Checking Heterogeneous Signal Characteristics Applying Assertion-Based Verification

Stefan Lämmermann[1], Alexander Jesser[2], Martin Rathgeber[1], Jürgen Ruf[1], Lars Hedrich[2], Thomas Kropf[1], Wolfgang Rosenstiel[1]

[1] University of Tübingen, Department of Computer Engineering,
D-72076 Tübingen, Germany
{laemmerm,rathgebe,ruf,kropf,rosenstiel}
@informatik.uni-tuebingen.de
[2] J. W. G.-University of Frankfurt a.M., Department of Computer Science,
D-60325 Frankfurt a.M., Germany
{jesser,hedrich}@em.cs.uni-frankfurt.de

Abstract. Heterogeneous system verification lacks on functional and formal verification methodologies. A verification gap exists between the different signal domains. To bridge this gap an assertion-based design method is essential. This requires the integration of the special analog characteristics in formal digital temporal assertions. Therefore, we defined a new set of mixed-signal assertions to improve the verification process. Our novel approach extends the assertion-based verification techniques for fast falsification. The proposed method is demonstrated by several examples which verify analog signal range behavior, slopes, frequencies, differential algebraic equations, and attenuation with SystemC-AMS simulations.

1 Introduction

Through the additional integration of analog and physical structures in todays microelectronic systems the development process forces engineers to use new elaborated system architectures and description languages. To control the development and verification process for heterogeneous systems a link between specification and models at different domains is necessary. Analog circuits usually exhibit continuous changes in voltage, current and timing, whereas digital systems behavior usually exhibits discrete changes in time and value. Additionally, physical behavior can be described by *differential algebraic equations* (DAE). The interactions and integrations of the different characteristics (analog/digital hardware and different physical phenomena) within one model challenge system designers in modeling and validation of heterogeneous systems.

Today complex mixed-signal system validation is still done by special *analog/mixed-signal* (AMS) simulators that only give an input pattern dependent result. The combination of the different properties between the domains is the challenging part of the verification process. One common verification technique for digital systems is *assertion-based verification* (ABV) [1] which joins desired temporal properties (assertions) and validation during simulation to provide more powerful system verification.

This enables fast falsification and better error localization. The specified properties can be easily included into the design or testbench. Additionally, the assertions can be used in all phases of the further verification process which improve the whole verification and validation process.

The contribution of this paper is the combination between the different special characteristics of analog behavior (signal range, slopes, DEAs, and attenuation), Boolean signals and temporal logic to generate properties for ABV of heterogeneous systems. To reach this aim we defined a novel language namely *mixed-signal assertions* (MSA) which is based on [2]. These MSAs can be integrated into the design and the corresponding testbench for verifying with the developed *SystemC-AMS Temporal Checker* (SCAC) library for SystemC-AMS [3] simulation.

The following sections discuss the application of MSA for heterogeneous systems. Section 2 gives a survey of current approaches regarding heterogeneous / AMS verification and ABV. Section 3 explains our contribution and implementation. The novel method with special important properties is demonstrated in Section 4. Finally, we conclude and point out further work.

2 Related Work

In the last decade *hardware description languages* (HDL) have been widely used to model and simulate systems. The main challenges in the design of heterogeneous systems are the different time and value characteristics. For this purpose VHDL [4] and Verilog [5] are extended with an AMS domain in which DAEs can be integrated [6] [7]. Another system description language called SystemC [8] was also extended with an AMS library [3] adding the ability to integrate DAEs. The library makes it possible to describe complete heterogeneous systems in a C++ environment.

For complex system designs simulation techniques are still an essential validation approach. Additionally, to improve the simulation, the well-established technique in digital system-on-chip design ABV [1] can be used. Assertions can be for instance expressed in the standard *property specification language* (PSL) [9]. An example is $assert(F[3](!(ack \| req)))$, whereby $assert$ represents the *verification layer*. The symbol $F[3]$ signifies the $eventually$-operator of the *temporal layer* and the last part includes the *Boolean layer*. ABV can be classified in *offline monitoring* checking after the simulation and *online monitoring* which is applied during the simulation. Online monitors can detect errors at once but generate an overhead at simulation time (fast falsification).

In contrary to the digital domain, analog design flows suffer from the absence of established ABV methods. Nevertheless, in recent years several academic approaches for verification of analog and mixed-signal systems [10] have been introduced. In [11] and [12] serveral offline monitoring property checking approaches based on modeling AMS designs in terms of equations are introduced. Typically, desired analog properties are characterized through continuous time and frequencies. Commercial tools such as Matlab [13] have recognized the necessity of monitors and support the observation of signals with static references. Unfortunately, a formal property specification language which combines discrete and continuous time and the frequency domain with temporal

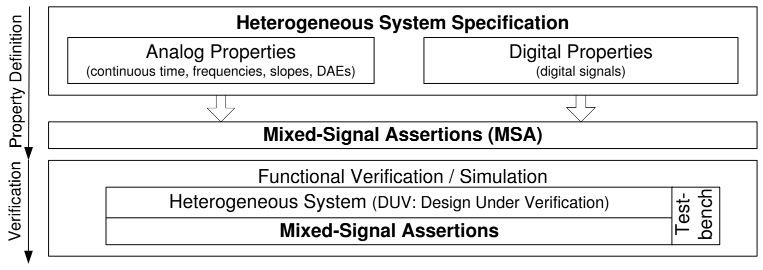
Property Definition
Verification
Heterogeneous System Specification
Analog Properties
(continuous time, frequencies, slopes, DAEs)
Digital Properties
(digital signals)
Mixed-Signal Assertions (MSA)
Functional Verification / Simulation
Heterogeneous System (DUV: Design Under Verification)
Mixed-Signal Assertions
Test-bench

Fig. 1. Verification process with MSAs.

logic for the verification process is missing. Formal specifications are still insufficiently resolved in analog design validation. Yet, all analog formal specifications are principally based on *linear temporal logic* (LTL) or *computation tree logic* (CTL) [14], that cannot fully cover desired analog behavior. However, some rudimentary incomplete academic analog specifications are introduced with the goal of creating a designer-oriented characterization. In [15] CTL-AT was specified which can describe dynamic behavior with time constrained temporal operators. An extension called *analog specification language* (ASL) [16] was defined in order to describe analog behavior. In [17] and [18] a PSL extension named *signal temporal logic* (STL) for describing limited analog properties for simulation is introduced. The temporal layer is defined by finite linear temporal logic in dense time. An online monitoring approach with *monitoring timed automata* (MTA) and VHDL-AMS designs is introduced in [19]. This approach builds up on testbench observer automata without temporal logic. Our work basically improves the above contributions based on the approach in [2], where MSAs are introduced for heterogeneous systems. The challenge of an ABV for heterogeneous systems consists in the combination of continuous and discrete time and frequencies with temporal logic with the ability to represent different signal characteristics.

3 Specification of Mixed-Signal Assertions

The challenge of ABV of heterogeneous systems is the combination of analog characteristics (continuous time domain, frequency domain, slopes, attenuations, and DAEs) and digital signals in formal temporal properties. These properties are specified in our novel MSA language. In contrast to the related work MSAs allow the specification of properties of analog, digital, and physical components for online monitoring. This includes all mentioned analog characteristics in combination with temporal logic and digital signals. In Fig. 1 the proposed verification process with MSA is shown. The upper part begins with the informal requirements which represent the system characteristics. The designer specifies the individual desired behavior of the analog and digital system modules and their interactions in formal MSAs. The result is a heterogeneous system embodying the *design under verification* (DUV) including MSAs for online monitoring.

In contrast to PSL, MSA contains an extension of the separated atomic proposition of the Boolean layer, which consists of digital signals, events, variable conditions, and

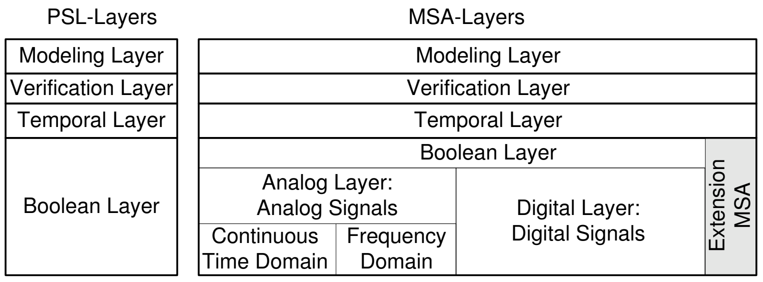

Fig. 2. PSL- and MSA-Layer.

analog signals in combination with their operators. Our MSA specification is based on the syntax and semantics of *finite linear temporal logic* (FLTL) [20], a subset of PSL. Therefore we had to extend the observer AR-automata (Accept/Reject-Automata) [20] by the set of MSA propositions. The temporal layer of PSL for the observer automata is the same as for MSAs. On the left hand side of Fig. 2 are the common PSL layers depicted. The right hand side shows the layers of MSA. The challenge of the heterogeneous system ABV process with MSA is the connection of the different characteristics from the domains, the different time references, and temporal logic. This is achieved by defining a clear syntax and semantics of the analog layer. The goal is to provide a common semantics for assertions which can be used for various heterogeneous design and verification tools for the whole verification process. The new MSAs are exemplarily implemented in SCAC for SystemC-AMS designs.

3.1 Definition of Mixed-Signal Assertions

As already mentioned MSA is a superset of FLTL. It extends FLTL on the atomic layer by a set of expressions that can be used to describe properties of analog signals. Using those new expressions analog signals can be verified in the continuous time domain as well as in the frequency domain. With the operators of the Boolean layer of FLTL various atomic propositions can be combined. Thus, it is possible to describe combined analog-digital properties which is useful concerning the verification of heterogeneous systems.

The syntax of MSA is described by Definitions 1 to 3. In these definitions $\mathcal{A}$, $\mathcal{D}$ and $\mathcal{V}$ are the finite sets of symbols for the analog signals ($\sigma : \mathbb{Z} \rightarrow \mathbb{R}$), digital signals ($\hat{\sigma} : \mathbb{Z} \rightarrow \mathbb{B}$), and variables ($\mathbb{Z} \rightarrow \mathbb{R}$), respectively. The non-terminal symbols are used for the different types of operators defined in the second column of Table 1. The third column of the table gives a short explanation of the semantical meaning of the operators.

First we define the atomic propositions for the continuous time domain. They are given by the set $\mathcal{SE}$ in the following definition:

Definition 1.

– *The set of **DV**-expressions $\mathcal{DV}$ is the smallest set satisfying:*
$\mathcal{A} \subset \mathcal{DV}$ *and* $\delta \in \mathcal{DV} \Rightarrow DV(\delta) \in \mathcal{DV}$

Table 1. MSA operators

	symbols	**semantical meanings**
arithmetic operators	$\odot \in \{+,-,*,/,\hat{}\}$	plus, minus, multiply, divide, power of
relations	$\boxdot, \square \in \{>,<,=,\geq,\leq\}$	greater, lower, equal, greater-equal, lower-equal
Boolean operators	$\bullet \in \{\&, \vert, \mapsto\}$!	and, or, implies not
quantifiers	$\circledcirc \in \{FA, EX, FX\}$	for all, exists, combination of for all and exists
analog operators	$AO \in \{ST, TO, NO\}$	several times, throughout, now
temporal operators	$\diamond \in \{G, F, X\}$	always, eventually, next
Cycle time intervals	$\triangleright \in \{[t_1], [t_2 : t_3]\}$ $t_1, t_2, t_3 \in \mathbb{N}, t_1 \leq t_2$	t_1 clock cycles of time interval t_2 begin and t_3 end of time interval

– *The set of **S**T/TO/NO-**T**erms $\mathcal{ST}$ is the smallest set satisfying:*
  $\mathbb{R} \cup \mathcal{DV} \subset \mathcal{ST}$ *and* $TIME \in \mathcal{ST}$ *and* $\tau, \vartheta \in \mathcal{ST} \Rightarrow (\tau \odot \vartheta) \in \mathcal{ST}$
– *The set of **S**T/TO/NO-**F**ormulas $\mathcal{SF}$ is the smallest set satisfying:*
  $\tau, \vartheta, \eta \in \mathcal{ST} \Rightarrow (\tau \boxdot \vartheta) \in \mathcal{SF} \wedge (\tau \boxdot \vartheta \square \eta) \in \mathcal{SF}$ *and*
  $\varphi, \varrho \in \mathcal{SF} \Rightarrow (\varphi \bullet \varrho) \in \mathcal{SF} \wedge (\neg\varphi) \in \mathcal{SF}$
– *The set of **S**T/TO/NO-**E**xpressions $\mathcal{SE}$ is the smallest set satisfying:*
  $\varphi \in \mathcal{SF} \Rightarrow AO(\varphi) \in \mathcal{SE}$

The next Definition declares the syntax of the atomic propositions describing properties of the frequency domain of analog signals which are given by the set $\mathcal{FF}$ in the following definition:

Definition 2.

– *The set of **FR**-expressions $\mathcal{FR}$ is the smallest set satisfying:*
  $\tau \in \mathcal{ST} \wedge \omega \in \mathbb{R} \Rightarrow FR(\tau, \omega) \in \mathcal{FR}$
– *The set of **F**requency-**T**erms $\mathcal{FT}$ is the smallest set satisfying:*
  $\mathbb{R} \cup \mathcal{V} \subset \mathcal{FT}$ *and* $\xi \in \mathcal{V} \Rightarrow AMP(\xi) \in \mathcal{FT}$ *and* $\tau, \vartheta \in \mathcal{FT} \Rightarrow (\tau \odot \vartheta) \in \mathcal{FT}$
– *The set of **F**requency-**F**ormulas $\mathcal{FF}$ is the smallest set satisfying:*
  $\tau, \vartheta, \eta \in \mathcal{FT} \Rightarrow (\tau \boxdot \vartheta) \in \mathcal{FF} \wedge (\tau \boxdot \vartheta \square \eta) \in \mathcal{FF}$ *and*
  $\varphi, \varrho \in \mathcal{FF} \Rightarrow (\varphi \bullet \varrho) \in \mathcal{FF} \wedge (\neg\varphi) \in \mathcal{FF}$ *and*
  $\psi \in \mathcal{FR} \wedge \xi \in \mathcal{V} \wedge \varphi \in \mathcal{FF} \Rightarrow \circledcirc(\psi, \xi, \varphi) \in \mathcal{FF}$

Considering Definitions 1 and 2 and the syntax definition of FLTL, we defined the following syntax of MSA:

Definition 3.
*The set of all feasible **MSA**-expressions $\mathcal{MSA}$ is the smallest set satisfying:*
$(\mathcal{D} \cup \mathcal{FF} \cup \mathcal{SE}) \subset \mathcal{MSA}$ *and* $\alpha, \beta \in \mathcal{MSA} \Rightarrow (\alpha \bullet \beta) \in \mathcal{MSA} \wedge (\neg\alpha) \in \mathcal{MSA}$ *and* $\alpha \in \mathcal{MSA} \Rightarrow \diamond(\alpha) \in \mathcal{MSA} \wedge \diamond \triangleright (\alpha) \in \mathcal{MSA}$

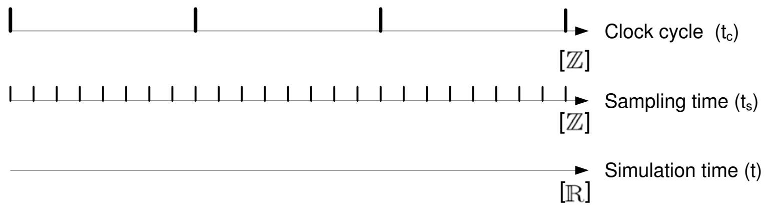

Fig. 3. Different times domains for MSA.

Our focus is the analog layer which is divided in two parts: At first, the continuous time domain with the analog operators ($\mathcal{SE}$) which evaluate the analog signal directly. Secondly, the frequencies operators ($\mathcal{FF}$) which analyze frequency domain of the analog signal.

The semantics description is focused on the new atomic propositions of analog signals. Therefore, we explained the evaluation function for the novel atomic propositions added to FLTL. This evaluation returns a Boolean value which indicates whether the atomic expression is true or false at a clock cycle which results in a Boolean trace $BTrace$. The real time and the relation between the digital discrete time steps triggered by a clock cycle and the sampling time of analog modules are the challenge for MSA which is shown in Fig. 3. The analog signals are discretized by sampling time (dense time). Digital signals use delta-time cycle which can occur during clock cycle and is done on the simulation behavior.

Digital Assertions as PSL or FLTL are based on the discrete clock cycle of the systems. The connection of the digital and analog part of MSA is realized by checking the analog signal during a clock cycle. The analog signal example used for the semantics is shown on the left hand side of Fig. 4. The ordinates are labeled with the amplitude and the abscissa with the time. The right hand side depicts the results of the discretization to dense time (sampling time) of the signal which is executed by the simulation tool. This is an analog function trace ($ATrace$). The $\mathcal{ST}$ expression allows the construction of formulas with the sampling time steps $TIME$ and analog signals. The analog characteristic is described by a $\mathcal{SF}$ like the inequation $(\sigma_s < -6.8)$ which represents the area where the property holds. An inequation is depicted with a bold line on the right hand side of Fig. 4.

The analog and frequency domains need two steps for the transformation from continuous time to Boolean values. The first step performs value discretization with inequations from dense time to Boolean dense time values. The construct $\mathcal{ST}$ of the syntax includes equations with analog signals, simulation time, and values. The special construct DV (derivative) is used to describe slopes and DAEs of an analog signal. The result of a $\mathcal{ST}$ is a value of $\mathbb{R}$ ($\mathcal{ST} \times \mathbb{Z} \rightarrow \mathbb{R}$). In the $\mathcal{SF}$ we specify the condition between the equations which are evaluated at a sampling point. Additionally, the conditions can be combined with logic operators. The results of $\mathcal{SF}$ is a Boolean value ($\mathcal{SF} \times \mathbb{Z} \rightarrow \mathbb{B}$).

For the common time domain we used the clock cycle. MSA needs new constructs analyzing the analog signal behavior during one clock cycle. This is the next step from Boolean dense time values to a Boolean value at a clock cycle. For this, we specify $AO := \{ST(h), TO(h), NO(h) | h \in \mathcal{SF}\}$ with TO (*throughout*), ST (*several times*) and NO (*now*). The meaning of $TO(\mathcal{SF})$ is that the condition formulated by a $\mathcal{SF}$ must hold during the whole clock cycle. In contrary, the condition of $ST(\mathcal{SF})$ holds at least once during a clock cycle. Operator $NO(\mathcal{SF})$ requires that the condition is proved immediate at the beginning of a clock cycle. The Boolean traces for TO, ST and NO depend on a clock cycle which is shown in Fig. 5. For TO and NO the traces are always false. In contrast, for ST the trace gets true once.

The frequency domain takes the discretized input signal shown in Fig. 4. Then a transformation from the continuous time to the frequency domain with a discrete *fast fourier transformation* (FFT) is executed for every clock cycle as depicted in Fig. 6. With $\mathcal{FF}$ the frequencies of an analog signal σ_s can be analyzed. The frequency domain is calculated from $FR(\sigma_s, \omega)$ as depicted in Fig. 6. The desired areas are determined by using spectral threshold values ω. The result is a set of frequencies $Freq$. The property is a set of allowed or not allowed frequencies $f \in \mathcal{FT}$ which are described by an inequation ($20 \leq f \leq 25$). This is shown in Fig. 6 as a highlighted box. The evaluation of the quantifier operations of $\mathcal{FF}$ result in a mapping to Boolean values trace triggered by the clock cycle. EX means that the desired frequencies are contained in the frequency band of the signal σ_s. FA has the behavior, that the frequency band contains in the desired frequency. FX is a combination of FA and EX. In this case the property is never satisfied because there are no frequencies between $20Hz$ and $25Hz$.

The accuracy of MSA depends on the sampling rate S. Through the sampling rate it can be avoided to lose important information of signal values or slopes. A high sampling rate decreases the chance to miss important informations. The disadvantage is a higher simulation time. The accuracy analysis of the frequency band f_{band} depends also on S as $f_{band} = 0.5 \cdot S$. A bad selected S can lead to *false positives* or *negatives*.

Thus, we can connect the resulting Boolean traces from all domains in the temporal layer of MSA in which the specified analog signals and frequencies operations are replaced with Boolean variables. Then the MSA is converted into an AR-automaton

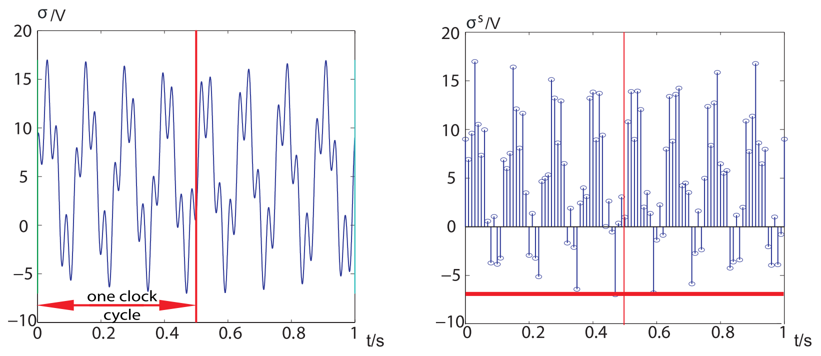

Fig. 4. Analog and resulting discretized signal.

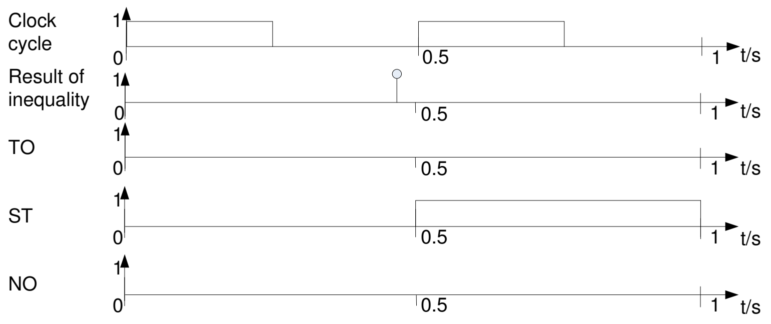

Fig. 5. Boolean trace for TO, ST and NO.

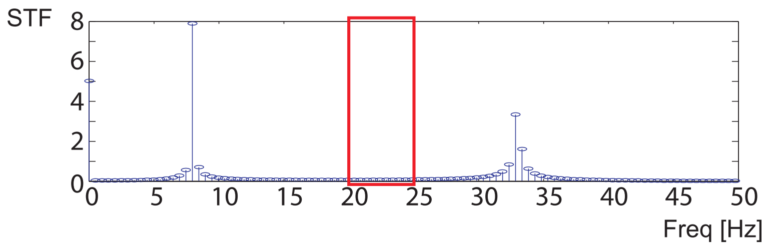

Fig. 6. Frequencies of the analog signal.

which can be executed during system monitoring. An AR-automaton can detect validation or violation of properties on finite system traces, or they are staying in pending state if no decision can be made. The trigger is the clock cycle and the states are dependent on the Boolean variables of the formula. Fig. 7 shows the *MSA example*, the transformed MSA with the Boolean variable $\hat{vd}$ and the correspondent AR-automata. An MSA allows the specification of heterogeneous, pure digital, or analog properties which provide powerful expressiveness for ABV. An *Example MSA* is shown below. It starts with a precondition which includes a digital temporal assertion. This assertion gets true if the value of the three bit vector $!n_2\&n_1\&!n_0$ switches at the next clock cycle to $!n_2\&n_1\&n_0$. The behavior of the postcondition contains only the analog signal vd. After two clock cycles the signal holds at the next clock cycle in the range of 3.9 and 4.1. The conditions are combined with an *implication* $(\mapsto)$ and hold once during the simulation through the temporal operator F.

MSA example:
$F(((!n_2\&n_1\&!n_0)\&X(!n_2\&n_1\&!n_0)) \mapsto X[2]\ TO(3.9 < vd < 4.1))$

3.2 Implementation of SCTC

We are using SystemC-AMS for modeling and simulation heterogeneous systems. The developed SCAC library implementation is used to generate automatically online monitoring checkers of MSAs. SCAC is based on the *SystemC Temporal Checker* (SCTC) [21] and on the novel MSA.

$$F\left(\left(!\,\hat{n}_2\hat{n}_1!\,\hat{n}_0 \ \&\ X\left(!\,\hat{n}_2\hat{n}_1\hat{n}_0\right)\right) \mapsto X\,[\,2\,]\,TO\left(3.9 < vd < 4.1\right)\right)$$

$$\Downarrow$$

$$F\left(\left(!\,\hat{n}_2\hat{n}_1!\,\hat{n}_0 \ \&\ X\left(!\,\hat{n}_2\hat{n}_1\hat{n}_0\right)\right) \mapsto X\,[\,2\,]\,\hat{v}d\right)$$

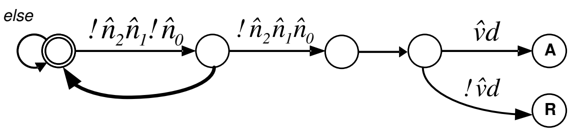
else
$!\,\hat{n}_2\hat{n}_1!\,\hat{n}_0$
$!\,\hat{n}_2\hat{n}_1\hat{n}_0$
$\hat{v}d$
A
$!\,\hat{v}d$
R

Fig. 7. AR-automata of the example MSA.

SCAC is developed for observation of formal properties during simulation with SystemC-AMS. The application of SCAC is done by monitors which are included in the SystemC-Code (design or testbench) without changing and affecting the SystemC model description. The architecture of SCAC extends SCTC with a trace and atomic proposition module which is shown in Fig. 8. The trace module uses the sampling time. The atomic proposition module and SCTC are based on clock cycles. Each analog signal is evaluated through the trace module and the atomic proposition module. The checker is realized as a separate module with a thread process dedicated to execute the AR-automata code corresponding to active properties. Checking the MSAs can be reduced by calling the sca_check(property) method. A cutout of the code is shown in Fig. 9.

The synthesis engine converts the plain property specification into an AR-automata. The AR-automata needs Boolean input variables. Therefore, digital signals are directly connected to SCTC. The analog signals are connected through a trace module reading the values of the signals at every t_s to generate atomic propositions triggered with t_c. The checking process begins with analyzing all analog operations to generate the Boolean input for the AR-automata which are evaluated for a clock cycle. Then each AR-automata is executed during the simulation.

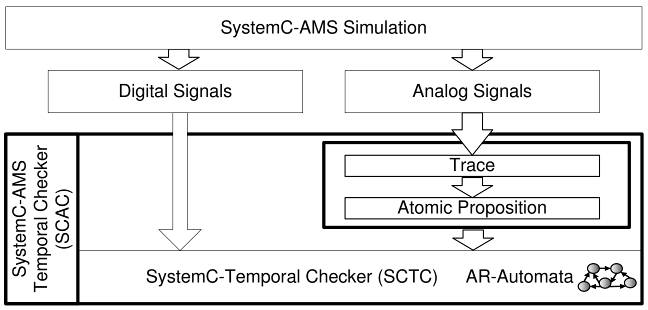

Fig. 8. SCAC implementation.

```
...
sca_check(”F(((!n2&n1&!n0)&X(!n2&n1&n0)) ↦ X[2] TO(3.9 < vd < 4.1))”);
...
```

Fig. 9. Include SCTC in SystemC-AMS code .

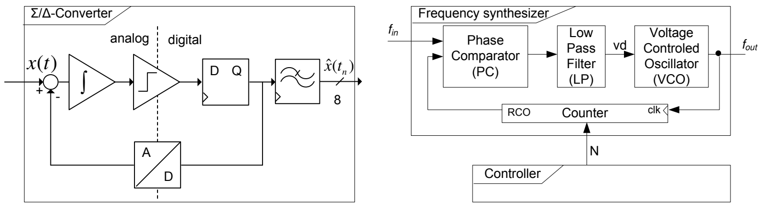

Fig. 10. Σ/Δ - converter and frequency synthesizer.

4 Case Study

We demonstrate in a comprehensible way the applicability of our novel method. To do this we used two examples of heterogeneous systems modeled in SystemC-AMS: a first order Σ/Δ-converter and a frequency synthesizer. With these examples we demonstrate our contribution using six typical analog properties given by MSAs. Significant for the following important properties is the combination of temporal logic, digital signals and the characteristics (signal range, slopes, attenuation, DAE, and frequencies) of analog signals.

4.1 Examples

Σ/Δ-Converter: On the left hand side of Fig. 10 the first order Σ/Δ-converter is shown. The system transfers a time and value continuous signal $x(t)$ into a time and value discrete fixed bit length signal $\hat{x}$. The considered Σ/Δ-converter consists of a subtracter, an integrator, an amplifier, a 1-bit quantizer, a digital low-pass filter, a 1-bit D/A-converter and a D-flip-flop.

Frequency Synthesizer: This system (shown on the right hand side of Fig. 10) is used to synthesize new frequencies. The output frequency f_{out} is always a multiple of the input frequency f_{in}. The factor by which f_{in} is multiplied to get f_{out} is given by the three bit input $n = (n_2n_1n_0)$. The frequency synthesizer consists of four components: a phase comparator (PC), a low pass filter (LP), a voltage controlled oscillator (VCO) and a counter. A clock cycle triggered controller switches n.

4.2 Experiments

Every MSA has at the beginning a *G* operator which is necessary to state that the property has to hold during the whole simulation.

Signal Range: Considering our frequency synthesizer, the signal vd controls which frequency is produced by the voltage controlled oscillator. When the input n is changed the frequency synthesizer has to adjust the value of vd. Depending on the application the correct value has to be reached in a certain time with a defined tolerance. An application could be for example expect that when n changes to (011) the new correct value (which is 4 in our example) will be reached after two clock cycles and with a tolerance of 0.1. This leads to the following MSA-expression:

MSA 1:
$G(((!n_2\&n_1\&!n_0)\&X(!n_2\&n_1\&n_0)) \mapsto X[2]\ TO(3.9 < vd < 4.1))$

Signal Slopes: Again we regard the signal vd of our frequency synthesizer after a change of input n has occurred. If for example the value of n is increased from (010) to (011) the value of vd has to increase from 2 to 4. An application could now require that during the first clock cycle after this change the signal vd strictly increases. This would mean that the slope of vd is always greater than zero throughout the first clock cycle after the change. This can be expressed with an MSA as follows:

MSA 2:
$G(((!n_2\&n_1\&!n_0)\&X(!n_2\&n_1\&n_0)) \mapsto X\ TO(DV(vd) > 0))$

Signal Attenuation: Sometimes the attenuation of a signal has to be verified. In the example of our frequency synthesizer we changed the value of n to (010) after 10ms of simulation time. After that vd approximately reaches the expected value of 2 after 2ms. The value of vd now swings around 2 with decreasing amplitude until we change the value again in 15ms of simulation time. For the time interval between 12 and 15ms an application could now expect a certain attenuation of the signal. If for example the condition

$$2 - 0.1 \cdot e^{-300t} < vd < 2 + 0.1 \cdot e^{-300t}$$

should hold (where t is the simulation time in seconds since 12ms) then we can express this with the following MSA:

MSA 3:
$G(TO(TIME > 0.012\&TIME <= 0.015 \mapsto 2 - (0.1 * 2.718\hat{}((-TIME + 12e - 3) * 3e2)) <$
$vd < 2 + 0.1 * (2.718\hat{}((-TIME + 12e - 3) * 3e2)))$

Analog Behavior with DAE: One part of our Σ/Δ-converter is an integrator. Obviously the input signal dif has to be the derivative of the output signal sum. We can verify this with the following MSA-expression:

MSA 4:
$G(TO(dif = DV(sum)))$

Frequency Range of a Signal: The most important condition that must hold for the frequency synthesizer is certainly $f_{out} = n \cdot f_{in}$. Knowing that f_{in} is 1 MHz, we want to check whether after n has changed to (010) the correct frequency of 2 MHz will be reached within two clock cycles with a tolerance of 0.1 MHz. This can be done with the following MSA:

MSA 5:
$G(((!n_2\&!n_1\&n_0)\&X(!n_2\&n_1\&!n_0)) \mapsto X[2]FX(FR(f_{out}, 0.3), x, 1.9e6 < x < 2.1e6))$

Strongest Frequency: In special cases it may become necessary to verify the value of the strongest frequency. If we consider the same situation as for MSA 5 the strongest frequency should be quite close to 2 MHz. To verify if this is true at least after two clock cycles and with a tolerance of 0.05 MHz we use the following MSA:

```
MSA 6:
$G(((!n_2\&n_1\&!n_0)\&X(!n_2\&n_1\&n_0)) \mapsto F[2]EX(FR(f_{out}, 0.3), x, 2.95e6 < x <$
$3.05e6\&FA(FR(f_{out}, 0.3), y, AMP(x) >= AMP(y))))$
```

4.3 Results

We verified the MSAs in a simulation environment for $50ms$ with the sampling rate $S = 1ns$. Table 2 summarizes the results we got during the process of validating. In the first column all desired MSAs are enumerated. The associated results are listed in the next column, showing at which t_c the property has been accepted or rejected. The entry **A** represents the acceptance state and **R** represents the rejection state of the AR-automata. The last entry informs about which port has been analyzed last, causing the rejection of the MSA. If an MSA is accepted, the inscription ”-” is placed. For example, the entry (R,43,f_{out}) for MSA 2 in column 2 means that the MSA is invalid at $t_c = 43$ due to port f_{out}. We set all MSAs with the G-operator which do not reject at the simulation run as accepted. The last columns show the simulation time for the design with MSA, the simulation without MSA and the resulting overhead in seconds.

Table 2. MSA verification results

MSA	**Example**	**A**/**R** at clock cycle	Simulation time in $seconds$		
			with MSA	w/o MSA	overhead
1	Frequency Synthesizer	R, 18, vd	48.4	45.1	3.3
2	Frequency Synthesizer	R, 16, vd	50.3	45.1	5.2
3	Frequency Synthesizer	R, 14, vd	52.6	45.1	7.5
4	Σ/Δ-Converter	A, - , -	46.3	40.2	6.1
5	Frequency Synthesizer	A, - , -	48.5	45.1	3.4
6	Frequency Synthesizer	A, - , -	49.3	45.1	4.2

The MSAs 1, 2, and 3 are rejected because the frequency generator does not fulfill these specification requirements. The reason for the invalid MSA 1 is caused by the tolerance of 0.1. This tolerance is too small for the oscillation of the vd-signal. The slope property does not hold, because the signal vd does not always increase after switching from (010) to (011). This is based on the feed back loop of the frequency synthesizer which oscillates until reaching the final frequency. The last invalid MSA 3 shows that the requested attenuation does not hold. These are counterexamples which show internal failures of safety properties. All the other MSAs are accepted for this simulation run.

The results in Table 2 show that MSAs are causing an additional simulation runtime overhead. The overhead from the AR-automata depends on the time factors and the complexity of the temporal property described in [20] and [21]. However, the overhead from our MSA extension SCAC is caused by the access to the analog or digital signals from the SystemC-AMS simulation kernel. This part consists only of the trace module and depends on the monitoring of the signals.

5 Conclusions and Future Work

This paper demonstrates the application of specifying and verifying properties of heterogeneous systems during simulation using MSAs. The case study shows the integration of the most important characteristics of analog behavior in combination with temporal logic in MSA. MSAs can be automatically translated into AR-automata as observer automata for SystemC-AMS with our SCAC library.

MSAs enable ABV in all phases of the verification process from specification to full validation of heterogeneous systems. This methodology enables now system simulation with fast falsification and better error detection. This method is a step towards complete functional verification of heterogeneous systems. The next step of MSA is to check the efficiency of the new methodology and include coverage metrics with MSA.

6 Acknowledgements

This work is partially funded by the Landesstiftung Baden-Württemberg in the scope of the BW-FIT project AmbiSense.

References

1. Foster, H.D., Krolnik, A., Lacey, D.: Assertion-Based Design (2nd Edition). Kluwer Academic Publishers (2004)
2. Lämmermann, S., Jesser, A., Weiss, R., Ruf, J., Hedrich, L., Kropf, T., Rosenstiel, W.: An Assertion Based Verification Methodology for SystemC-AMS Designs. In: The 15th Workshop on Synthesis And System Integration of Mixed Information Technologies (SASIMI'09). (Mar 2009) 434–439
3. Open SystemC Initiative: SystemC-AMS. (Mar. 2007)
4. IEEE: IEEE Standard VHDL language reference manual IEEE 1076. (May 2002)
5. IEEE: IEEE Standard for Verilog - Hardware Description Language IEEE 1364. (Apr. 2006)
6. IEEE: IEEE Standard VHDL Analog and Mixed-Signal Extensions IEEE 1076.1. (Nov. 2007)
7. Accellera: Verilog language reference manual - Analog and Mixed-Signal Extensions to Verilog HDL. (Nov. 2004)
8. Open SystemC Initiative: VA Software Corporation and Open SystemC Initiative: Standard IEEE 1666. (Mar. 2006)
9. Design Automation Standards Committee: IEEE Standard for Property Specification Language (PSL), Version 1.1 Standard IEEE 1850. (Oct. 2005)
10. Zaki, M.H., Tahar, S., Bois, G.: Formal Verification of Analog and Mixed Signal Designs: Survey and Comparison. In: IEEE Northeast Workshop on Circuits and Systems (NEWCAS'06). (June 2006) 281–284
11. Grabowski, D., Grimm, C., Barke, E.: Semi-Symbolic Modeling and Simulation of Circuits and Systems. In: IEEE Sysposium on Circuits and Systems 2006 (ISCAS'06). (May 2006)
12. Zaki, M.H., Al-Sammane, G., Tahar, S., Bois, G.: Combining Symbolic Simulation and Interval Arithmetic for the Verification of AMS Designs. In: International Conference on Formal Methods in Computer-Aided Design (FMCAD'07). (Nov. 2007) 207–215
13. Mathworks: Matlab. http://www.mathworks.com.
14. Kropf, T.: Introduction to Formal Hardware Verification. Springer (1999)

15. Grabowski, D., Platte, D., Hedrich, L., Barke, E.: Time Constrained Verification of Analog Circuits using Model-Checking Algorithms. In: Proceedings of the First Workshop on Formal Verification of Analog Circuits (FAC'05). Volume 153 of 3. (Apr. 2005) 37–52
16. Steinhorst, S., Jesser, A., Hedrich, L.: Advanced Property Specification for Model Checking of Analog Systems. In: 9. ITG/GMM-Fachtagung, Entwicklung von Analogschaltungen mit CAE-Methoden (Analog'06). (Sept. 2006) 63–68
17. Maler, O., Pnueli, A.: Extending PSL for Analog Circuits. Technical report, Research Report, PROSYD Deliverable D 1.3/1 (2005)
18. Maler, O., Pnueli, A., Nickovic, D.: Checking Temporal Properties of Discrete, Timed and Continuous Behaviors. In: Pillars of Computer ScienceEssays, LNCS 4800, Springer. (Februar 2008)
19. Dong, Z., Zaki, M., Al-Sammane, G., Tahar, S., Bois, G.: Checking Properties of PLL Designs using Run-time Verification. In: Proceedings of Internatonal Conference on Microelectronics (ICM'07). (Dec. 2007) 125–128
20. Ruf, J., Kropf, T., Rosenstiel, W.: A Simulation Guided Property Checking Based on a Multi-Valued AR-Automata. In: Design, Automation, and Test in Europe (DATE'01). (Mar. 2001) 742–748
21. Weiss, R., Ruf, J., Kropf, T., Rosenstiel, W.: Efficient and Customizable Integration of Temporal Properties into SystemC. In: Forum on Specification and Design Languages (FDL'05). (Sept. 2005) 271–282