Analog and Mixed-Signal Modeling with VHDL-AMS

Abdulhadi Shoufan and Sorin Alexander Huss
Integrated and Circuits and Systems Lab, Technische Universität Darmstadt, Germany
Email: {shoufan,huss}@iss.tu-darmstadt.de

***Abstract*—This paper describes a compact tutorial on VHDL-AMS. Besides presenting its basic concepts, the tutorial highlights a top-down design approach using this language and details its model execution. The tutorial is intended for instructors with previous knowledge on VHDL and looking for a concise way to introduce VHDL-AMS on a pedagogical basis.**

I. Introduction

Since its standardization in 1999 [1], VHDL-AMS gained in importance increasingly. This language enables the modeling and simulation of analog and mixed-signal systems operating in different physical domains such as electrical, mechanical, and thermal domains. In contrast to VHDL, which received considerable educational attention at an early stage, see e.g. [2]–[4], contributions on teaching VHDL-AMS are rather rare.

Based on our long-time experience on its teaching at the Technische Universität Darmstadt and at the Baden-Württemberg Cooperative State University Stuttgart, this paper describes a compact tutorial on VHDL-AMS. To make the tutorial reusable and extendable for educators, we pay a special attention to pedagogical aspects. This includes –besides a thorough investigation of the VHDL-AMS concepts- a clear definition of the learning outcomes of the tutorial and provides a content structure according to these learning outcomes.

The remainder of the paper is structured as follows. Section II defines the learning outcomes and the structure of our tutorial. Section III details the tutorial content. Section IV presents the assessment and the evaluation. Section V concludes the paper.

II. Learning Outcomes and Tutorial Structure

In the style of constructive alignment [5] we formulate the learning outcomes of our tutorial as follows:

1) Identify the limitations of VHDL in modeling mixed-signal and mixed-domain systems and describe how VHDL-AMS covers these limitations.
2) Understand and apply a top-down design approach for analog systems using VHDL-AMS.
3) Model mixed-signal systems in VHDL-AMS.
4) Model mixed-domain systems in VHDL-AMS.
5) Understand and apply modeling in the frequency domain in VHDL-AMS.
6) Explain the model execution in VHDL-AMS.

While all the learning outcomes aim at widening the declarative knowledge of students, learning outcomes 2, 3, 4, and 5 also tend to deepen the functioning knowledge by applying the learnt theory to model and simulate practical examples using commercial tools. According to these learning outcomes our tutorial is structured in 7 units as depicted in Table I. An hour in this table is a 45-min lecture hour.

TABLE I: Tutorial Structure and Content

Unit	Subject	Form and Hours
1	Review of VHDL	Lecture (5h)
2	Introduction into VHDL-AMS	Lecture (2h) + Lab (3h)
3	Signal-flow and physical modeling	Lecture (2h) + Lab (3h)
4	Mixed-signal modeling	Lecture (2h) + Lab (3h)
5	Mixed-domain modeling	Lecture (2h) + Lab (3h)
6	Modeling in frequency domain	Lecture (2h) + Lab (3h)
7	Model execution	Lecture (3h)

III. Unit Content

A. Unit 1: Review of VHDL

As our tutorial is visited by students from different disciplines we consider precognition in VHDL as desired but not as a prerequisite. For students without any experience in VHDL, Unit 1 gives a concise introduction. A useful method to introduce VHDL relies on answering two general questions: What is hardware and how does VHDL describe it? The answers to these questions are outlined in Fig. 1 and Fig. 2.

According to Fig. 1, a hardware unit has an interface and an internal architecture. The architecture can be described by its structure as a connection of smaller components or by the function it realizes. A hardware function relates to processing signals by applying some operations to these signals and

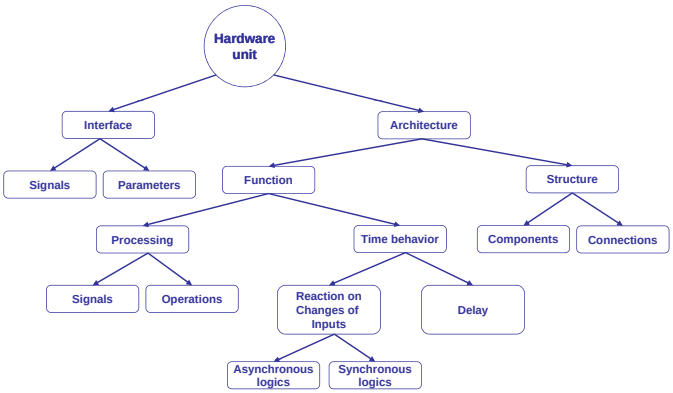

Fig. 1: General Description of a Hardware Module

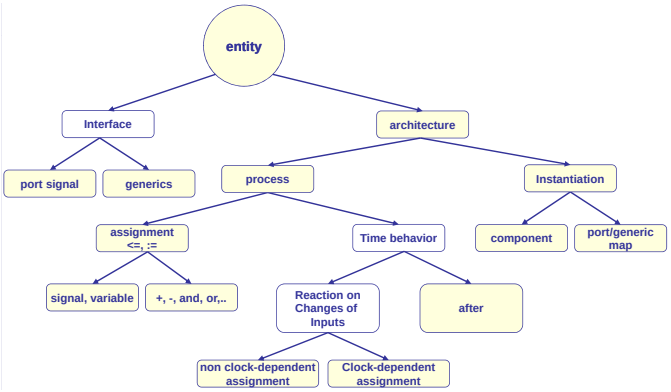

Fig. 2: General View on how VHDL Describes Hardware

producing other signals. Timing behavior of hardware can be specified by the processing delay, on the one hand, and by the speed of hardware reaction to changes of input signals, on the other. Asynchronous logic is specified by immediate reaction. Synchronous logic, in contrast, only reacts after a clock edge.

Fig. 2 outlines how VHDL reflects the hardware properties pointed out in Fig. 1. Based on this outline, we present in Unit 1 small VHDL examples, which illustrate the different language constructs. After that, more detailed aspects in VHDL are treated such as sequential versus concurrent statements, signal versus variables, and multiple-valued logic. In a next phase, the bit-accurate modeling and the cycle-accurate modeling in VHDL are treated.Bit-accurate modeling is important to building efficient data paths. The necessity of bit-accurate modeling can be demonstrated by using the integer type to model an 8-bit adder. The synthesis of this model will result in a 32-bit adder, which is highly inefficient. Cycle-accurate modeling corresponds to hardware description on the register transfer level (RTL), which is supported by most commercial synthesis tools. The advantage of cycle-accurate modeling can be demonstrated by writing a behavior model and an RTL model for a simple system, and by synthesizing them and showing which maximal clock frequency each model achieves. Other important topics in this lecture are parallelism, pipelining and finite state machines.

B. Unit 2: Introduction into VHDL-AMS

The purpose of this unit is to learn the essential concepts of VHDL-AMS including quantities, branch quantities, terminals, natures and simultaneous statements. An appropriate educational way for that is to show why VHDL fails to describe analog systems. In fact, VHDL is able to model an analog system, however, not in all the details required on the physical level. On this level, analog systems are characterized by the following three properties:

1) Value and time continuity
2) Acausality
3) Physical nature

The restrictions of VHDL to reflect the first property can be demonstrated by attempting to model a simple sinusoidal signal source as depicted in Listing 1. The problem of this model is the tricky way of generating a time-dependent

Listing 1: A Signal Source VHDL Model

```
1  entity SOURCE is
2    generic (AMP: real; FREQ: real);
3    port (signal SOURCE_SIGNAL: out real);
4  end entity source;
5  architecture BEHAVIOR of SOURCE is
6  signal CLK: std_logic:= '0';
7  begin
8    process (CLK)
9      variable V : real :=0.0;
10   begin
11     if (CLK'event and CLK= '1') then
12       SOURCE_SIGNAL <= AMP*sin(2.0*MATH_PI*
             FREQ*V);
13       V := V + 20.0e-6;
14       CLK <= not CLK after 10 us;
15     end if;
16   end process;
17 end architecture BEHAVIOR;
```

sinusoidal function. The function SIN defined in the package IEEE.MATH_REAL accepts only real values, while the time in VHDL is of discrete type. Note that the signal SOURCE_SIGNAL is a real signal, i.e. value-continuous, but still time-discrete which does not correspond to the physical case. To solve this problem VHDL must be enhanced by a new object class which represents value- and time continuous objects. This object class is denoted as *quantity* in VHDL-AMS. Additionally, a new time type must be defined, which allows this continuity. Therefore, VHDL-AMS provides a universal time type, which supports both time-discrete signals and variables, on the one hand, and time-continuous quantities, on the other. Modeling SOURCE_SIGNAL as a quantity can be seen in Listing 2. Note that the function ***now*** in VHDL-AMS is of the type floating-point, in contrast to the discrete type in VHDL.

To discuss the acausality property of analog systems consider Listing 3, which shows a possible VHDL model for a resistor. The problem of this model is considering the voltage as an input and the current as an output. In particular, physical systems are acausal, which means that not only the resistor voltage affects its current, but the opposite is true, too: The higher the resistor current, the more power is dissipated in other circuit elements and, thus, the lower is the resistor voltage. This acausality is attributed to the exchange of charge and energy with the environment and to the conservation laws ruling this exchange. To model these aspects new objects are required, which have no in/out mode, on the one hand, and

Listing 2: A Signal Source VHDL Model

```
1  ...
2  architecture BEHAVIOR of SOURCE  is
3  quantity SOURCE_SIGNAL: real;
4  begin
5    SOURCE_SIGNAL == AMP*sin(2.0*MATH_PI*FREQ*
         now);
6  end architecture BEHAVIOR;
```

Listing 3: A Resistor VHDL Model

```
1 entity RESISTOR is
2   generic (R: real);
3   port (signal V: in real;
4         signal I: out real);
5 end entity RESISTOR ;
6 architecture BEHAVIOR of RESISTOR is
7 begin
8   I <= V/R;
9 end architecture BEHAVIOR;
```

are subject to conservation laws, on the other. This object is denoted as *branch quantity* in VHDL-AMS and can either be a potential quantity (*across*) or a flow quantity (*through*). Note that a potential and a flow quantity are always needed to reflect the conservation of energy and charge. Listing 4 shows the VHDL-AMS model of the resistor.

Listing 4 also addresses the third property of analog systems. As a real system may operate in several physical domains, each branch quantity describing this system must be assigned to the domain it operates in. This aspect is essential to model the energy exchange between different physical domains and the associated energy conversion, e.g., from thermal to electrical form. A physical domain is modeled in VHDL-AMS as a *nature*. Assigning a branch quantity to some nature is accomplished by defining it between two access points of this nature. Such access points are referred to as *terminals* in VHDL-AMS.

Besides these essential concepts Unit 2 also treats:

1) The different types of simultaneous statements: simple, if, case, null, and procedural statements.
2) Model solvability rules
3) Tolerance groups and the attribute *tolerance*
4) The predefined attributes *dot* and *integ*

In the lab students model and simulate several electrical circuits including passive elements and analyze several aspects such as the resonance behavior of an RLC circuit.

C. Unit 3: Signal-flow and Physical Modeling

This unit introduces a highly important aspect of VHDL-AMS, which is the description of analog systems on different abstraction levels. This aspect distinguishes VHDL-AMS from other languages and tools, which only support signal-flow models such as MATLAB or circuit-level models such as

Listing 4: A Resistor VHDL-AMS Model

```
1 entity RESISTOR is
2   generic (R: real);
3   port (terminal P1, P2: ELECTRICAL);
4 end entity RESISTOR ;
5 architecture BEHAVIOR of RESISTOR is
6 quantity V across I through P1 to P2;
7 begin
8   V == I*R; -- or I == V/R;
9 end architecture BEHAVIOR;
```

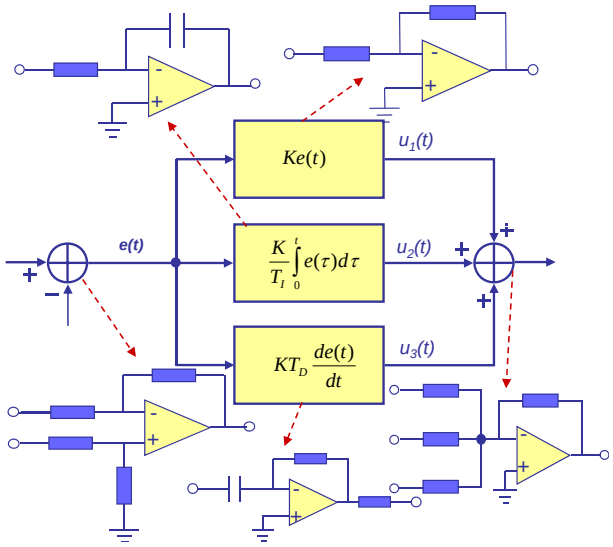

Fig. 3: A PID Controller: From Signal-Flow to Implementation

SPICE. *Free quantities* and *port quantities*, which are essential concepts for writing signal-flow models, are learnt in this unit. For a complete understanding of the top-down design approach we first describe a PID controller as a signal-flow model and write a testbench to simulate it. In the next phase, which is a lab session, each block is successively replaced by its implementation model using operational amplifiers as depicted in Fig. 3. The refined models are simulated using the same testbench used to simulate the signal-flow model.

D. Unit 4: Mixed-signal Modeling

In this unit three aspects of mixed-signal modeling in VHDL-AMS are learnt. These are:

1) How to convert a digital signal into an analog quantity using the attributes *ramp* and *slew*?
2) How do digital events affect the operation conditions of the analog system using *break* statement?
3) How does the analog part release digital events using the attribute *above*?

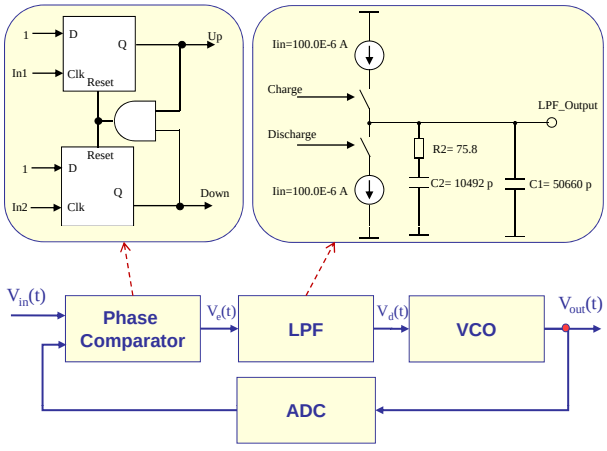

Fig. 4: A Digital PLL as a Mixed-Signal System

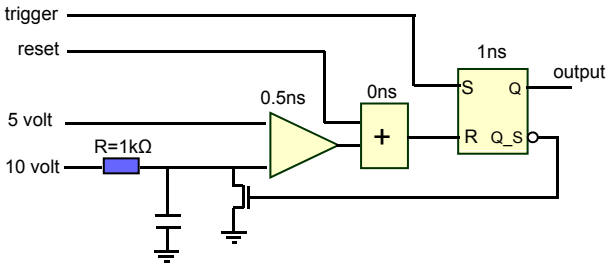

Fig. 5: A Simple Timer for Illustrating Model Execution

Several examples such as AD converters, DA converters, and switch transistors are represented. In the lab a digital phase-locked loop is modeled and simulated, see Fig. 4. A digital PLL is a suitable case study for mixed-signal modeling as it includes both a digital-analog interface (charge-pumb LPF) and an analog-digital interface (ADC). In a further step, the PLL is extended to a frequency synthesizer using a counter.

E. Unit 5: Mixed-domain Modeling

In this unit the following topics are treated:

1) Understanding the IEEE standard packages describing the different physical natures
2) Declaration of own natures
3) Modeling of non-electrical systems
4) Modeling of heterogeneous systems

In the lab students model a gearbox, a DC motor, and a self-heating diode.

F. Unit 6: Modeling in Frequency Domain

In this unit the following points are discussed:

1) The predefined enumeration *domain_type*, which has the values *quiescent_domain*, *time_domain*, and *frequency_domain*.
2) The predefined signal *domain* of the type *domain_type*, which is used by the simulator to determine the simulation mode. At the beginning of simulation, *domain* is always set to *quiescent_domain* to determine the operation point of the system. After that, *domain* is either set to *time_domain* for transient simulation or to *frequency_domain* for frequency simulation. In the latter case a small-signal model is created.
3) *Source quantity* used for the frequency analysis.
4) The attribute *ltf*, which defines the Laplace transfer function for some quantity.

The topics of this unit are illustrated using filter circuits and different transfer functions.

In the lab a stability analysis of the PID controller studied in Unit 3 is performed.

G. Unit 7: Model Execution

The last unit treats the execution of a mixed-signal model in VHDL-AMS, which proceeds in three phases: analysis, elaboration, and execution. Fig. 5 shows a mixed-signal circuit which we use to illustrate the topics of this unit. Note that the timer includes analog components (R and C) and digital components (flipflop and OR-gate), as well as an analog-digital component (comparator) and a digital-analog component (MOS). The topics treated here are:

1) The elaboration of the analog part of a model. This results in the structural equation set describing the conservation equations, e.g., the mesh and node equations.
2) The elaboration of the digital part of a model. This results in a set of processes communicating through nets.
3) The VHDL-AMS simulation cycle which is an extension of the VHDL simulation cycle and includes the analog simulation between two digital events.
4) The interaction between the digital kernel and the analog solver. This interaction occurs when a break statement is executed by the digital kernel or when some threshold defined by the *above* attribute is exceeded during analog simulation.

IV. Assessment and Evaluation

The assessment for our course can be described both as formative and summative, according to [5]. In the formative assessment, which is usually applied in the lab, students perform the design assignments described above to deepen the functioning knowledge. In the lab, students have the possibility to ask questions and get feedback about their design and reached results. Lab assignments are not graded. In this way, we intend that students feel relaxed in accomplishing the assignments, which is essential for success. The summative assessment is performed in form of a written 90-minute exam to see whether and how well students acquired the intended functioning and declarative knowledge. In 2008 the exam was written by 29 students and passed by 24 of them, which corresponds to the general experience with this course.

On their part, students evaluate our course every year anonymously by filling out forms with questions about the technical and didactical aspects of the course. Fortunately, our course has always got above-average marks. This was confirmed by the student council, which awarded it twice as the best course in our department in 2005 and 2007.

V. Conclusion

A compact tutorial on VHDL-AMS was presented to help instructors with planning similar courses on modeling mixed-signal and mixed-domain systems.

References

[1] The Institute of Electrical and Electronic Engineers, Inc, "Ieee standard vhdl analog and mixed-signal extensions," *IEEE St 1076.1-1999*, 1999.
[2] M. Moussavi, "Logic design using vhdl," *29st IEEE Frontiers in Education Conference*, 1999.
[3] A. Sagahyroon, "From ahpl to vhdl: a course in hardware description languages," *IEEE Transactions on Computers*, vol. 43, pp. 449–454, 2000.
[4] A. Etxebarria, I. Oleagordia, and M. Sanchez, "An educational environment for vhdl hardware description language using the www and specific workbench," *31st IEEE Frontiers in Education Conference*, 2001.
[5] J. Biggs and G. Tang, *Teaching for Quality Learning at University*. McGraw-Hill Education, New York, USA, 2004.