International Journal of Control and Automation
Vol. 9, No. 6 (2016), pp.97-104
http//dx.doi.org/10.14257/ijca.2016.9.6.09

FV-HS: Formal Verification for AMS SoC Based on Symbolic Computing

Jingbo Shao[1,3], Jing Wu[2] and Jingyu Liu[1*]

[1]*College of Computer Science and Information Engineering, Harbin Normal University, Harbin 150025, China*
[2]*College of Zhengzhou Normal School for Kindergarten, Zhengzhou 450099, China*
[3]*Postdoctoral Research Station of Information and Communication Engineering, Harbin Engineering University, Harbin 150001, China*
zro_bo@163.com

Abstract

This paper proposes a novel methodology for AMS SoC formal verification based on Hybrid Scheme combined with symbolic computing and LHPN model, FV-HS. The paper is concerned with a class of AMS designs, continuous-time AMS designs i.e., tunnel diode oscillator for research target. Firstly, Labeled Hybrid Petri Net model is established for safety property verification of tunnel diode oscillator, then mathematical expression for this model is extracted for efficiency enhancement, and then proof policy built in computer algebra Maple is applied to the corresponding LHPN model for tunnel diode oscillator to verify the property. The proposed method is implemented on tunnel diode oscillator and experiment results demonstrate the advantages of the proposed method over previous method. The proposed method overcomes the drawbacks of LHPN, makes full use of the merits of LHPN and symbolic computing, simplifies the workflow of algorithm and enhances the efficiency.

Keywords*: Symbolic computing, AMS SoC, formal verification, Labeled Hybrid Petri Net*

1. Introduction

Embedded components are becoming core in a growing range of electronic devices. Cornerstones of embedded systems are analog and mixed signal (AMS) designs, which are integrated circuits indispensible at the interface between electronic system and the real world environment. Analog circuit helps to secure the correct and steady operation of system. It follows that AMS design has become the necessity in people's daily life. The verification for AMS designs is concerned with the assurance of correct functionality, in addition to checking whether an AMS design is robust with respect to noises and current leakage.

Analog circuit helps to secure the correct and steady operation of system. It follows that AMS design has become the necessity in people's daily life. Therefore formal verification for AMS design is extremely important for SoC design and account for the most efforts and time of the IC designers. Moreover tight time-to-market and requirements for productivity improvement pose great challenges to tools development for formal verification of AMS SoC. AMS circuits can produce unexpected behaviors due to various reasons. Stemming from different process procedure, the faulty components differ from chip to chip. These AMS circuits are mostly used in safety-crucial systems such as communication vehicles, ABS system

* Corresponding author: liujingyu926@163.com

ISSN: 2005-4297 IJCA

etc. Research materials show that nowadays of 75% circuits are AMS circuits, while 50% faults occur on analog part [1]. Therefore it is increasingly important for designers to ensure the correctness of AMS circuit design. Formal verification is among the techniques to secure the correctness of AMS design.

Checkmate developed by CMU researchers is used to verify diode oscillator and Σ-Δ demodulator, and d/dt is applied to verify biquadratic lowpass filter [2-5]. Discrete analog transition structure (DATS) is proposed by Sebastian to accurately express optimization of nonlinear analog circuit. The accuracy of the proposed model is close to instant analysis with less number of states. The corresponding algorithm for formal verification of AMS SoC maps the partition into DATS. Such method is able to detect the errors hidden in circuit design [6]. Recently equation based geometry planning is more frequently applied to automatic synthesis for analog circuits and become widely adopted, and game between solution exactness and complexity helps to find equilibrium between them for optimal solution.

Labeled Hybrid Petri Net (LHPN) is developed for formal verification of AMS design [7]. LHPN outperforms pervious methods in that previous ones mostly require the designers to master hybrid automaton or hybrid Petri Net for AMS description, and they does not necessarily follow the rigid Hardware Description Language requirements; with LHPN, the designers can describe the AMS circuits of interest using their familiar language therefore LHPN becomes more popular and universal. LHPN can deal with heterogeneous components of AMS system, e.g. the analog parts and digital parts. However this approach suffers from the drawbacks of frequent model transformation, hence resulting in complicated work flow and requires a large amount of efforts of the designers.

In [8], a novel method of symbol computing is proposed for property verification of AMS design. The major idea is to verify AMS design using the same way as SAT, BDD based formal verification for digital system, to verify system automatically.

These above methods for formal verification of AMS SoC fall into two categories, *e.g.*, logic reasoning based formal verification and model checking based ones. Logic reasoning is time-consuming, the designers need to master rich mathematical experiences, while model checking usually meet with explosion of combination space, resulting in shortage in main memory space. In order to avoid the occurrence of the above problems, we develop labeled hybrid Petri Net method combined with proof policy embedded in Maple for AMS SoC verification, FV-HS. FV-HS takes good use of the merits of LHPN and overcomes its disadvantages and incorporates the proof policy of symbolic computing in Maple. Tunnel diode oscillator is taken for research objective, LHPN model based AMS SoC formal verification is established, and proof policy embedded in Maple is borrowed for further property verification of the LHPN model. Experimental results on benchmark are analyzed to show the advantages of the proposed method over previous one.

Generally, the verification is not complete because of limitations in time and memory. To overcome this problem, we observed that under certain conditions and for some classes of specification properties, the verification can be complete if we complement the LHPN model with other symbolic methods such as abstraction based method.

This paper describes a novel verification framework for the verification of AMS SoC that support intervals on the rates of change for the continuous variables. The proposed method begins with a model of the AMS SoC described using LHPN. VHDL-AMS allows the designers to use it to describe AMS SoC and the description is compiled into LHPN automatically. Then maple inbuilt proof policy is utilized accordingly for further property verification for tunnel diode oscillator. If the property is verified then the proposed algorithm terminates, otherwise a counter-example against the design model can be obtained, then the verification is complete and a failure is reported.

Tunnel diode oscillator is a widely used circuit; stability analysis has always been tough challenges for researchers and experts. A tunnel diode oscillator shown as Figure 1 is said to be saturate, if the input signal is bounded, and the output one remains bounded. The property is extremely essential because this nature might degrade circuit performance. Circuits using tunnel diode oscillator run the risk of saturating, which causes the circuit to be unstable. It is important to ensure that this never happens under any conditions. Therefore, the verification property for this circuit is whether or not the output voltage V_{out} can rise above 0.5 V or fall below 0 V. Aiming at the target of property verification for tunnel diode oscillator, this paper uses the same parameter as [6] to demonstrate that the circuit is never saturate, which actually belongs to safety property verification for hardware.

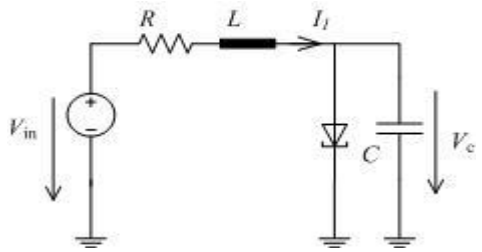


Figure 1. Illustration for Tunnel Diode Oscillator (V_{in}=0.3V, L =1μ h, C=1pf)

2. Preliminary Knowledge for LHPN

2.1. Basics for LHPN

In order to verify AMS SoC, it is necessary to develop a model which is capable of representing both Boolean (digital) behavior as well as continuous (analog) behavior. This paper, therefore, applies LHPNs for a VHDL-AMS compiler as well as a model generator from simulation data.

This section presents LHPN. An LHPN is a directed graph consisting of labeled transitions and the places[6], expressed in form of $N=<P,T,T_f,B,V,F,L,M_0,S_0,Q_0,R_0>$, with each component defined as follows:

1) P denotes a finite state set of places,

2) T denotes a finite set of transitions for discrete behavior,

3) $T_f \in T$ denotes a finite set of transitions for continuous behavior,

4) B denotes a finite set of Boolean signals,

5) V denotes a finite set of continuous variables,

6) $F \in (P \times T) \cup (T \times P)$ denotes the flow relationship between transitions and the places,

7) L is the labels defined as follows,

8) $M_0 \in P$ denotes the set of places of initially labeled,

9) $S_0:B \rightarrow \{0,1,X\}$denotes the initial values for each Boolean variable,

10) $Q_0:V \rightarrow \{Q \cup -\infty\} \times \{Q \cup \infty\}$denotes the range for each continuous variable,

11) $R_0:V \rightarrow \{Q \cup -\infty\} \times \{Q \cup \infty\}$denotes initial value range for changing rate of each continuous variable,

The major component of LHPN is label. We do not need changing rate of continuous variables, therefore the tuple defined in [6] is modified to delete label RA, hence labels in LHPN is expressed as $L=<En,D,BA,VA>$:

1) $En:T \rightarrow P$ is label of transition $t \in T$ with enabling condition,

2) $D:T \rightarrow |Q| \times (|Q| \cup \{\infty\})$ is label of transition $t \in T$ with upper bound and lower bound $[d_l(t),d_u(t)]$ on delay,

3) $BA: T \times B \rightarrow \{0,1,\mathbf{unc}\}$is label, when t occurs, Boolean value is assigned to Boolean variable,

4) $VA: T \times V \rightarrow (Q \times Q) \cup \{\mathbf{unc}\}$is the label for each transition $t \in T$ when it occurs, assignment for continuous variable $v \in V$ ranging from $[a_l(t,v), a_u(t,v)]$.

Changing rate for variables are not considered here, hence label *RA* is omitted. For the above expressions, *b* is Boolean signal, *v* is continuous variable, *k* is rational variable, **unc** denotes the value is not changed. These formulas satisfy the following grammar:

$$\phi ::= \mathbf{true} \mid \mathbf{false} \mid b \mid \neg\phi \mid \phi \wedge \phi \mid v \geq k$$

The behavior model for TDO is as follows:

$$h(v_c) = \begin{cases} 6.0105v_c^3 - 0.9917v_c^2 + 0.0545v_c & 0 \leq v_c \leq 0.055 \\ 0.0692v_c^3 - 0.0421v_c^2 + 0.0004v_c + 8.95794 * 10^{-4} & 0.055 \leq v_c \leq 0.35 \\ 0.2634v_c^3 - 0.2765v_c^2 + 0.0968v_c + 0.0112 & 0.35 \leq v_c \leq 0.50 \end{cases} \quad (1)$$

According to formula (1), the curve of volt-ampere characteristics is shown as Figure 2, where region II is the negative region.

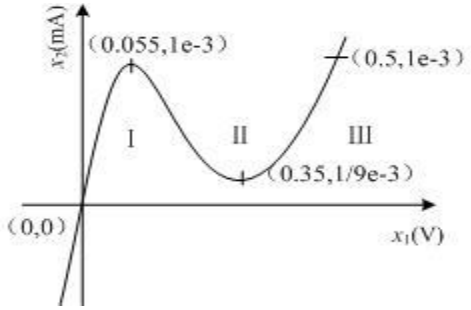
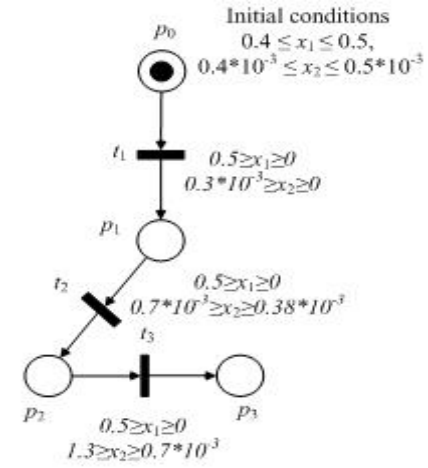


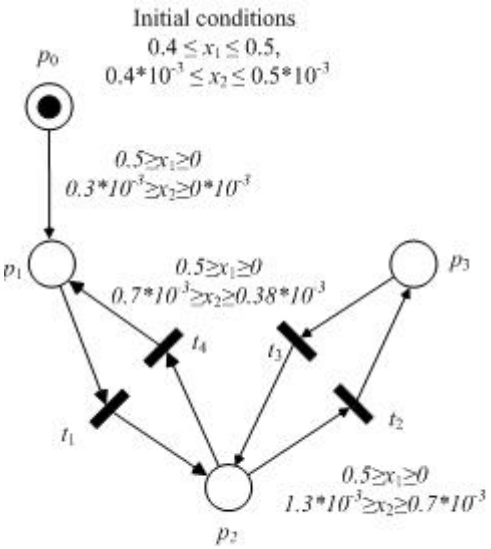
Figure 2. Curve of Volt-Ampere Characteristics for Tunnel Diode Oscillator

2.2. LHPN model for Tunnel Diode Oscillator

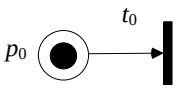
Behaviors of tunnel diode oscillator can be transformed into LHPN. Figure 3 shows an LHPN model for the tunnel diode oscillator. In Figure 3, the components of the LHPN are called states and transitions; the round circle with black tokens denote the initial markings; the round symbols denote states *P*; the rectangular symbols denote transitions *T*. They are interlinked by arrows which show the direction of flow. The black tokens will be removed by the occurrence of transitions from the preset the one transition and added to the post set of it.



a) LHPN Model for TDO in nonoscillating state



b) LHPN Model for TDO behavior in oscillating state



$\neg\{x_2 \ge 0.3*10^{-3}\}, x_2 \ge 0.7*10^{-3}$
c)LHPN model for TDO in nonoscillating state

Figure 3. LHPN Model for Tunnel Diode Oscillator

In Figure 3a, when the circuit is launched, the tunnel diode oscillator is experiencing transition from event t_1 to state p_1 from initial state p_0, then such token is removed from the preset for transition t_1, added to the post set for transition t_1, when system transit from event t_2 to state p_2, the token is removed from the preset of places for transition t_2, added to the postset of places for it, and system transits from event t_3 to state p_3, token is removed from preset of places and added to postset of t_3, and then remains in state p_3 therefore oscillating behavior cannot be accomplished. While in Figure 3b, starting from one state, tunnel diode oscillator transits from p_1, p_2 and p_3 sequentially to conduct behavior of oscillating.

Figure 2 shows that voltage ranges from [0, 0.5]v and current [0,1] mA. To verify if the circuit is in state of oscillating, the current should vary between [0.3, 0.7] mA. For LHPN model of TDO oscillating behavior, LHPN model only reasons for the three initial states, and it's easily found out that transition for fault in Fig. 3c is fired means the circuit might saturate. Our target is to demonstrate the circuit keeps oscillating periodically, *e.g.*, in state expressed as Figure 3b.

3. Symbolic Description for Tunnel Diode Oscillator

Figure 1 shows that a tunnel diode oscillator consists of a capacitor, an inductance and a tunnel diode. If the voltage or inductance that passes through the capacitor is nonzero, then the ideal LC oscillator will keep oscillating. Therefore the objective for verification is to verify if the given tunnel diode oscillator keeps oscillating under given initial conditions.

The differential equations for tunnel diode oscillator in Figure 2 are:

$$\frac{dV_c}{dt} = \frac{1}{C}(-hV_c + I_l)$$

$$\frac{dI_l}{dt} = \frac{1}{L}(-V_c - RI_l + V_{in})$$

$$dV_c = \frac{1}{C}(-hV_c + I_l)dt$$

$$dI_l = \frac{1}{L}(-V_c - RI_l + V_{in})dt$$

Transformations for the above equations are performed in order to transform the differential equations into recursive ones for tunnel diode oscillator.

$X(k+1) = B\,X(k) + A\,u(k)$

Thus we have the recursive equations:

$x_1(k+1) = c_1x_1(k) + b_1x_2(k) + a_1u(k)$

$x_2(k+1) = c_2\,x_1(k) + b_2x_2(k) + a_2u(k)$

Where A, B is matrix, parameters $u(k)$ is input, $V_c=x_1, I_l=x_2$. To verify whether the circuit oscillates or not, our task boils down to verify if the circuit satisfies conditions $0 \le x_1 \le 0.5$ and $0.3 \le x_2 \le 0.7$.

Therefore property P for tunnel diode oscillator can be expressed as:

$P(k) = \text{ForAll}(k \ge 0, \text{Cond}, 0.3 < x_2(k) < 0.7)$

$P(k) = \text{ForAll}(k \ge 0, \text{Cond}, 0 < x_1(k) < 0.5)$

4. Formal Verification for Tunnel Diode Oscillator

The last section transforms tunnel diode oscillator into LHPN model, to get rid of the drawback of complexity of the verification method; theoretically property verification for tunnel diode oscillator is to be carried out in terms of its LHPN model accordingly.

The in-built mathematical induction in Maple is adopted to prove that, under given conditions and parameters, $p(n)$ holds for all the natural number n. If for all $k>0$, the property holds then the property is verified, while if property holds for some values of k, we say the property is not verified, thus a counterexample is obtained.

Step one:

$P(t_0) = -1 < x_3(k) < 1$

Obviously the equation holds.

Step two:

$P(k)$ and $P(k+1)$ holds for all values of k is to be verified.

$P(k) = ForAll(k \ge 0, Cond, -1 < x_3(k) < 1)$

$P(k+1) = ForAll(k \ge 0, Cond, -1 < x_3(k) < 1)$

Thus we have:

$x_1(k+1) = if\,(b_1x_1(k) + u <> 0, x_1k) + b_1u - a_1a, x_1(k) + b_1u + a_1a)$

$x_2(k+1) = \text{if}\,(b_1x_1(k) + u <> 0, c_1x_1(k) + x_2(k) + b_2u(k) - a_2a, c_1x_1(k) + x_2(k) + b_2u(k) + a_2a)$

Property expressions after simulation are:

$P(k+1) = \text{if}\,(b_1x_1(k) + u <> 0,$

$0 < c_2x_2(k) + x_3(k) + b_3u(k) - a_3 < 0.5,$

$0.3 < c_2x_2(k) + x_3(k) + b_3u(k) + a_3 < 0.7)$

Thereby proves the stability of TDO theoretically.

5. Experimental Results

A differential equation discretization method similar to that proposed in [9] is utilized for the tunnel diode oscillator. In the model for the tunnel diode oscillator, 16 discrete regions are required to model the oscillatory and non-oscillatory behavior of the circuit. The property is verified for a range of initial conditions in which Il is between 0.45 and 0.55 mA and V_c is between 0.4 and 0.47V. As expected, the property verifies with R = 200 in 14.62 s after finding 17703 state sets, and the property does not verify with R = 242 in 0.34 s after finding 1826 state sets. We also attempted this verification using the

HyTech tool [10], but it is unable to complete due to arithmetic overflow errors. HyTech can complete analysis with less precision on the rates, but the model of the circuit no longer produces oscillation. Therefore, the verification results are incorrect. Our method also outperforms the PHAVer model checker on the diode oscillator which verifies it in 72.8 s [7]. This demonstrates that our method can provide a significant performance improvement over exact methods without loss in verification accuracy.

Table 1 shows the experimental results comparison among PHAVer[8], LHPN[6] and the proposed method. The parameters used for the experiment stems from [2]. The initial conditions are $I_l \in [0.4,0.5]$mA,$V_c \in [0.4,0.5]$v. Our verification aims at modeling for oscillating behavior and non-oscillating behavior of tunnel diode oscillator using 16-bit discrete regions under specific circuit parameters and initial conditions. When R=200Ω, LHPN model uses 14.62s for property verification. 17703 state sets are found in 14.62s when R=242Ω, 1826 state sets are found in 0.34s when R=242Ω. We tried to use tool HYTECH for the verification without success due to overflow errors. PHAVer model checking accomplish oscillating property checking for tunnel diode oscillator within 72.8s, however the proposed method only takes 10.39s, hence achieves higher efficiency than that in [6].

Table 1. Verification Results Comparisons on Tunnel Diode Oscillator

Approach	PHAVer[8]		LHPN[6]		Proposed	
Instance	Time(s)	OK?	Time(s)	OK?	Time(s)	OK?
TDO (Oscillating)	72.8	N/A	14.62 （R=200Ω）	Yes	10.39	Yes
TDO (non-oscillating)	n.r.	N/A	0.34 （R=242Ω）	No	0.30	No

Table 2 shows the verification results for the proposed method and LEMA. |Φ| represents the number of states. The proposed method consumes 14.32 seconds for formal verification, while LEMA uses 18.06 seconds. The proposed method verifies the oscillating behavior and non-oscillating behavior for tunnel diode oscillator. LEMA verifies the oscillating state for tunnel diode oscillator and fails to verify the non-oscillating behavior with longer CPU time. The experimental results demonstrate that the proposed method for formal verification of tunnel diode oscillator is promising.

Table 2. Verification Results Comparisons

Circuit	Proposed method			LEMA		
	|Φ|	Time(s)	Verifies?	|Φ|	Time(s)	Verifies?
TDO (oscillating)	17623	14.32	Yes	18524	18.06	Yes
TDO (Non-oscillating)	1804	0.48	Yes	1885	1.9	No

6. Conclusion

This paper describes a novel approach to formal verification for AMS SoC, FV-HS. tunnel diode oscillator is taken for instance for formal verification of AMS SoC. Firstly AMS SoC design is expressed as LHPN model transformed from its AMS-VHDL description, then mathematical induction embedded in Maple is applied for further verification of LHPN to overcome the disadvantages of complicated work flow, enhance the efficiency, and simplify the procedure of

formal verification. The proof for stability of tunnel diode oscillator has been conducted under given parameters, and the counterexample for property violation is also obtained. The proposed method has been applied to tunnel diode oscillator; it can be automatically integrated into the application flow and overcomes the limits of bounded time for exhaustive method. The proposed method achieves low complexity and high efficiency.

Acknowledgments

The author Jingbo Shao would like to express her gratitude to the helpful hints from her supervisor and the support from her colleague in College of Computer Science and Information Engineering, Provincial Key Lab and Center of Engineering Research and Development, Harbin Normal University. The research work is supported by Scientific Research Fund of Heilongjiang provincial Education Department (under grant No. 12531183).

References

[1] IBS Corporation. Industry reports, **(2003)**.
[2] S. Gupta, B. H.Krogh and R. A. Rutenbar, "Towards formal verification of analog designs", International Conference on Computer-Aided Design, **(2004)**, pp. 210-217.
[3] T. Dang, A.Donze and O. Maler, "Verification of analog and mixed-signal circuits using hybrid systems techniques", Formal Methods for Computer Aided Design, **(2004)**, pp. 21-36.
[4] S. Sebastian and H. Lars, "Trajectory-directed discrete state space modeling for formal verification of nonlinear analog circuits", IEEE/ACM International Conference on Computer-Aided Design, Digest of Technical Papers, **(2012)**, pp. 202- 209.
[5] A. K. Singh, M. Lok, K. Ragab, C. Caramanis and M. Orshansky, " An Algorithm for Exploiting Modeling Error Statistics to Enable Robust Analog Optimization", IEEE/ACM International Conference on Computer-Aided Design, Digest of Technical Papers, **(2010)**, pp. 62-69.
[6] G Al-Sammane, MH Zaki and S Tahar , "A Symbolic Methodology for the Verification of Analog and Mixed Signal Designs", Proceedings of the conference on Design, automation and test in Europe, **(2007)**, pp. 249-254.
[7] S. Little, D. Walter, C. Myers, R. Thacker, S. Batchu and T. Yoneda, "Verification of analog/mixed-signal circuits using labeled hybrid Petri nets", IEEE Transactions on computer-aided design of integrated circuits and systems, vol. 30, no.4, **(2011)**, pp. 617-630.
[8] G. Frehse, B. H .Krogh and R. A. Rutenbar, "Verifying analog oscillator circuits using forward/backward refinement", In Proc. Design, Automation and Test in Europe (DATE), IEEE Computer Society Press, **(2006)**, pp. 257-262.
[9] W. Hartong, L. Hedrich and E. Barke, "Model checking algorithms for analog verification", Proceedings of DAC, **(2002)**, pp. 542-547.
[10] T. A. Henzinger, "P. –Hho, and HWong-Toi. HyTech: a model checker for hybrid Systems", Software Tools for Technology Transfe r[J], no. 1, **(1997)**, pp. 110-122

Author

Jingbo Shao, she received the M. E. degree and Ph.D. degrees from Harbin Engineering University, Harbin, China, in 2007, and 2008, respectively. She is currently an associate professor with the Department computer science and information engineering, Harbin Normal University, Harbin, China. Her current research interests include computer aided design, VLSI design automation.

