Zbornik radova 50. Konferencije za ETRAN, Beograd, 6-8. juna 2006, tom I
Proc. 50th ETRAN Conference, Belgrade, June 6-8, 2006, Vol. I

MISSION LEVEL MODELING AND SIMULATION ENVIRONMENT FOR HARDWARE/SOFTWARE SYSTEMS

Bojan Anđelković, *Elektronski fakultet Niš*
Volker Zerbe, *Technical University of Ilmenau, Germany*

Abstract – *During the modern, mixed-signal system-on-a-chip design flow there is a strong demand for modeling and simulation at different levels of abstraction. A hierarchical mission level modeling and simulation environment for hardware/software systems is presented in this paper. The presented modeling language provides mission level system simulation in the context of its operational conditions. Also, it enables integration of lower level, implementation models to analyze the impact of low-level technological parameters to the complete system behaviour.*

1. INTRODUCTION

Modern mixed-signal System-on-a-Chip (SoC) designs consist of heterogeneous components from different physical domains from analog, digital and mixed-signal hardware components to embedded software and non-electrical elements. Therefore, such designs require powerful modeling and simulation languages covering system architecture, mission/operational environment modeling, embedded software, register-transfer-level, analog and mixed-signal behavioural modeling and transistor level modeling. Also, they should provide proper verification at all abstraction levels, from system down to implementation. Since it is necessary to validate the system concept very early in the design flow, mission/operational and system level modeling and simulation at high level of abstraction are very important during the design process. In addition hardware and software design teams need a common modeling language capable to describe various electronic and non-electronic components as well as software routines.

Standard hardware description and verification languages used for system and implementation level modeling, such as VHDL-AMS, SystemC, OpenVera, PSL, SystemVerilog do not meet all of the requirements in modern mixed-signal SoC design [1].

This paper presents the idea of developing a uniform modeling and simulation environment covering different levels of abstraction in the design of mixed-signal SoC. In this way designers can describe mission and system level modules and, after system validation at this level, replace some of the components with lower level, more detailed models (digital circuits, transistor level models etc.) to verify the complete system implementation in its working environment. It is also possible to analyze how low level technological parameters influence the performance of the whole system. Such modeling and simulation framework covers the complete system design flow from mission/operational level down to implementation.

2. MISSION LEVEL MODELING AND SIMULATION

Design flow of a mixed-signal SoC starts with modeling of architecture at high-level of abstraction to decrease simulation time and to get an early feedback of the complete system behavior. The system architecture should be refined for functional correctness at mission/operational level to validate the complete system in the context of its operational conditions. Mission level design and simulation integrate architectures, functions, system environments and missions into a single framework. In this approach a virtual prototype of the entire heterogeneous system including typical operational conditions is developed. In that way it is possible to verify the impact of design changes and implementation decisions on overall system performance, improving chances of first-pass system success. After validation at mission and architectural levels, hardware/software partitioning is performed and functional models of the system components are developed. At the end system modules and the complete system should be modeled and validated at implementation level. Development of a hierarchical hardware/software system modeling and simulation environment using a common description language and associated simulator, rather than multiple languages and simulators covering different abstraction levels can reduce design time and errors during the design flow [2].

Hierarchical modeling and simulation at various levels along the SoC design flow will be illustrated by an example of an electronic compass system [3].

First, it is necessary to model and simulate the system at mission level. The system mission is to help in orientation on the way from point A back to A via B and C, respectively, as shown in Fig. 1.

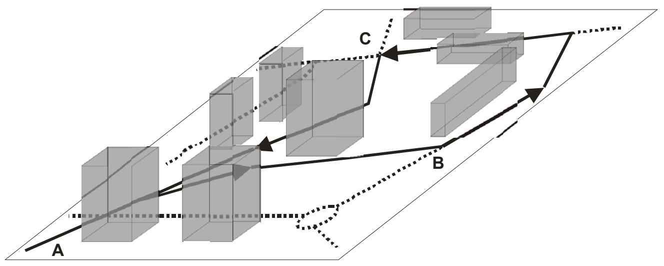

Fig. 1 *Mission of the electronic compass*

The compass at the output generates the value of azimuth α. It is the angle between magnetic north and the heading direction. Magnetic north is the direction of “horizontal” component of the earth’s magnetic field, the earth’s field component perpendicular to gravity. Azimuth α can be calculated as [3]:

$$\alpha = \arctan\left(\frac{B_{ey}}{B_{ex}}\right) \tag{1}$$

where B_e denotes earth's magnetic induction, and B_{ex} and B_{ey} are magnetic induction components in heading direction and sideward, respectively. Therefore, the compass should incorporate magnetic sensors to measure two horizontal earth's magnetic induction components, B_{ex} and B_{ey}.

During mission level modeling and simulation it is necessary to model the whole system and validate its functionality on the desired path. Therefore, together with the high level system model, appropriate model of the system environment should be built.

In order to develop and simulate models at mission level the software tool MLDesigner is used [4]. It is an integrated platform for modelling and analyzing the architecture, function and performance of high level system designs, either as a standalone system or as a system operating in the context of larger systems, working environment or scenarios (i.e., missions). Modeling is based on creating block diagrams using predefined primitive modules (primitives) from libraries. It is also possible to develop own primitives in a C++ like language. Multi-domain modules can be combined and simulated together. Primary domains are continuous time, discrete event, dynamic data flow and synchronous data flow. Two subdomains - finite state machine and higher order functions, augment these primary domains.

Top level of the electronic compass mission level model developed in MLDesigner is shown in Fig. 2.

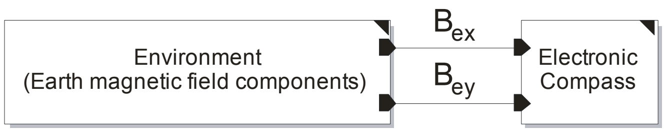

Fig. 2 *Electronic compass mission level model – top level*

The environment module (Fig. 3) consists of blocks "**Bex File Data**" and "**Bey File Data**" that enable reading of appropriate magnetic induction components for the path given in Fig. 1 from files.

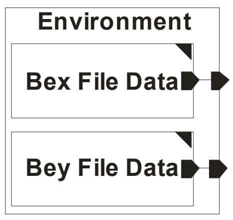

Fig. 3 *Structure of the compass environment module*

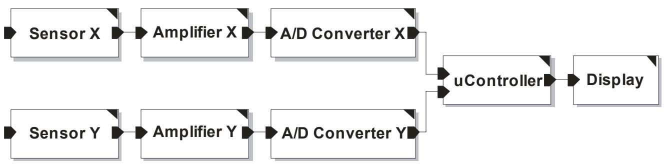

Fig. 4 *Structure of the electronic compass mission level model*

The structure of the compass mission level model is shown in Fig. 4. It consists of the following building blocks: magnetic field sensors (*Sensor X* and *Sensor Y*), amplifiers, A/D converters and microcontroller.

Sensor output voltages are proportional to appropriate magnetic induction components. In mission level model the magnetic field sensor is modeled at high level using the equation [5]:

$$S_A = \left|\frac{V_H}{B}\right| \tag{2}$$

where S_A denotes the absolute sensitivity of a Hall magnetic sensor, V_H is the Hall voltage and B is the appropriate component of the magnetic induction. Fig. 5 shows the description of the sensor in Ptolemy language used in MLDesigner [4]. Since the sensor voltages are proportional to magnetic induction components, the azimuth α can be calculated by applying the *arctan* function to the ratio of the two sensor signals. The microcontroller executes software routine that calculates the necessary *arctan* function using CORDIC (COordinate Rotating DIgital Computing) algorithm [6]. This algorithm is a very efficient way of computing trigonometric functions because it only uses low-end functions like adding, shifting and reading of look-up tables. Fig. 6 shows mission level simulation results of the compass for the path given in Fig. 1. The diagram shows azimuth values for pairs of magnetic induction components at different points along the path.

```
input
{
  name {Be }
  type {float }
}

output
{
  name {Vh }
  type {float }
}

defparameter
{
  name        {S }
  type        {float }
  default     {"1.0" }
  desc        {"Sensor sensitivity." }
}

go
{
  Vh%0 << double(S) * double(Be%0);
}
```

Fig. 5 *Ptolemy language description of the magnetic sensor*

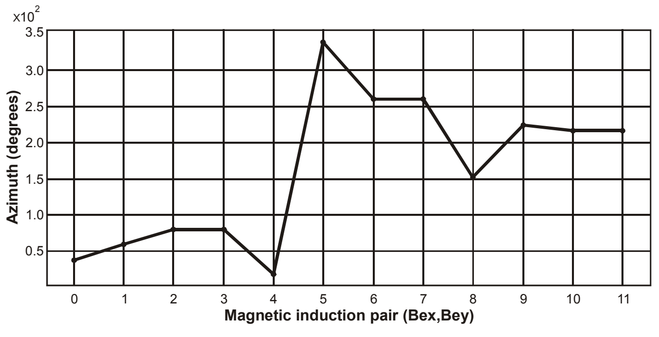

Fig. 6 *Electronic compass mission level simulation results*

3. LOW-LEVEL MODELS INTEGRATION

AleC++ (Analog and Logic Electronic C++) is a proprietary object-oriented Hardware Description Language (HDL) developed for use in the simulator Alecsis [7]. It can be used for modeling of hardware/software systems from various domains at different levels of abstraction. Being a superset of C++ it can be used to describe analog, digital and mixed-signal hardware components, software modules, as well as non-electrical elements. AleC++ also provides some additional useful modeling features not found in other standard HDLs [7].

Since MLDesigner models are hierarchical and based on primitives described in a C++-like language it can be concluded that these models can be translated into AleC++. In order to accomplish that task, it is necessary to map mission level design elements into the equivalent AleC++ language elements by developing appropriate translator. The details about the translation process and generation of equivalent AleC++ models can be found in [8]. In this way it is possible to incorporate MLDesigner models in simulations with the Alecsis simulator.

After the translation of mission level modules into equivalent AleC++ ones, it is possible to replace some of mission level descriptions with lower level models. It enables the designer to analyze the impact of low-level technological and environmental parameters to the complete system performance. As an example the mission level model of magnetic sensor is replaced by a lower level equivalent circuit model. The representation of a CMOS integrated cross-shaped Hall sensor by a lumped circuit model is used [5]. That model provides clear relationship with the physical effects in the sensor and with the material and environmental parameters. It consists of only conventional circuit elements so it is compatible with conventional circuit analysis methods. The sensor is represented using four basic circuit cells BC1, BC2, BC3 and BC4 with MOS transistors as shown in Fig. 7. All cells are generated by rotating the cell BC1 by 90°. Inside the cell BC1 VX and VY are zero-value voltage sources used to measure currents through the MOSFETs. The influence of the magnetic field is represented by the two current controlled current sources FX and FY. The control currents are being those measured by VY and VX, respectively. The "current gains" of these sources K_{xy} and K_{yx} are proportional to the magnetic induction.

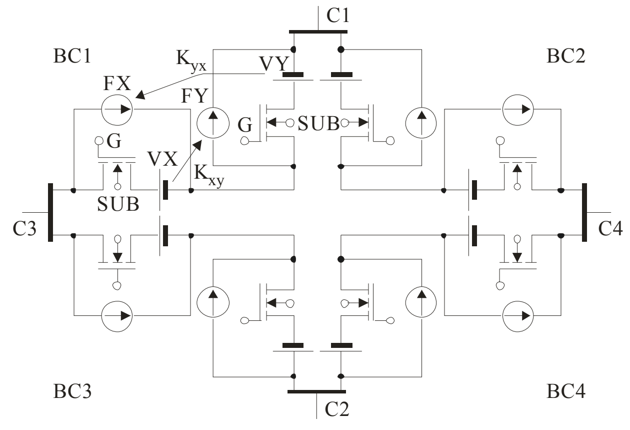

Fig. 7 *Equivalent circuit model of the cross-shaped Hall sensor*

```
module HallSensor (node c3,c4) {
...
//BC1
//MOSFET
mx1(n4, g, c3, 0) {model=my_nmos; l=2u; w=2u; };
//Voltage source
Vx1(n4,n2) 0v;
//Current controlled current source
Fx1(c3,n2,Vy1);

my1(n5, g, n2, 0) {model=my_nmos; l=2u; w=2u; };
Vy1(n5,c1) 0v;
Fy1(n2,c1,Vx1);
...
action (double B) {
    process initial {
...
        Fx1->beta=k*B;
...
```

Fig. 8 *AleC++ description of the cross-shaped Hall sensor*

The AleC++ description of the cross-shaped sensor model shown in Fig. 7 is given in Fig. 8.

In practice there are offsets in the magnetic field sensor voltages due to the tolerances of the sensor elements [3]. It means that the sensor output voltages deviate from zero, if no magnetic field is applied. As each sensor output voltage is proportional to appropriate magnetic induction component plus offset, the azimuth is calculated as:

$$\alpha = \arctan \frac{V_y + V_{oy}}{V_x + V_{ox}} \tag{3}$$

where V_{ox} and V_{oy} are the sensor offset voltages. The maximum azimuth error is proportional to the magnitude of the offsets.

The impact of the offset voltages to the compass functionality can be easily implemented in AleC++ model of the magnetic field sensor as an additional parameter. Fig. 9 shows AleC++ code for top-level description of the compass that includes the influence of the offset voltages to the compass performance. It is assumed that $V_{ox} = V_{oy}$.

```
root top() {

HallSensor sensor;
SDFGain ampX, ampY;
SDFuController uCtrl;
SDFADConverter conv1, conv2;
...
//Magnetic Sensor
sensor(vx,vy) {Bx=0.02;By=0.004; offset=0.002;};
```

Fig. 9 *Top-level structure of the compass system including the impact of the sensor offset voltages*

Generated simulation results for offset voltages between -3mV and 3mV are given in Fig. 10. Calculated azimuth values using mentioned CORDIC routine are represented by boxes. Dashed line shows azimuth values when *arctan* function is applied directly to the sensor output voltages.

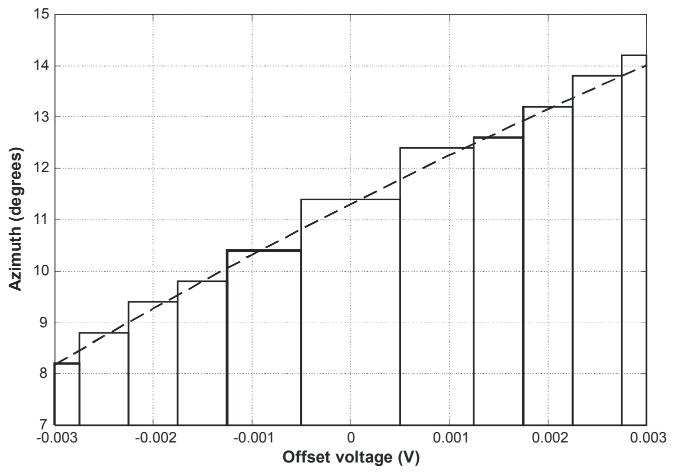

Fig. 10 *Simulation results of the compass in the presence of the sensor offset voltages*

4. CONCLUSION

A mission level modeling and simulation environment for hardware/software systems based on the hardware description language AleC++ is presented in this paper. It extends powerful modeling capabilities of AleC++ with possibility to use mission level modules. The developed hierarchical design framework covers the complete design flow of complex mixed-signal SoC from mission/operational and system down to implementation level. It also enables to easily incorporate lower level models into high level system descriptions, and gives the designer an opportunity to analyze how low-level component technological and environmental parameters influence the whole system functionality and performance.

REFERENCES

[1] B. Anđelković, V. Litovski, V. Zerbe, "New Aspects in HDL's Performance Evaluation", in Proc. *IEEE Region 8 EUROCON 2005 Conference*, 2005, pp. 499-502

[2] T. Zhang, K. Chakrabarty, R. B. Fair, "Behavioral Modeling and Performance Evaluation of Microelectrofluidics-Based PCR Systems Using SystemC", *IEEE Trans. On Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, No. 6, pp. 843-858, June 2004.

[3] Stork, T., *Application Note – Electronic Compass Design using KMZ51 and KMZ52*, Philips Semiconductors, Systems Laboratory Hamburg, Germany, 2000.

[4] *MLDesigner Documentation, Version 2.4*, MLDesign Technologies, Inc., 2003.

[5] R. S. Popović, *Hall Effect Devices*, *Second Edition*, IOP Publishing, 2004.

[6] *Angular Position Development Kit for the 2SA-10*, *Operation Manual*, GMW, 2005., www.gmw.com/magnetic_measurements/Sentron/ sensors/documents/AN_125KIT_manual.pdf

[7] V. Litovski, D. Maksimović, and Ž. Mrčarica, "Mixed-Signal Modeling with AleC++: Specific Features of the HDL", *Simulation Practice and Theory 8*, pp. 433-449, 2001.

[8] B. Anđelković, V. Litovski, V. Zerbe, "A Mission Level Design Language Based on AleC++", *IEEE 25th International Conference on Microelectronics (MIEL 2006)*, (accepted)

Sadržaj – U toku projektovanja hibridnih sistema na čipu postoji izražena potreba za modelovanjem i simulacijom na različitim nivoima apstrakcije. U ovom radu opisano je okruženje za hijerarhijsko modelovanje i simulaciju hardver/softver sistema. Opisani jezik za modelovanje omogućava simulaciju sistema na nivou misije u kontekstu uslova u kojima sistem radi. Takođe, on omogućava integraciju modela nižeg nivoa da bi se analizirao uticaj tehnoloških parametara na ponašanje kompletnog sistema.

OKRUŽENJE ZA MODELOVANJE I SIMULACIJU HARDVER/SOFTVER SISTEMA NA NIVOU MISIJE

Bojan Anđelković, Volker Zerbe