations with conservative Quantities, i.e. voltages and currents. The simulation system will keep track of the Kirchhoff-Equations. The behavioral model components could easily be transformed into physical models, consisting of electronic circuitry.

Basically one neuron is modelled by an VHDL-AMS entity with two electrical terminals. The internals consist of several current paths, which lead each to one differential equation system.




```
   constant v_sr: REAL := -90.0E-3; -- equilibrium potential
   constant s_sr: REAL := 0.25;
   constant v_0sr: REAL := -40.0E-3; -- half activation potential

begin
   -- 1. membrane equation (rest of...)
   i_g == Cm*vd'dot;
   i_l == g1*(vd - vl);
   -- 2. Fast ionic currents for action potentials
   i_d == rho * g_d * a_d * (vd - v_d);
   a_d == 1.0/(1.0+exp(-s_d*(vd-v_0d)));
   i_r == rho * g_r * a_r * (vd - v_r);
   a_r'dot == phi * (1.0/(1.0 + exp(-s_r*(vd - v_0r)))- a_r)/tau_r;
   -- 3. slow ionic currents for oscillatory activity
   i_sd == - rho * g_sd * a_sd * (vd - v_sd);
   a_sd'dot == phi * (1.0/(1.0 + exp(-s_sd*(vd - v_0sd)))- a_sd)/tau_sd;
   i_sr == - rho * g_sr * a_sr * (vd - v_sr);
   a_sr'dot == phi * (1.0/(1.0 + exp(-s_sr*(vd - v_0sr)))- a_sd)/tau_sd;
end architecture behavior;
```

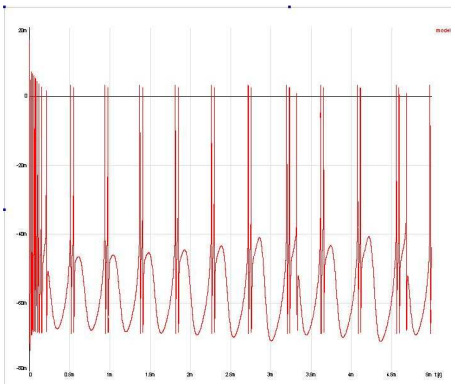
The figure shows the model structure and a part of the VHDL-AMS source code, where currents and voltages are modelled by a branch-quantity with a through-component for each current path. Absolute quantities are necessary for the activation variables and their defining differential equations.

As an advantage to electronic simulators VHDL-AMS allows the direct coupling of different natures. In case of the HH-neuron, we introduce the temperature as additional port into the model. In a network, the temperature will be specific for each neuron.

[1]Supported in parts by BioSim, NoE project No. 005137

A noise-model can be easily implemented with the box-muller model (*VHDL-AMS Modeling of VCSEL including Noise*, Mohamed KARRAY , Patricia DESGREYS and Jean-Jacques CHARLOT, TELECOM Paris). It has been shown, that the VHDL-AMS library functions deliver good approximations for a Gaussian probability distribution.

Transformation of a programming language based simulation model into a hardware oriented model has also to observe typical values of the model parameters, for the simulator will work on physical units. Numerical problems, which arise during numerical solution of the differential equations, can be avoided by proper substitution of real physical units.



In the presentation we will compare typical results of the VHDL-AMS-model (figure above) to an existing Java-based procedural model. Differences occur due to the automatic timestep control of the simulator, which leads to an enhanced accuracy, but also to relative long simulation times and potential instable simulations. Another issue of the hardware-oriented model is its startup-behavior, an example is shown in the figure. As simulation tools there exist several VHDL-AMS systems.