Formal Verification of Analog Circuit Parameters Across Variation Utilizing SAT

Merritt Miller*, and Forrest Brewer†
Department of Electrical and Computer Engineering, UCSB
*merrittmiller@umail.ucsb.edu, †forrest@ece.ucsb.edu

[1] ***Abstract*—A fast technique for proving steady-state analog circuit operation constraints is described. Based on SAT, the technique is applicable to practical circuit design and modeling scenarios as it does not require algebraic device models. Despite the complexity of representing accurate transistor I/V characteristics, run-time and problem scaling behavior is excellent.**

***Index Terms*—Analog Verification, Discrete Representation, Circuit Modeling, SAT**

I. Introduction

Analog circuit verification has been a topic of rapidly increasing interest in recent years with verification strategies based on improving symbolic modular (SMT), interval analysis engines, as well as simulation mixed with other solution strategies[20], [8], [11], [5]. The vast majority of these techniques seek to create provable models that describe properties of the time evolution of analog circuits. Our work has a much more humble goal: quickly determining circuit steady-state operational bounds over transistor and environment variation. Classically, analog designers have relied on the tried and tested Monte-Carlo characterization of the operating space and its accelerated variants[16], [1] that depend on assumption on the statistics of the variations and the simplicity of the boundary topology. For complex circuits, however, or wide ranges of device variability, both assumptions are problematic, and the results are only stochastically conclusive.

The early work analog model checking was done by Kurshan[13], who formalized arguments for discrete proof spaces over continuous functions. Hartong[10] took a representation tack including modeling Ids(Vds, Vgs) from data as we have also done. Similar transistor modeling approches were used by Little, Meyers[14] and Yan[19], [18]. All of these works used specialized techniques to create discrete atomata for the purpose of forward projection of the analog circuit state. Hedrich did tolerancing verification based on polynomial bounding and projecting symbolically, however, his transistor models were linear approximations [11]. Work on variance estimation by Nassif [15] abstracted the sources of variance for timing and other parameters and performed stochastic modeling. Alternative simulation acceleration approaches by Signhee [16] improved the ability to simulate close to the bounds of circuit margins.

Validation by Simulation has advantages and difficulties. Simulation readily accepts circuit scales far larger than any verification scheme known to the authors. On the other hand, it does not readily link to formal analysis or property proof. To manage device and environmental variations with simulation, Monte-Carlo methods (or similar means of extracting statistics) are typically used, requiring a large number of configurations to be checked. This process tends to be time consuming. Additionally, due to its stochastic nature and the complexity of analog circuit behavior, such methods are usually incomplete. Abstractly, such methods verify behavioral properties for single parameter selections and initial conditions on each simulation run. In contrast, the methods proposed in this work describe properties known to hold over sets of intervals in the parameter space, providing formally supported bounds on circuit parameters.

Practical use of analog verification techniques requires methods that efficiently scale to typical problem sizes with accuracy sufficient to at least match that of the device models. They must provide concrete bounds on behavior regardless of whether the nominal behavior or statistical distribution can be compactly expressed. We believe that although the safety property is formally simple, it is nonetheless quite important in practical variance modeling and design for variance compensation. Its simplicity allows for practical scale robust verification in reasonable time.

This work describes an approach to proving properties on steady-state circuit behavior that scales to practical circuit size and complexity while using practical (non-symbolic) device models. The approach, by design, only utilizes methods that can easily be mapped to a boolean satisfiability (SAT) problem, a problem for which high quality solvers exist. Despite the limitation to steady-state properties, several valuable properties such as operation points and margins, headroom can be described and verified in very reasonable run times.

It is important to note that a similar effort was made by Tiwary et al.[17]. The general concept of modeling method and behavior discovery were the same, on the other hand a SMT solver was used instead of a ILP or SAT solver, and hence different assumptions about the bounds of the problem were made as well as having a slightly different formulation. This work also has device variation considered within the presented model.

II. Verification

This work targets the verification of steady-state properties of analog circuits. The verification process considers both a

[1]978-3-9815370-0-0/DATE13/©2013 EDAA

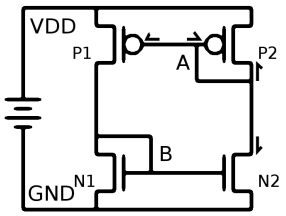

Figure 1. Simple circuit showing nodes (A, B, VDD, GND), devices (transistors p1, p2, n1, n2 and a voltage supply) and branch currents for node A

circuit description and a set of imposed constraints. A circuit is defined as a bi-partite graph of nodes and devices. All edges common to a node share a common potential. In general, voltages are defined as differences in potential between two nodes. All voltages are expressed as relative to a single ground (GND) node. The modeled behavior is time invariant (steady state), hence the circuit state is precisely the set of voltages on all circuit nodes. This representation implicitly enforces Kirchoff's voltage law (KVL), requireing the total voltage drop around a loop to be zero. There are additional constraints on nodes (enforcing current balance) and on devices which map voltages to currents that must also hold for a valid circuit model. In general, a circuit may have multiple or even infinite sets of states that meet all constraints, these states are called 'operating points'. Our interest in verification is to determine exterior bounds on node voltages for which no operating points exist given the circuit and imposed constraints.

An example analog circuit is shown in Fig. 1. In addition to containing the state of the circuit, nodes also serve connect currents for the devices. In the example circuit VDD, GND, A, and B are all nodes of interest and interact with at least two devices each. Kirchoff's current law (KCL) requires that nodes have currents balance into and out of the node. Fig. 1's node A has its currents marked by small arrows. The currents flowing in the direction of these arrows must sum to 0 for the circuit to represent a physical system. This gives a constraint on valid circuits – Eq. 1.

$$\forall n,\, n \in Nodes \sum_{i \in n's\, currents} i = 0 \qquad (1)$$

Devices are connected to nodes and direct currents between nodes. Fig. 1 has four devices N1, N2, P1,and P2 representing two types of transistors (NMOS & PMOS) as well as a source device. In simplest form, FET transistors have a single dominant current flowing between source (S) and drain (D); The current is a function of two voltages: the difference between the gate (G) and the source, and between the drain and the source. For FETs, ignoring the influence of the back , two differential voltages (Vds, Vgs) determine the current flow (Ids). While the methodology can fully support models of body bias, for simplicity this was left out of the models treated here.

Device modeling is a critical part of the process. A fully specified device in a design can have a range of different behaviors in implementation due to manufacturing variation as well as aging processes. This work uses a bounded behavior model; such a model is very general – agnostic to the nature or cause of device variation. This model can be built with just test data and no need for other known physical parameters Fig. 3 shows a slice of a transistor model that can be used for bounded behavior proofs. The model, by definition, captures the outer bounds of possible behavior for the device. Bounds that are conservative can still be used without invalidating the proof, mathematically expressed as Eq 2.

$$f_{lower}(\{nodes\}) \leq i_{device} \leq f_{upper}(\{nodes\}) \qquad (2)$$

The computed bounds on circuit behavior, as compared to monte-carlo simulation is depicted in Fig. 2 with lines representing computed bounds and points representing simulation. This is a result generated from the circuit shown in Fig. 1 over all potential variations in the FET device models. This simple bound will be used later in the paper to determine verifiable properties of interest to the user. The discrete solver implementation to efficiently and conservatively determine these bounds is discussed below.

III. Mapping to SAT

This work approaches verification using a discrete solver to handle the proof elements of the procedure. The end goal is to use a SAT solver in specific. To ease the translation of constraints the problem is first cast as a 0-1 ILP, also known as Psuedo-Boolean, problem (a known NP complete problem[4]). There are a number of translation techniques that will allow such a problem to be easily re-cast as a SAT problem[6], [2]. The high quality of existing SAT solvers is what makes them attractive for practical application. The solver used here is a version of MiniSat+, described in [6], modified to take SAT clauses in addition to Pseudo-Boolean constraints.

Node voltages are represented as fixed point binary numbers stored as an unsigned magnitude and a sign bit. The solver has a single scale factor for all voltages to be represented. The proof will be invalidated if values that are unrepresentable are needed creating a lower bound on the scale factor. A scale factor that is larger than needed will require more bits in representation to achieve the same precision as one that is sufficient yet smaller. The single discrete value of a voltage will be used to represent an entire range of voltages in the continuous version of the problem.Mathematically the range of possible values for a representation V is expressed in Eq.3, where n is the number of representation bits. The smallest difference between adjacent binary representations is called the Least Significant Bit Value or LSB for short.

$$voltage \in scale\, factor \times (V + [-0.5, 0.5]);\, V \in \mathbb{B}^n \qquad (3)$$

Device currents are handled in a distinct manner from the node voltages. Devices are specified by an upper and lower bounding function on currents given voltages. This means that a binary representation of a current can have exactly one value when mapped back into continuous space; the model bears the requirement of providing a range of possible values by

providing a lower and upper bound value; this constraint is expressed in Eq. 4.

$$repr(i_{lower}) < i_{lower} \leq i_{upper} < repr(i_{upper}) \quad (4)$$

The implementation of KCL in the system requires the current into and out of a given node to balance. This constraint is implemented by taking a sum of all of the currents into and out of a node and then constraining it to have a small absolute value. Two constraints, Eqs. 5 and 6, are created to implement this.

$$i_{node} = \sum_{devices} i_{device} \quad (5)$$

$$-\varepsilon < i_{node} < \varepsilon; \varepsilon = scale\,factor \times \#currents_{node} \quad (6)$$

Eq. 5 creates a requirement that each node have a list of currents going into and out of it. During circuit translation the software keeps such a list while implementing the constraints due to device models. As a final step the currents associated with the nodes are then summed to create this constraint.

Eq. 6 uses a small absolute value to prevent two specific cases from failing. One case arises for nodes where devices have no tolerance in their current representations (for example when the model is built expecting higher resolution than what the solver chooses to use). Another case occurs when a scaling factor within the system (eg $I_A = 2I_B$) is used and can cause a similar problem. Due to the discretization of circuit state and device models, a sum may have as much as half a bit of error (relative to the underlying model) per device current so an amount of slop equal to this potential error is allowed. Increasing the solver resolution reduces this tolerance relative to any absolute value presented, hence the resulting error can be made arbitrarily small.

The discretization of the state creates subsets of the continuous state space that contain potential operating points of the circuit. The conjunction of the above Pseudo-Boolean constraints cannot be satisfied for any discrete state whose patch does not contain an operating point. Thus the solver finds all states in the desired exterior bounds as unsatisfiable. Bounds on operation are found by iteratively looking for the largest satisfying value for any voltage of interest. By construction, these bounds can be determined to any level of accuracy by simply decreasing the granularity of the discrete space. In practice, FET models derived from measurements provide sensible limits to increasing accuracy of the analysis.

The circuit introduced in Fig. 1 has two nodes that are non-trivial to specify: A and B. In actual operation the voltage of these nodes is highly dependent on device models. Validation by simulation gives a set of operating points that is graphed in Fig. 2. This figure shows a region where the behavior of circuit A is likely to be and a large region where no simulation found operating points. For any interval or value of one parameter, the verification above can quickly find upper and lower voltage bounds on the other parameter providing an absolute limit to potential operating points since one or more circuit constraints must be violated. The value of this is that circuit properties are complex in general and there is no simple way to determine the practical bounds on operating regions (i.e. extrema of Monte-Carlo searches). This technique can find such bounds directly.

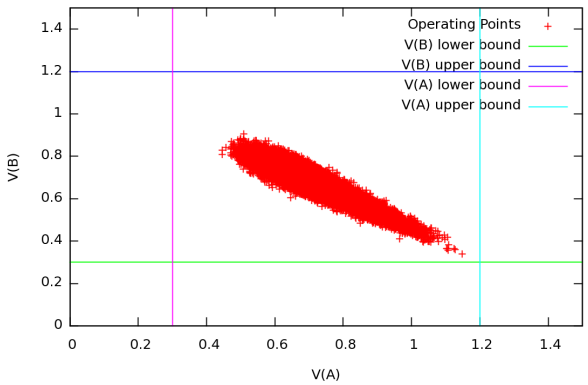

Figure 2. Map of verification results for circuit in Fig.1

IV. Device Models

Device models are the functions used to map nodal voltages into device currents. The models specifically map into a range of currents representing the possible upper and lower limits on device current for a given circuit state. The voltages at the nodes the device is attached to are the dependent variables in this mapping and the resultant device current is the independent variable. The device model must obey Eq.4 so the solver can correctly operate on the exterior bounds of circuit operation. The device model creates a solver variable, I_{device}, and places bounds on that variable. The bounds created may or may not be dependent on the voltages at the devices terminals.

A. Linear devices

Linear terms are easy to express in the pseudo-Boolean language making the modeling of linear devices easy. There are three kinds of linear device modeled in this work: voltage sources, current sources, and resistors. Each device adds simple constraints to the system. All three kinds of device have only two terminals. For two terminal devices there is one voltage of interest; the difference potential between the two nodes. Since each voltage number actually represents a range of voltages, this difference has an associated range of values. The magnitude of this range is the combined ranges of the two underlying voltages. To limit error a differential voltage is not computed directly, instead the software tracks the difference and incorporates it algebraically in to subsequent calculations.

1) Current sources: Current sources have a single systemic constraint; they must have a specified current flowing through them. This is modeled as two constants, a lower bound on current and an upper bound on current. This creates a simple device constraint defined in Eq. 7. The lower and upper bound of operation are constants given as part of the definition of the source.

$$I_{lower-bound} \leq I_{device} \leq I_{upper-bound} \quad (7)$$

2) Voltage sources: Voltages sources also have a single constraint; they limit the difference in voltages between their terminals. The voltage source requires the differential voltage constraint in Eq. **??** and adds a constraint expressed in Eq. 8. Ideal voltage sources provide whatever current is required to keep the voltage between its terminals as expected. The current flow through the device is effectively unbound. Because of the KCL constraint, the source has to have a unbound current specified, so the system also adds Eq. 9 to the systemic constraints.

$$V_{lower-bound} \leq V_{diff} \leq V_{upper-bound} \tag{8}$$

$$-I_{max} \leq I_{device} \leq I_{max} \tag{9}$$

3) Resistors: Resistors have a current that is proportional to the voltage difference between their two terminals. Like the current source they add a relation to current but now the upper and lower bounds are dependent on the terminal voltages. The resistor is specified as having a minimal and maximal resistance R_{min} and R_{max} respectively. The computed voltage difference becomes a constraint expressed in Eq. **??**; the resistor model is implemented as described in Eq. 10.

$$R_{min} \times V_{diff} \leq I_{device} \leq R_{max} \times V_{diff} \tag{10}$$

B. Non-linear devices

There is one type of non-linear device of interest for this work: the transistor. Mapping non-linear devices in 0-1 ILP requires multiple linear constraints to create a non-linear one. Due to the nature of 0-1 ILP (and ILP in general) polytopes are easy to express; the models used are expressed as a union of polytopes.While the solver can understand models of this generality creating them efficiently is difficult. Generating meshes with bounded error has a number of heuristics [7], [12]. For simplicity, this work chooses all edges to be parallel to one of the variables defining the space. Because of the underlying SAT solver used, the union is written as a series of constraint cubes, one of which must be obeyed. This can be visualized in the model shown in Fig. 3. This figure shows the continuous space boundaries of behavior and the step like polytope approximation.

Models of this type have proven useful in other applications. For example, sub-division of continuous space is a useful method for solving for time domain transitions[9], [10]. Piecewise transistor models have also been used to make computation easier in simulators[3]. Union of polygons was also used in the interval based solver in [11]

1) Spice/Monte-Carlo Extraction based models: The first set of transistor models this work considers were based on an underlying data set created by Monte-Carlo simulation preformed in spice. These models are based on a .13μ Technology. The bounding region was determined by the central 99.9% of 30,000 Monte-Carlo runs performed in spice. The monte-carlo runs took approximately 15 minutes of computer time to execute and a further minute or so to reduce to an acceptable model. A number of models were built from this data set. The main PMOS and NMOS models were created

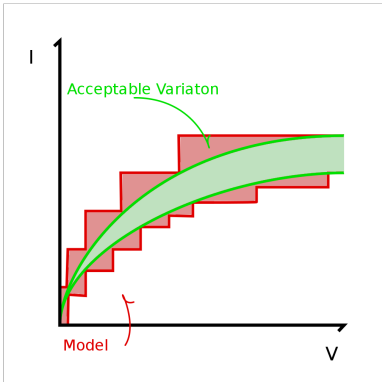

Figure 3. Transistor model, showing natural gutter of acceptable behavior and conservative discrete union of polytope model

with 50mV resolution in Vgs and Vds and a 100nA resolution in Ids. This yielded models that consists of 230 and 188 cubes in Vgs,Vds,I. A version limited to 40 cubes in each using a 150mV step was also created for faster approximations. Another much higher resolution model set with more than 10,000 cubes per transistor was created for scalability testing. This more accurate model required much more data; it was built with an expanded 100,000 point run, and still had insufficient data in some scattered regions. The high cube count model could be improved with a larger data set.

2) ASU PTM Corner-case transistor curve based model set: The models generated for the examples in this paper were created form the ASU PTM models from their nano-spice tool. 3σcorner models were downloaded for the 130nm process node and considered the outside bounds of operation for the transistors in question. The devices were crafted with 5% variation in L_{eff} and 15mV in V_{th} variation. These models were subdivided using regular steps then these regions were merged as to maintain an error less than a given threshold a model using 20mV steps in V_{GS} and V_{DS} and 2% accuracy in I_{DS}. This gives a model with approximately 2,000 cubes for each of NMOS and PMOS. Because of the smooth and simple nature of the PTM model re-casting these models is easy by comparison to the Monte-Carlo method.

C. Model impacts on problem complexity

The design of optimal models is beyond the scope of this work but remains important. A device model is needed for each device in the circuit. Clauses that represent the model are added to the constraint set for each device instance that uses that model. Thus the SAT problem complexity will roughly grow linearly with the number of polytopes used in the device models.

V. Applied Cases

A. Resistor divider – output voltage

The resistor divider is a very basic case to test some of the concepts of the system. As a basic test a divider is constructed using two resistors each $1k\Omega \pm 5\%$ with a voltage source of 1.5V. The solver is asked to find the minimum and maximum exterior bounds for the possible values of the output voltage.

Source	Monte #1	Monte#2	ASU PTM
Vgs steps	125mV	50mV	20mV
Vds steps	375mV	Acc. Dependent	
I_{dev} accuracy	Fixed Vds	10%	2%
NMOS cubes	40	188	1936
PMOS cubes	40	230	1936
Runtime (amp.sp)	1.7s	16.2s	397s
#Vars	4402	21062	190210
#Constraints	5090	26510	243986

Figure 4. Table of models. The amp.sp test has 14 transistors and finds 8 bounds and is discussed more fully in Sec. V-B

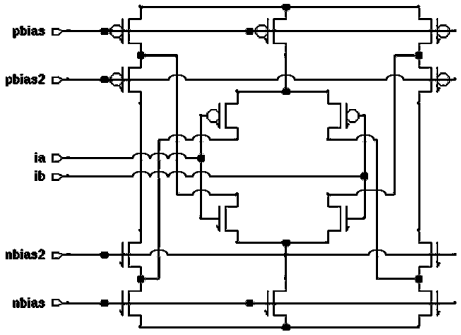

Figure 5. Differential amplifier with multiple bias points

For this example case the solver quickly gives the answer: the output value is bound on the upper side by 0.7875V and the lower side by 0.7125V. Due to the finite nature of the solver these values will be rounded to the nearest least significant bit of the representation. The maximum value is rounded up and the minimum is rounded down because the exterior bound is being discovered, if the actual bound cannot be represented a safer approximation is used.

B. Differential amplifier – requirements on bias voltages to support output current

The differential amplifier test is a more complicated case. The circuit shown in figure5 is translated into a spice file along with some designer constraints. first a output drive requirement:$|V(on) - V(op)| < .05$; $I_{load}(on, op) = 50\mu A$. Next a limit on the required input to create that drive:$|V(ia) - V(ib)| < .1$. Finally a restriction that the answer must be close to centered relative to the supply:$.45 \times V(vdd) < V(ia) < .55 \times V(vdd)$

The more complex constraints added allow for setting up the exterior conditions on the amp. In this case the delivered load is $50\mu A$ the inputs are within 5% of mid-way between the rails and with an offset less than .1v. The bias voltages deemed viable are the ranges of bias voltages that make the circuit work given the presented conditions. In other words bounds placed on pbias are imply that pbias for a working circuit would never be outside that range.

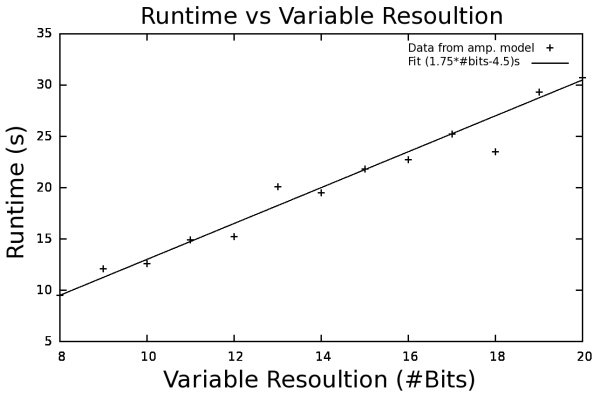

Figure 6. Time to solve for 4 bias voltages of circuit from Fig. 5 vs number of bits used to represent a voltage or current

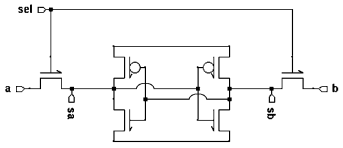

Figure 7. Sram cell

Given the higher transistor complexity of this amplifier it makes it ripe for a study in how bit-width impacts operational speed of the solver. Fig. 6 shows the time dependence of solution on number of bits used in the representation. The growth is close to linear and an approximate fit is shown.

C. SRAM characterization

SRAM cells, due to the large number that are produced, are a common area of study for failure. Because of the multiple operating states of an SRAM cell there are steps to analysis, so that individual states can be isolated and studied. Additional constraints are added to the system to cut the space into cases. One case will be established to discover bounds on the metastable region, and another is constructed to find the bounds of the stable states.

1) Meta-stable region: The first operating point of the SRAM cell that we can discover is the region where the meta-stable point exists. The meta-stable region is the region where there is neither strong drive out of the region, nor any attraction to the region. If we assume that for a given cell there is reasonable transistor matching there is an easy condition for meta-stability: when the two storage voltages are equal (sa and sb in the schematic in Fig 7). The additional constraint is written formally as $V(sa) = V(sb)$.

With the models described above the solver determines that this meta-stable state is bounded within the region of $.6v \leq sa, sb \leq .8v$

2) Stable state: The next operating mode of the cell to be considered is the behavior when the cell stores a value. In this case one of the voltage storage nodes is above the other. Due to the nature of the transistor models used the meta stable region can also satisfy this requirement, so the answer requires

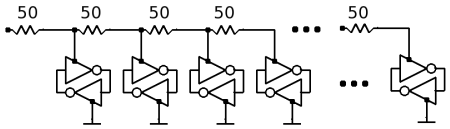

Figure 8. Schematic of SRAM array test. 10 SRAM cells used as part of this test

the exclusion of the meta stable region. Due to the symmetry of the problem the constraint $V(sa) > .8\,; V(sb) < .6$ is sufficient for this exclusion

The results of this run indicate that $1.4v \leq sa \leq 1.5v$ and $0v \leq sb \leq .1v$. This means that the high storage value can be as low as 1.4v and the low storage voltage can be as high as .1v given the model and other circuit assumptions

D. SRAM array

The SRAM array test consists of an array of SRAM cells connected by a power network that has resistive elements, and hence a voltage drop that is state dependent. The cells used are the same as the cells from Sec. V-C; from that analysis the voltage required to force the cells out of the meta stable region is known (not between .6V&.8V when the supply is 1.5V). The network topology used is shown in Fig. 8. The successive drops means that each cell is operating at a different voltage than the other cells; which should impact the behavior. The array test seeks to verify the properties of an SRAM cell that is the 10th in the chain. The test thus has 20 NMOS transistors, 20 PMOS transistors and 10 resistors with non-trivial relation to the result.

The test exercises the limits of the ability of the solver. With the medium complexity transistor model the problem contains 75k Pseudo Boolean constraints. The run takes approximately 20 minutes to verify that, for the last cell, the voltage drop is significant enough that the final SRAM cell is incapable of reliably storing a bit (the model predicts that cells that will always store one value or another are possible outcomes). This is consistent with the single cell test since the voltage droop is approximated to be down to .9v in worst case, allowing the lower edge of meta-stable to crash into the low-stable state.

VI. Conclusions

This work presents a practical method of verifying safety in steady state. Models agnostic to the physics of a device are used to produce bounds on total circuit behavior. The method presented uses proven and stable solvers to render rapid solution to problems containing many devices with reasonable scaling with problem complexity. Proof resource and time scaling with problem size was roughly linear showing the potential for large circuit utility. A problem was noted with the scaling of device models was noted and future work may readily improve computation time by reducing model complexity required for a given level of accuracy.

References

[1] K.J. Antreich, H.E. Graeb, and C.U. Wieser. Circuit analysis and optimization driven by worst-case distances. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 13(1):57 – 71, jan 1994.

[2] O. Bailleux, Y. Boufkhad, O. Roussel, et al. A translation of pseudo boolean constraints to sat. *Journal on Satisfiability, Boolean Modeling and Computation*, 2:191–200, 2006.

[3] Min Chen, Wei Zhao, Frank Liu, and Yu Cao. Fast statistical circuit analysis with finite-point based transistor model. In *Proceedings of the conference on Design, automation and test in Europe*, DATE '07, pages 1391–1396, San Jose, CA, USA, 2007. EDA Consortium.

[4] Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, STOC '71, pages 151–158, New York, NY, USA, 1971. ACM.

[5] W. Daems, G. Gielen, and W. Sansen. Simulation-based generation of posynomial performance models for the sizing of analog integrated circuits. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 22(5):517 – 534, may 2003.

[6] N. Eén and N. Sörensson. Translating pseudo-boolean constraints into sat. *Journal on Satisfiability, Boolean Modeling and Computation*, 2(3-4):1–25, 2006.

[7] Michael Garland and Paul S. Heckbert. Surface simplification using quadric error metrics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '97, pages 209–216, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[8] G.G.E. Gielen and R.A. Rutenbar. Computer-aided design of analog and mixed-signal integrated circuits. *Proceedings of the IEEE*, 88(12):1825 –1854, dec. 2000.

[9] S. Gupta, B. H. Krogh, and R. A. Rutenbar. Towards formal verification of analog designs. In *Proceedings of the 2004 IEEE/ACM International conference on Computer-aided design*, ICCAD '04, pages 210–217, Washington, DC, USA, 2004. IEEE Computer Society.

[10] W. Hartong, L. Hedrich, and E. Barke. Model checking algorithms for analog verification. In *Design Automation Conference, 2002. Proceedings. 39th*, pages 542 – 547, 2002.

[11] L. Hedrich and E. Barke. A formal approach to verification of linear analog circuits wth parameter tolerances. In *Proceedings of the conference on Design, automation and test in Europe*, DATE '98, pages 649–655, Washington, DC, USA, 1998. IEEE Computer Society.

[12] A.D. Kalvin and R.H. Taylor. Superfaces: polygonal mesh simplification with bounded error. *Computer Graphics and Applications, IEEE*, 16(3):64 –77, may 1996.

[13] R.P. Kurshan and K.L. McMillan. Analysis of digital circuits through symbolic reduction. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 10(11):1356 –1371, nov 1991.

[14] S. Little, N. Seegmiller, D. Walter, C. Myers, and T. Yoneda. Verification of analog/mixed-signal circuits using labeled hybrid petri nets. In *Computer-Aided Design, 2006. ICCAD '06. IEEE/ACM International Conference on*, pages 275 –282, nov. 2006.

[15] S.R. Nassif. Modeling and analysis of manufacturing variations. In *Custom Integrated Circuits, 2001, IEEE Conference on.*, pages 223 – 228, 2001.

[16] Amith Singhee and Rob A. Rutenbar. From finance to flip flops: A study of fast quasi-monte carlo methods from computational finance applied to statistical circuit analysis. In *Quality Electronic Design, 2007. ISQED '07. 8th International Symposium on*, pages 685 –692, march 2007.

[17] S.K. Tiwary, A. Gupta, J.R. Phillips, C. Pinello, and R. Zlatanovici. First steps towards sat-based formal analog verification. In *Computer-Aided Design - Digest of Technical Papers, 2009. ICCAD 2009. IEEE/ACM International Conference on*, pages 1 –8, nov. 2009.

[18] Chao Yan and M.R. Greenstreet. Verifying an arbiter circuit. In *Formal Methods in Computer-Aided Design, 2008. FMCAD '08*, pages 1 –9, nov. 2008.

[19] Chao Yan, F. Ouchet, L. Fesquet, and K. Morin-Allory. Formal verification of c-element circuits. In *Asynchronous Circuits and Systems (ASYNC), 2011 17th IEEE International Symposium on*, pages 55 –64, april 2011.

[20] Mohamed H. Zaki, Sofiï¿œne Tahar, and Guy Bois. Formal verification of analog and mixed signal designs: A survey. *Microelectronics Journal*, 39(12):1395 – 1404, 2008.