Summary of Research Results and Software Packages

Research results of NeoCAD COSMOS project are summarized in subjects below.

1. MCAST – Model Compiler based on Abstract Syntax Tree

MCAST (Model Compiler based on Abstract Syntax Tree), is a UW developed EDA tool to facilitate the development and implementation of advanced device models. It achieves this goal by compiling device models from high-level behavioral language VHDL-AMS descriptions to target simulators (SPICE, SPECTRE, Leader etc) automatically, and efficiently.

More specifically, with MCAST, model development would starts with mathematical description of the device physics in high-level behavioral language VHDL-AMS; then the description would be automatically compiled by MCAST into C / Fortran code with optimized execution efficiency; finally, the C code will be linked either statically or dynamically with target simulators. The traditionally most error prone and time-consuming part of model development is coding in C or Fortran. With MCAST, this burden on model developers can be completely removed without the sacrifice of model accuracy and model evaluation efficiency.

As a result, the application of MCAST will significantly shorten the time and lower the cost of device model development, and subsequently lead to more robust and diversified device models. Beyond device model development, MCAST could be used for Mixed Signal, and multi-physical domain modeling and simulation, which would especially benefit the circuit designers from system level exploration to verification as well.

The simulators supported by MCAST are SPICE3f5 (UC Berkeley), Leader (IBM), and SPECTRE (Cadence). It also generates MATLAB code for debugging purpose.

The MCAST technology has been applied to thermal effect modeling, circuit-electromagnetic effects modeling, lumped-distributed systems modeling, partial differential equation modeling, BPSK transceiver system modeling, and others.

The following organizations attended our MCAST workshop in November 2003: IBM, Motorola, Boeing, and University of Washington.

The MCAST Technology has been transferred to: MIT Lincoln Laboratory, University of Arkansas.

Members participated in this project are: Bo Wan, Bo Hu, Lili Zhou, Eric Nomad, Zhao Li, Cherry Wakayama, Chris Baker, Dr. Pavel Nikitin.

2. IPRAIL – Intellectual Property Reuse-based Analog IC Layout

Layout properties related to device matching and symmetry, parasitics, current density in interconnects, thermal, and substrate parasitics greatly affect analog/RF design performance. The complexity involved in modeling these layout-effects for incorporation into layout-automation engines is immense. Thus, traditionally, analog/RF layouts have been crafted manually by expert layout-designers to squeeze-in the desired performance through intricate layout-geometries. IPRAIL presents a reuse-centric approach to analog/RF layout automation.

Starting from an existing layout, IPRAIL first extracts a set of constraints and then optimizes these constraints to generate the target layout. Thus, the "good" properties of the existing layout are retained in generating new layouts. This is especially useful for generating new layouts during process migration and design spins for different performance specification.

We have proposed and implemented a set of key techniques and framework for constraint generation and layout optimization and some novel methods for detection of circuit matching and layout symmetry. We have investigated techniques for handling RF-specific layout issues including handling of large number of vias/contacts and well-characterized passive devices. We have introduced new layout dependent substrate parasitic models and presented the key techniques for constraint reduction that enables retargeting of large analog designs. We also have developed a new method for optimizing transistors layouts during retargeting.

IPRAIL has been used to retarget different classes of analog/RF layouts like: operational amplifiers, comparators, voltage-controlled oscillators and analog-to-digital converters. Layouts that used to take several weeks for manual drawing can be generated in a few minutes.

Overall, this reuse-based layout automation technique has proved to be of great significance in reducing analog/RF design cycle time. Current research is directed towards interconnect sizing/spacing for optimization of circuit performance. This entails parasitic extraction and development of new algorithms for layout optimization with non-linear constraints.

Members participated in this project are: Sambuddha Bhattacharya, Nuttorn Jangkrajarng, Roy Hartono, Chris Baker.

3. FROSTY – A Fast Hierarchy Extractor for Industrial CMOS Circuit

Circuit recognition and extraction is a very important task in VLSI CAD field. This type of tool is widely used in many commercial CAD tools for post-layout simulation, layout function verification, formal verification and design for test.

FROSTY is an automatic CMOS circuit recognition and extraction tool. It reads in two files – file1 and file2, where file1 is the description of an object circuit and file2 defines some higher-level digital blocks, such as DFF, Latch, adder, etc. FROSTY automatically recognize all the standard CMOS gates, such as INV, NAND2, AOI12, etc. in the file1. FROSTY automatically checks whether there are instances of the digital blocks in file1. Finally, FROSTY outputs a Verilog format RTL level netlist and a header file which contains the functional definitions of all used standard CMOS gates, the two files can be simulated in any digital simulators.

Member participated in this project is: Lei Yang.

4. Fast Linear/Nonlinear Solver

Three different types of fast time-domain simulation methods have been developed, which are designed to speedup parasitic-sensitive VLSI circuit simulation. Typically, parasitic-sensitive circuits consist of large linear networks, i.e., power/ground networks, substrate, interconnects, etc., and relatively small nonlinear circuits. Efficient simulation of such circuits presents a complexity challenge to SPICE-like circuit simulators. With our proposed methods, we have shown orders of magnitude speedup over SPICE3 on power/ground network and substrate examples.

SILCA (Semi-implicit Iterative Linear Centric Analysis): SILCA is based on two linear centric ideas to reduce the number of LU factorizations during time-domain circuit simulation. Three techniques have been implemented in SILCA: 1) a semi-implicit iterative integration scheme, 2) a successive variable chord method, and 3) a piecewise weakly nonlinear definition of MOSFET models. SILCA is developed based on the SPICE3 open source code.

New progresses on solver research include *a Coupled Iterative/Direct Method* and an *Efficiently Preconditioned Iterative Method*. The former exploits the advantages of different types of solvers, while the latter makes use of the previously factorized L and U matrices as the preconditioner. Both methods have demonstrated improved speed for solving large-scale problems.

Member participated in this project is: Zhao Li.

5. Model Order Reduction and Symbolic Analysis

The goal of model order reduction is to significantly improve simulation speed without losing much accuracy by replacing a high-order model (slow) by an approximate low-order model (fast). Model order reduction techniques are especially useful for system-level design and verification. It has been recognized as an important behavioral modeling tool in EDA industry.

We have investigated several new model order reduction techniques that can be applied to parametric models, nonlinear models, and large-scale interconnect/parasitics models. We also have studied numerical computation methods that can greatly improve model reduction speed and accuracy.

Symbolic approach to behavioral modeling is another important part of modeling research. We have investigated a new symbolic network analysis technique that is potentially applicable to large analog networks. We have established a new tree enumeration method that is conceptually much simpler than those in the literature and implemented it in software.

Members participated in this project are: Dr. Guoyong Shi, Bo Hu.

List of Publications and Submissions

Model Compiler and Applications

[1] B. Wan, B. Hu, L. Zhou and C-J.R. Shi, "MCAST: An Abstract-Syntax-Tree Based Model Compiler for Circuit Simulation," Custom Integrated Circuits Conference, 2003. Proceedings of the IEEE 2003 , 21-24 Sept. 2003, Pages:249 - 252.

[2] L. Zhou, B. Hu, B. Wan and C.-J. R. Shi, "Rapid BSIM Model Implementation with VHDL-AMS/Verilog-AMS and MCAST Compact Model Compiler", SOC Conference, 2003. Proceedings. IEEE International [Systems-on-Chip] , 17-20 Sept. 2003 Pages:285 - 286.

[3] B. Wan and C-J.R.Shi, "Hierarchical multi-dimensional table lookup for model compiler based circuit simulation", Design, Automation and Test in Europe Conference and Exhibition, 2004. Proceedings, Volume: 2 , 16-20 Feb. 2004.

[4] P. V. Nikitin, V. Jandhyala, D. White, N. Champagne, J. Rockway Jr., C.-J. R. Shi, C. Yang, Y. Wang, G. Ouyang, R. Sharpe, and J. Rockway Sr., "Modeling mixed circuit-electromagnetic effects in electronic design flow", IEEE ISQED conference, San Jose, March 2004

[5] P. Nikitin, E. Normark, and R. Shi, "Distributed electrothermal modeling in VHDL-AMS", IEEE BMAS conference, Oct. 2003, San Jose, CA

[6] P. Nikitin, W. Yam, and R. Shi, "Parametric equivalent circuit extraction for VLSI structures", IFIP International VLSI-SoC Conference, Dec. 2003, Darmstadt, Germany

[7] P. Nikitin, C.-J. R. Shi, and B. Wan, "Modeling partial differential equations in VHDL-AMS", IEEE Systems-on-Chip Conference, Portland, OR, Sept. 2003, pages:345 – 348.

[8] E. Normark, L. Yang, C. Wakayama, P. Nikitin and C-J. R. Shi, "VHDL-AMS Behavioral Modeling of a π/4 DQPSK Transceiver System", Submitted to IEEE International Behavioral Modeling and Simulation Conference, 2004.

Layout Automation

[9] N. Jangkrajarng, S. Bhattacharya, R. Hartono and C-J. R. Shi, "Automatic Analog Layout Retargeting for New Processes and Device Sizes", *Proc. ISCAS*, May 2003, pp. 704-707.

[10] N. Jangkrajarng, S. Bhattacharya, R. Hartono and C-J. R. Shi, "IPRAIL – Intellectual Property Reuse-based Analog IC Layout Automation", *Integration – The VLSI Journal*, vol. 36, pp. 237-262, Nov 2003.

[11] S. Bhattacharya, N. Jangkrajarng, R. Hartono, C-J. R. Shi, "Hierarchical Extraction and Verification of Symmetry Constraints for Analog Layout Automation", *Proc. ASPDAC*, Jan 2004, pp. 400-405.

[12] N. Jangkrajarng, S. Bhattacharya, R. Hartono and C-J. R. Shi, "Multiple Specification Radio-Frequency Integrated Circuit Design with Automatic Template Driven Layout Retargeting", *Proc. ASPDAC*, Jan. 2004, pp. 394-399.

[13] Z. Li, R. Suravarapu. R. Hartono, S. Bhattacharya, K. Mayaram, and R. Shi, "CrtSmile: A CAD Tool for CMOS RF Transistor Substrate Modeling Incorporating Layout Effects", *Proc. ASPDAC*, Jan. 2004, pp. 163-168.

[14] S. Bhattacharya, N. Jangkrajarng, R. Hartono, C-J. R. Shi, "Correct-by-Construction Layout-Centric Retargeting of Large Analog Designs", *Proc. DAC*, Jun 2004, pp. 139-144.

[15] S. Bhattacharya, N. Jangkrajarng, R. Hartono, C-J. R. Shi, "Challenges and Techniques for Layout Automation of Radio-Frequency Integrated Circuits", Invited Paper for *Asia Pacific Microwave Conference*, Dec. 2004.

[16] R. Hartono, N. Jangkrajarng, S. Bhattacharya and C-J. R. Shi, "Automatic Device Layout Generation for Analog Layout Retargeting" Accepted for *The International Conference on VLSI Design*, Jan. 2005.

Fast Linear/Nonlinear Solver

[18] Zhao Li and C.-J. Richard Shi, "A Coupled Iterative/Direct Method for Efficient Time-Domain Simulation of Nonlinear Circuits with Power/Ground Networks", IEEE Int. Symp. on Circuits and Systems, May 2004.

[19] Zhao Li and C.-J. Richard Shi, "SILCA: Fast-Yet-Accurate Time-Domain Simulation of VLSI Circuits with Strong Parasitic Coupling Effects", IEEE/ACM International Conference on Computer-Aided Design, pp. 793-799, Nov. 2003.

Layout Extraction and FPGA Application

[18] L. Yang and C-J. R. Shi, "FROSTY – A Fast Hierarchy Extractor for Industrial CMOS Circuits", Submitted to Integration – The VLSI Journal.

[19] L. Yang and C-J. R. Shi, "FROSTY – A Fast Hierarchy Extractor for Industrial CMOS Circuits", IEEE International Conference on Computer-Aided Design, Nov 2003, San Jose, CA.

[20] L. Yang, M. Shen, C-J. R. Shi and H. Liu, "An FPGA Implementation of Low-Density Parity-Check Decoder with Multi-Rate Capacity", Submitted to Proceedings of Asia and South-Pacific Design Automation Conference, 2004.

Symbolic Analysis

[21] A. Manthe, Z. Li, and R. Shi, "Symbolic analysis of analog circuits with hard nonlinearity", IEEE Design Automation Conference, June 2003

[22] G. Shi, and C.-J.R. Shi, "A new topological approach to symbolic network analysis," Submitted to IEEE Trans. Circuit Systems Part I, 2004 (in review).

Model Order Reduction

[23] B. Hu, G. Shi, and C.-J.R. Shi, "Symbolic model order reduction," in Proc. IEEE International Workshop on Behavioral Modeling and Simulation (BMAS), pp. 34 - 40, 2003.

[24] G. Shi and C.-J.R. Shi, "Parametric reduced order modeling for interconnect analysis," Proc. Asia South-Pacific Design Automation Conference (ASP-DAC), Yokohama, Japan, January 2004, pp. 774 - 779.

[25] G. Shi and C.-J.R. Shi, "Model order reduction by dominant subspace projection: error bound, subspace computation and circuit application," UWEE Technical Report, No. UWEETR-2003-0021, 2003. Also Submitted to IEEE Trans. Circuit Systems Part I (revised).

Model Compiler (MCAST) Report

Executive Summary

MCAST (Model Compiler based on Abstract Syntax Tree), is a UW developed EDA tool to facilitate the development and implementation of advanced device models. It achieves this goal by compiling device models from high-level behavioral language VHDL-AMS descriptions to target simulators (SPICE, SPECTRE, Leader etc) automatically, and efficiently.

More specifically, with MCAST, model development would starts with mathematical description of the device physics in high-level behavioral language VHDL-AMS; then the description would be automatically compiled by MCAST into C / Fortran code with optimized execution efficiency; finally, the C code will be linked either statically or dynamically with target simulators. The traditionally most error prone and time-consuming part of model development is coding in C or Fortran. With MCAST, this burden on model developers can be completely removed without the sacrifice of model accuracy and model evaluation efficiency.

As a result, the application of MCAST will significantly shorten the time and lower the cost of device model development, and subsequently lead to more robust and diversified device models. Beyond device model development, MCAST could be used for Mixed Signal, and multi-physical domain modeling and simulation, which would especially benefit the circuit designers from system level exploration to verification as well.

The simulators that MCAST supports are SPICE3f5 (UC Berkeley), Leader (IBM), and SPECTRE (Cadence). It also generates MATLAB code for debugging purpose.

Publication

[1] B. Wan, B. Hu, L. Zhou and C-J.R. Shi, "MCAST: An Abstract-Syntax-Tree Based Model Compiler for Circuit Simulation," Custom Integrated Circuits Conference, 2003. Proceedings of the IEEE 2003 , 21-24 Sept. 2003, Pages:249 - 252.

[2] L. Zhou, B. Hu, B. Wan and C.-J. R. Shi, "Rapid BSIM Model Implementation with VHDL-AMS/Verilog-AMS and MCAST Compact Model Compiler", SOC Conference, 2003. Proceedings. IEEE International [Systems-on-Chip] , 17-20 Sept. 2003 Pages:285 - 286.

[3] B. Wan and C-J.R.Shi, "Hierarchical multi-dimensional table lookup for model compiler based circuit simulation", Design, Automation and Test in Europe Conference and Exhibition, 2004. Proceedings, Volume: 2 , 16-20 Feb. 2004.

[4] P.V. Nikitin, C.R. Shi, B. Wan. , "Modeling partial differential equations in VHDL-AMS [mixed signal systems applications]";SOC Conference, 2003. Proceedings. IEEE International [Systems-on-Chip] , 17-20 Sept. 2003 Pages:345 – 348 .

MCAST workshop participating companies, Universities

IBM,
Motorola,
Boeing,
University of Washington

Technology Transferring

MIT Lincoln Laboratory
University of Arkansas

Students trained

Bo Wan, Michael Greaves, Eric Nomad, Zhao Li, Bo Hu, Lili Zhou, Cherry Wakayama, Chris Baker, Tao Long, Hsiu-Chi Hsu

MCAST: An Abstract-Syntax-Tree based Model Compiler for Circuit Simulation*

Bo Wan, Bo P. Hu, Lili Zhou, and C. -J. Richard Shi
Department of Electrical Engineering
University of Washington, Seattle, WA 98195

Abstract: This paper introduces *MCAST*: a *M*odel *C*ompiler---based on *A*bstract *S*yntax *T*rees---that reads compact device models described in high-level languages VHDL-AMS/Verilog-AMS and automatically generates the simulator device code in C that can be directly linked with existing circuit simulators such as SPICE3. We report, for the first time, the successful implementation of industry-grade device models, including EKV, BSIM, and BSIM-SOI, in VHDL-AMS/Verilog-AMS. For a set of industry test circuits, MCAST yields exactly the same simulation results as, and comparable speed to, that of model code implemented manually, while existing model compilers are either limited in scope, restricted to very simple models, or orders of magnitude slower than manual implementations.

1. Introduction

Circuit simulators such as SPICE [7] are the corner stone of modern VLSI design methodologies. The use of such simulators requires device models to be built in the simulators. Unfortunately, the effort to implement a new device model into a simulator is tedious, error prone, and requires a deep understanding of underlying simulator code. As the result, it takes on average one to two years for a new device model to become available in a commercial circuit simulator for circuit designers to use after it is first developed by a modeling engineer. Due to implementation considerations, a device model is only considered *completely defined* after its implementation in a simulator, which means it is either not, or hard to be, accessible and maintainable by model developers.

A potential solution to this problem is a compact model device compiler. A model compiler can read compact device models described using high-level design languages such as VHDL-AMS or Verilog-AMS, and generate automatically the device simulator code that can be linked with a circuit simulator such as SPICE. Since only behavior-related device equations need to be described, such a model development and qualification process takes only at most days to weeks. Furthermore, models will be highly maintainable and reusable. A GUI for generating VHDL-AMS code from equations can also be used by model developer if needed [11].

* This research was supported by DARPA NeoCAD Program under Grant No. N66001-01-8920 and NSF CAREER Award under Grant No. 9985507.

Previous model compiler attempts include ADMIT [2], iSMILE [3], MAST/Saber[9], ADMS [4], [5] and [6]. The principal difficulty of wide acceptance of model compilers is the performance of generated code. It is known to be 10 to 1000 times slower than manual implementation even for MOS Level 1 model and simple circuits. The speed further deteriorates as the complexity of a model and the size of a circuit increase [10]. In MCAST, several optimization techniques are implemented based on the construction of AST. This leads to strong improvements on the efficiency of the generated codes.

2. MCAST Foundation and Architecture

The MCAST foundation is to represent the device model written in VHDL-AMS using AST. For example, below is a MOS Level 1 device model written in VHDL-AMS.

```
Vgstmp := Vg - Vs;
Vdstmp := Vd - Vs;
Vgdtmp := Vgstmp - Vdstmp;
IF Vdstmp >= 0 THEN
        Forward := 1;           -- forward mode
        Vds := Vdstmp;
        Vgs := Vgstmp;
ELSE
        Forward := -1;          -- reverse mode
        Vds := - Vdstmp;
        Vgs := Vgdtmp;
END IF;

IF Vgs <= Vth THEN      -- cut off
   Idstmp := 0.0;
ELSE
   IF Vgs-Vth <= Vds THEN -- saturation
      Idstmp := Beta * (((Vgs -Vth)**2)/2);
   ELSE -- triode
      Idstmp := Beta*((Vgs-Vth)*Vds
                         -((Vds**2)/2));
   END IF;
END IF;
Ids := Forward * Idstmp;
```

The AST representation is shown in Figure 1. The root of the tree is the variable Ids, where leaf nodes can be constants or terminal voltages. Different from traditional AST used in computer science, we introduce a new type of *Switch (SW)* node to represent the widely used *if-else-endif* structure in VHDL-AMS.

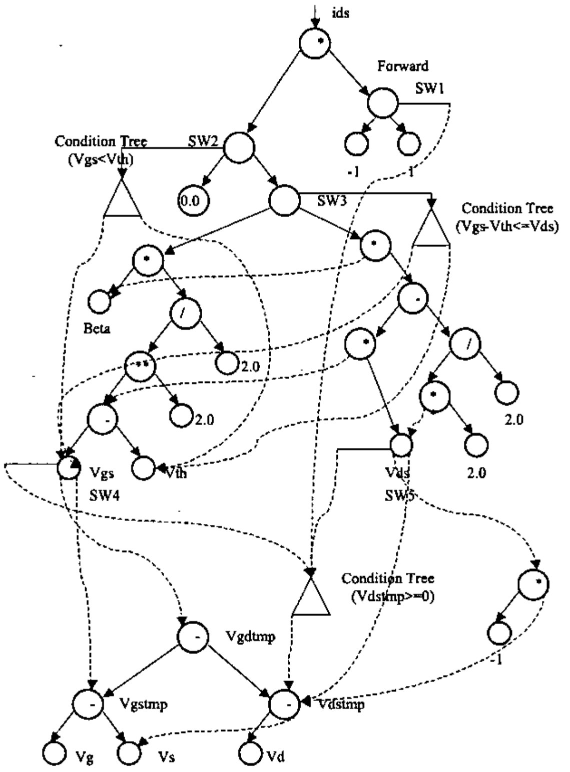


Figure 1. An AST example for MOS Level 1 model.

The architecture of MCAST is depicted in Figure 2. It starts from the VHDL-AMS description file of a device model. MCAST first parses the input file, checks errors and stores the device information in an intermediate format structure. Then the AST tree representation of device models and the needed derivatives are constructed, and derivatives are generated by automatic differentiation. Next, techniques are used to optimize the AST for both device equations and their derivatives. Finally, device codes that include device definition, device setup, device loading, derivative calculation and matrix element stamping, interfaces to the target simulator, truncation error checking, and convergence limiting are generated from the optimized AST.

3. AST-Driven Code Optimization

The success of a model compiler depends critically on the efficiency and robustness of the generated device code, in particular, the portion of code responsible for filling in the Jacobian matrices and the right-hand-side vectors. This so-called *element stamping* consists of device model evaluation, equivalent conductance and equivalent current source evaluation (derivative derivations). High fidelity device models such as BSIM can involve potentially hundreds of parameters, variables and intermediate variables, and thousands of lines of equations. Our effort has been focused primarily on how to generate *element stamping code* that uses minimum amount of computations over the entire simulation run.

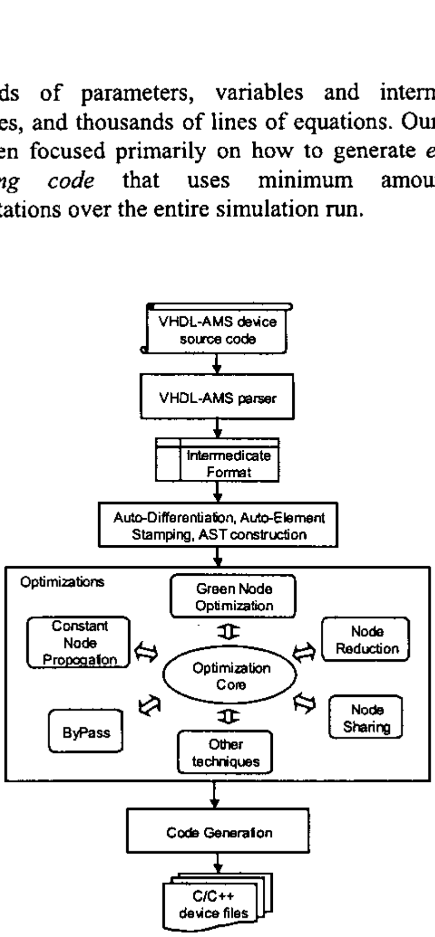


Figure 2. MCAST model compiler architecture.

A. Green Node Optimization: A straightforward implementation of element stamping, as done in most existing model compilers, is to use automatic differentiation to generate the code for element stamping and incorporate the code directly for solving systems of linear equations at every nonlinear iteration of each time point. This is equivalent to evaluating the AST at every iteration. In contrast, MCAST colors the AST into three colors: *green* (the node needed to be evaluated only once over the entire simulation), *purple* (the code needed to evaluated once at each time point, and *red* (the code needed to be evaluated at every nonlinear iteration of each time point). Special cases of green nodes include model/instance parameter calculation and range checking. This can be substantial for models like BSIM that have hundreds of parameters. Examples of purple nodes include those calculating performances such as power consumption (multiplying currents and voltages at every time point). Red nodes include examples such as those device equations whose terminal voltages changed at each iteration. With AST, MCAST identifies automatically green nodes by a bottom up traversal of the AST to check on how each node depends on leaf nodes (constants, parameters, or variables), and purple nodes by a top down traversal of the AST to check if a node is required at every iteration or only once at each time point.

We note that exploring this computational latency *manually* was instrumental to the success of SPICE over general-purpose numerical simulators. MAST/Saber from Analogy [9] uses specific language constructs or compiler derivatives for a model developer to indicate parameter checking or performance calculation. This still is a huge burden on model developers, and makes the model description less readable. Further, only a limited amount of optimization can be achieved. As the result, MAST/Saber has achieved a limited success for semiconductor circuits.

B. Bypass: MCAST employs automatic node bypass and device bypass. MCAST uses Automatic Differentiation in the generation of the Jacobian Matrix elements associated with the device. Automatic differentiation may generate large amount of intermediate dummy nodes that are just associated with the dumb operations with +0, -0, *0, *1,*-1. These nodes can be bypassed or compressed in code generation. Device bypass is a well-known method first implemented in SPICE2 [7]. It offers a reduction by allowing previously calculated results to be used again for current iteration, when the terminal voltages/currents of the device of current iteration have not changed over a limit from its previous value (often, this limit is set empirically). Otherwise, this device would have to be re-evaluated. MCAST incorporated device bypass automatically.

C. Constant Propagation: Model designers often define some frequently used constants for the new device in VHDL-AMS file, such as kTq, CONSTvt0, etc in BSIM3, to make the VHDL-AMS source file easier to understand and maintain, they also may define some new constants based on those already defined constants. These constants are necessary for the readability of the source VHDL-AMS code. MCAST can detect those constants and replace them by values during code generation.

D. Node Reduction: If some nodes do not affect the element stamping, they are redundant, and can be removed from the AST. This case often occurs at the early stage of model evaluation.

E. Node Sharing: MCAST uses pattern matching to find the duplicate sub-expression trees inside an AST. Different from classical ASTs where such duplicates are always shared, MCAST AST has conditional Switch (SW) nodes, and such duplicates can be shared only if their parent nodes have the same or similar conditions. MCAST categories these duplicates based on their conditions and computational costs, and determines if they are to be shared or not.
We note that green node optimization and bypass are specific to model compilers and are for the first time automated in MCAST. The other three techniques are well known in compiler theory. However, with automatic differentiation for derivative calculation used in model compilers, these features as in VHDL-AMS/Verilog-AMS will not be able to be recognized by the C compilers from the generated C code from model compilers.

4. Experimental Results

Several device models, including MOSFET level 1, level 3, BSIM3, BSIMSOI, EKV, Thermal-Electrical (heating resistor) and Opto-Electrical (laser diode) device models have been implemented with MCAST, and linked with Berkeley SPICE3, and tested on thirteen benchmark circuits. For all the circuits, the simulation using the compiled models generated by MCAST from VHDL-AMS yields the same results as that implemented manually in Berkeley SPICE3. As an example, the simulation results of a voltage-controlled oscillator are shown in Figure 3: the two curves match perfectly.

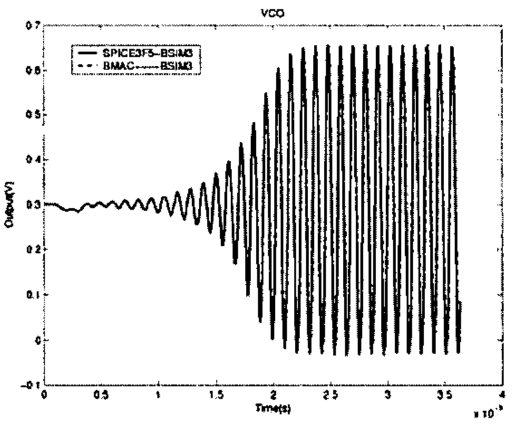


Figure 3. The simulation results of VCO using the MCAST-compiled BSIM3 model and the manually implemented BSIM3 model.

MCAST generated device codes are linked to SPICE3 source code to compare with human optimized codes (existing built-in device model codes in SPICE3). Figure 4 shows the speed comparison on benchmark circuits for MOSFET level 3 and BSIM. MCAST Level 3 model code is 10%-100% faster than the hand codes (except for one circuit), MCAST BSIM code is less than 70% slower than human optimized BSIM code.

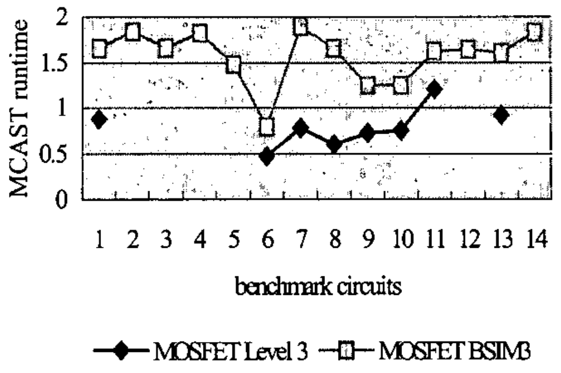


Figure 4. The speed ratios of MCAST-compiled vs manually optimized device codes.

We compare SPICE3 integrated with MCAST generated BSIM3 model with the best commercially available VHDL-AMS/Verilog-AMS simulator** on adders with increasing number of bits. The results are shown in Figure 5. Our model compiler with AST-driven optimization is two to three orders of magnitude faster than the commercial behavioral simulator. Furthermore, MCAST technology scales linearly with the size of a circuit, where the cost of the commercial simulator increases exponentially. This demonstrates that the MCAST technology is especially suitable for handling large circuits.

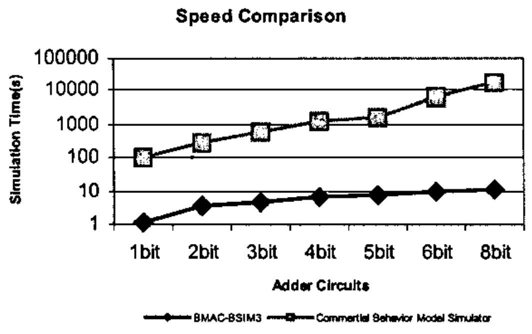


Figure 5. Speed comparison of MCAST-BSIM3 and a commercial behavior model simulator.

Figure 6 shows the speedup breakdowns of different optimization techniques over thirteen benchmark circuits, in terms of device evaluation time per iteration. The overall speed up is about 4 to 5 times, where green node optimization achieves a speed up of about 3.5, and an average 50% for other techniques.

** We attempted all the existing commercial behavioral simulators, and the one used here is from a leading vendor and has the best performance comparing to other simulators.

5. Conclusions

We presented MCAST---a model compiler that can automatically compile compact device models in high-level modeling language VHDL-AMS into the simulator code such as SPICE. Several industry-grade device models including EKV, BSIM, and BSIM-SOI have been implemented using MCAST. Simulation of a set of industry circuits has shown that MCAST has the same accuracy and comparable performance as human optimized device code. Further, MCAST scales linearly with the size of circuits. In addition, two mixed-technology device models (thermo-electrical and opto-electrical) were successfully implemented.

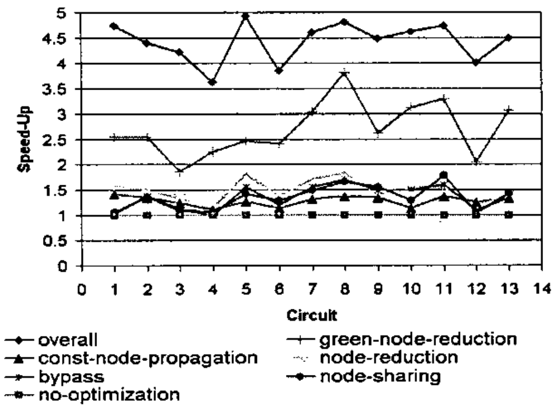


Figure 6. The speedup breakdown of various optimization techniques.

References

[1] Ken Kundert, "Automatic Model Compilation – An Idea Whose Time Has Come", *The Designer's Guide*, May 2002.

[2] S. Liu, K.C.Hsu, P.Subramaniam, "ADMIT-ADVICE Modeling Interface Tool", *IEEE Custom Integrated Circuits Conference*, 1988

[3] A.T.Yang, and S.M.Kang, "iSMILE: A Novel Circuit Simulation Program with emphasis on New Device Model Development", *26th Design Automation Conference*, 1989

[4] Laurant Lemaitre, Colin McAndrew, and Steve Hamm, "ADMS-Automatic Device Model Synthesizer", *IEEE Custom Integrated Circuits Conference*, May 2002

[5] R. V. H. Booth, "An Extensible Compact Model Description Language and Compiler", *Proc. IEEE BMAS*, pp. 39-44, Oct. 2001.

[6] M. Zorzi, N. Speciale, G. Masetti, "Automatic Embedding of a Ferro-electric Capacitor Model in Eldo", *Proc. IEEE BMAS*, Oct. 2001

[7] L. W. Nagel, "SPICE2 – A computer program to simulate semiconductor circuits, " Univ. of California, Berkeley, ERL Memo ERL-M520, May 1975.

[8] Y. Cheng and C. Hu, *MOSFET Modeling & BSIM3 User's Guide*, Kluwer Academic Publisher, 1999

[9] MAST/Saber User Manual, Analogy Inc.

[10] Hal Carter, "Modeling and Simulating Semiconductor Devices Using VHDL-AMS", BMAS 2000

[11] H. A. Mantooth, http://mixedsignal.eleg.uark.edu/paragon.html

#250: Rapid BSIM Model Implementation with VHDL-AMS/Verilog-AMS and MCAST Compact Model Compiler

Lili Zhou, Bo P. Hu, Bo Wan, and C. -J. Richard Shi
Department of Electrical Engineering
University of Washington, Seattle WA 98195

Abstract

VHDL-AMS and Verilog-AMS are behavioral languages extended from widely used VHDL/Verilog for the analog and mixed-signal applications. By describing models in behavioral language VHDL-AMS/Verilog-AMS and then compiling them with model compiler such as MCAST, we have successfully implemented the industry-grade device models (including BSIM, BSIMSOI) rapidly and with low cost, which was previously a tough and high cost work. The implementation for the first time demonstrated the capability and advantages of this new method compared to the traditional methods in device modeling.

1. Introduction

Device modeling is the foundation in circuit design and simulation. The traditional method of translating the device model equations to C/Fortran code requires a lot of inter-stages work and a large amount of expertise in software engineering. Such kind of work is tedious, error-prone and very time-consuming. Considering the device model may need to be updated after some important physical effects have been identified, the heavy burden and high cost of maintaining models prevent many innovative new models from being accepted into commercial simulators [1].

Using the new method of device modeling in VHDL-AMS/Verilog-AMS with model compiler, the above problem can be tackled in an efficient and robust way. We can easily translate the model equations from the model document into high-level description VHDL-AMS code, and use the model compiler MCAST [6] to generate C code that can be easily merged into target simulators. The entire process can take less than two hours.

To demonstrate the practical significance of this promising methodology, it must have the capability to handle industry grade device models. We have successfully implemented the most complex device models, such as BSIM3/BSIMSOI, in both VHDL-AMS and Verilog-AMS. With the model complier MCAST (currently only supports VHDL-AMS) or other Verilog-AMS model compiler, we obtain the automatically generated C code that can be directly compiled into circuit simulators such as SPICE3. The simulation results verified that our behavioral models have the same accuracy as the original BSIM models.

2. Device Model Implementation based on Model Compiler Methodology

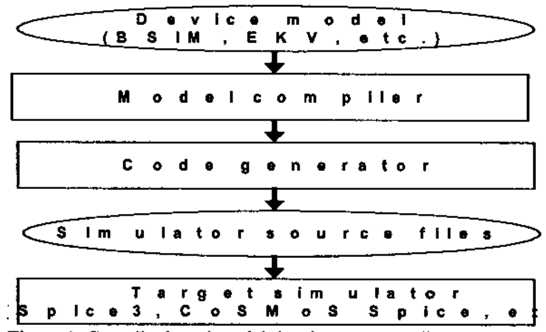


Figure 1. Compiler based model development paradigm.

Fig. 1 shows the flow of new device model development based on model compiler MCAST. It starts from the user input which is a VHDL-AMS file describing a device model. Model compiler parses the information and stores it in an intermediate format [6]. During the code generation, multiple device source codes are produced according to different target simulators. These codes will be compiled and linked with the source files of a target simulator to create a new simulator with the new device models. Using this new simulator allows circuit designers to simulate a circuit consisting of new device models.

3. Compact Device Modeling in Behavioral Languages VHDL-AMS/Verilog-AMS

VHDL/Verilog has been used extensively in the design and verification of digital systems since introduced in 1980s'[2]. The recent extensions to VHDL-AMS/Verilog-AMS greatly enhanced their capability to support the hierarchical description and simulation of continuous and mixed-continuous/discrete systems with conservative and nonconservative semantics [2]. A typical VHDL-AMS model has an **entity** with one or more **architectures**. The entity includes the description of the ports of the model and the definition of its generic parameters. The architecture part contains the detailed implementation of the model. Verilog-AMS has some similar properties as VHDL-AMS. The analog behavior described in **analog** module and the behavioral descriptions are mathematical mappings, which relate the input and signals in terms of a large signal or time-domain behavioral description.

4. Experimental Results

Twelve benchmark circuits from different sources were used to test the accuracy and stability of our new device models both in VHDL-AMS and Verilog-AMS. Table 1 shows the statistics of the models in different levels of VHDL-AMS code, where the source code of MCAST and SPICE3 only include code of setup, parameter-calculation and load part.

Table 1. Device Models

Device Model	# Nodes	# Parameters	Complexity (#line of VHDL-AMS code)	MCAST generated codes	SPICE3f5 source codes
MOS Level 1	4	2	40	901	1837
MOS Level 3	4	121	870	3768	2322
BSIM3	4	412	2228	10777	6637
BSIMSOI	6	787	2502	12801	9974

Fig. 2 shows the output waveforms of one test bench with our BSIM3 device models in VHDL-AMS compared to the original SPICE3. The example is an industry-grade class-E power amplifier design used by NeoCAD, which provides high current drive to a load. This example is also tested using the BSIMSOI device models, and MCAST is capable of capturing the same accuracy as SPICE3. The two curves are overlapped with numerical difference in the figure. For Verilog-AMS code, because our complier MCAST currently only supports VHDL-AMS, we only test the Verilog-AMS code model in the Cadence environment. Fig. 3 shows the comparison of CPU time of our BSIM3 model compiled by MCAST with SPICE3. The curves show the simulation times of the behavioral device model compiled by MCAST is about 70% as fast as hand optimized SPICE3 and even in some cases faster.

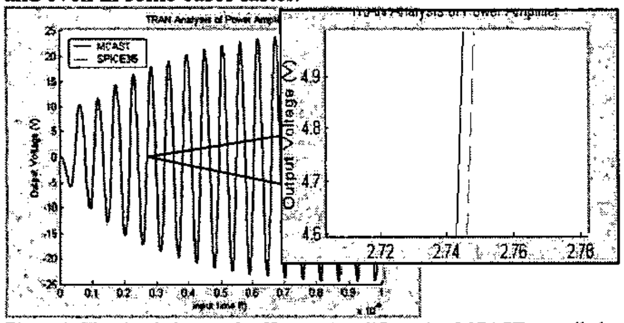


Figure 2. The simulation result of Power Amplifier using MCAST-compiled BSIM3 model and the manually implemented BSIM3 model

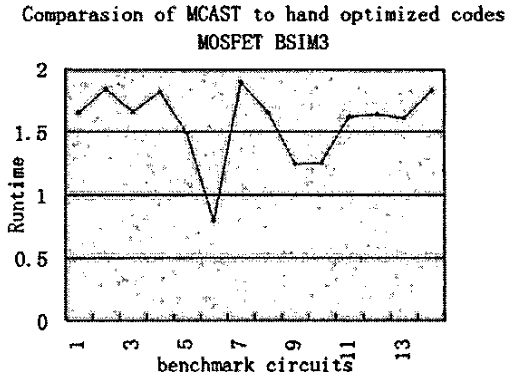


Figure 3. The comparison of CPU time of our BSIM3 model compiled by MCAST with SPICE3.

From these results, we can see that our new models match the manually implemented models accurately. For the sensitive RF circuits such as VCO, Power Amplifier and real industry designs such as Boeing's Comparator, our models are stable, accurate as the manually implemented models but with much lower cost.

5. Conclusions

In the paper, we presented device modeling based on behavioral language VHDL-AMS/Verilog-AMS. We for the first time successfully implemented the industry grade MOS device model BSIM3 and BSIMSOI rapidly and with low cost. And the experimental data shows the accuracy and practical significance of our models and the model compiler MCAST as well. It is quite promising that such kind of methodology could become a very important way for future model developers to shorten their model development cycle and to deliver better device models.

References

[1] Ken Kundert, "Automatic Model Compilation – An Idea Whose Time Has Come", *The Designer's Guide*, May 2002.

[2] Ernst Christen, Member, IEEE, and Kenneth Bakalar, "VHDL-AMS—A Hardware Description Language for Analog and Mixed-Signal Applications" , IEEE Transactions on Circuits and Systems—II: Analog and digital signal Processing, VOL. 46, NO. 10, October 1999.

[3] Dan, Fitzpatrick and Ira miller, "Analog behavioral modeling with the Verilog-A language", Kluwer Academic Publisher

[4] http://www-device.eecs.berkeley.edu/~bsim3/

[5] Y. Cheng and C. Hu, *MOSFET Modeling & BSIM3 User's Guide*, Kluwer Academic Publisher, 1999

[6] Bo Wan, Bo Hu, Lili Zhou, "MCAST: An Abstract-Syntax-Tree based Model Compiler for Circuit Simulation", CICC, 2003

Hierarchical Multi-Dimensional Table Lookup for Model Compiler based Circuit Simulation*

Bo Wan and C.-J. Richard Shi
Department of Electrical Engineering, University of Washington
{wanbo,cjshi}@ee.Washington.edu

Abstract— In this paper, a systematic method for automatically generating hierarchical multi-dimensional table lookup models for compact device and behavioral models with any number of terminals is presented. The method is based on an Abstract Syntax Tree representation of analytic equations. Expensive part of the computations represented by abstract syntax trees are identified and replaced by two-dimensional table lookup models. An error-control based optimization algorithm is developed to generate table lookup models with the minimal amount of table data for a given accuracy requirement. The proposed method has been implemented in the model compiler MCAST and the circuit simulator SPICE3. Experimental results show that, compared to non-optimized compilation based simulation, the simulation using the proposed table lookup optimization method is about 40 times faster and achieves sufficiently accurate results with error less than 1-2%.

Index Terms— Model Compiler, Abstract-Syntax-Tree, Hierarchical Multi-dimensional Table Lookup, Optimization, Circuit Simulation.

I. INTRODUCTION

Manually implementing a compact device model into a circuit simulator is becoming increasingly difficult. It takes on average one to two years for a new device model to become available to circuit designers in a commercial circuit simulator after it is first developed by model developers [1]. This sets a big barrier between model developers and circuit designers; on one hand, a lot of new models are created each year but only a small portion of them are implemented, while on the other hand, the need of using new models is increasing.

In modern deep sub-micron designs, many new effects such as leakage currents need to be considered, which may not be captured in a previous developed device model. Therefore, circuit designers would like to have more freedom to modify device models to meet their specific requirements. Unfortunately, currently there is no convenient way for circuit designers to add the specific effects into their circuit simulator. They have to wait for simulator vendors to take action.

Several compact device model compilers are emerging as a solution for this problem [2][3][4][5][6]. With a model compiler, designers can describe models in high level behavioral languages such as VHDL-AMS or Verilog-A(MS), and then compile automatically to a target simulator. The process for model development and qualification is therefore greatly shortened.

However, a major bottleneck for the mainstream use of model compiler technologies is that the efficiency of automatically generated model is not as good as of manually written device model. It has been shown in [7] that it can be typically 10 to 1000 times slower even for MOS Level 1 model and simple circuits due to the high evaluation cost of automatically generated model. The speed further deteriorates as the complexity of a model and the size of a circuit increase.

To improve the simulation efficiency of automatically generated models, optimization technologies in the process of model compilation become crucial. Some techniques have been reported in [2], which are compiler based and do not trade off between the accuracy and the speed. Results in [2] show that the efficiency can be close to that of manually written codes.

In this paper, we present a systematic method to automatically generate hierarchical multi-dimensional table lookup models for devices with any number of terminals and any set of equations. Table lookup is an attractive way to speed up the simulation by trading off memory and a little bit of accuracy. It has been applied to the simulation of MOSFET transistors [8][9][10][11][12] before. However, all the previous efforts were ad hoc, and designed specifically for a particular device with particular set of equations (MOSFETS in most cases). No works report using table lookup for general device models with any set of equations and any number of terminals (for example, BSIMSOI has six terminals), as required in model-compiler based circuit and behavioral simulation.

This paper details a systematic table lookup method and its implementation in the MCAST model compiler to generate accurate hierarchical multi-dimensional table lookup models for analytical compact devices. In particular, we describe in Sections II and III the use of Abstract Syntax Tree to build table lookup hierarchy and a table lookup algorithm. An error-control based method for table sizing is presented in Section IV. Section V describes test results with our implementation on MOSFE Level 3 model and a set of benchmark circuits.

II. ABSTRACT SYNTAX TREE REPRESENTATION

A compact device model compiler can read compact device models described using high-level behavioral languages such

* This research was supported by DARPA NeoCAD program under Grant No. N66001-01-8920 and NSF CAREER Award under Grant No. 9985507.

as VHDL-AMS or Verilog-AMS, and automatically generate device simulator codes that can be linked with a circuit simulator such as SPICE.

A compact device model is described as a set of time-dependent ordinary differential equations. These equations must be formulated before they can be solved. Using automatic modeling techniques described in [2][14][20][22], these equations can be transformed into a set of nonlinear functions (2.1) to calculate their corresponding entries in the Jacobian matrix and the right hand side (RHS) vectors. These functions will be evaluated during simulation.

$$y_i = f_i(x_1, x_2, \cdots, x_m, c_1, c_2, \cdots, c_n) \tag{2.1}$$

where x_i are independent variables, such as voltages of device terminals. Since *if-else-endif* block is frequently used when describing complex device models such as BSIM3 and BSIMSOI, c_i are used to formulate condition descriptions. The functions f_i are currently composed of the following operators {+, -, *, /, ^, log, exp}. The operators in c_i include {>, >=, ==, <, <=}. Each function f_i is mapped to an Abstract-Syntax-Tree (AST) that forms the foundation of MCAST and the optimization algorithms.

Figure 1 shows one of the AST of a MOSFET level 1 model. Full description of this model can be found in [2]. The root of the tree is variable Ids, where leaf nodes can be constants or terminal voltages. Different from traditional AST used in computer science, we introduce a new type of *Switch (SW)* node to represent the widely used *if-else-endif* structure in VHDL-AMS. One SW node represents one condition in (2.1).

III. HIERARCHICAL TWO DIMENSIONAL TABLE LOOKUP ALGORITHM

High computational complexity is a major challenge for device model evaluation. The basic idea of our table lookup method is to replace computation-intensive blocks by two-dimension tables to save the evaluation time. Below, we first describe a table build up algorithm.

A. Building the hierarchy of tables

Our table lookup method starts with the calculation of the evaluation costs of all of the basic operators {+, -, *, /, ^, log, exp, Boolean operators}, etc. The evaluation cost of an operator is an empirical value and is defined as the relative ratio of the running time of the operator to the running time of the "+" operation. This is achieved by taking the average value of 10^6 tests. The evaluation cost of "+" is assigned to 1. Since the evaluation costs may be different on different machine, they are measured in real time when the compiler runs.

The building process of the hierarchical table lookup model is a *reduction* process in which a sub-tree representing a computation-intensive block of the AST is reduced to a two-dimension table. Table 1 shows the reduction algorithm, which is a depth-first, recursive algorithm. It starts from the root of the AST to be optimized, but the real reduction process is bottom-up from the leaf nodes.

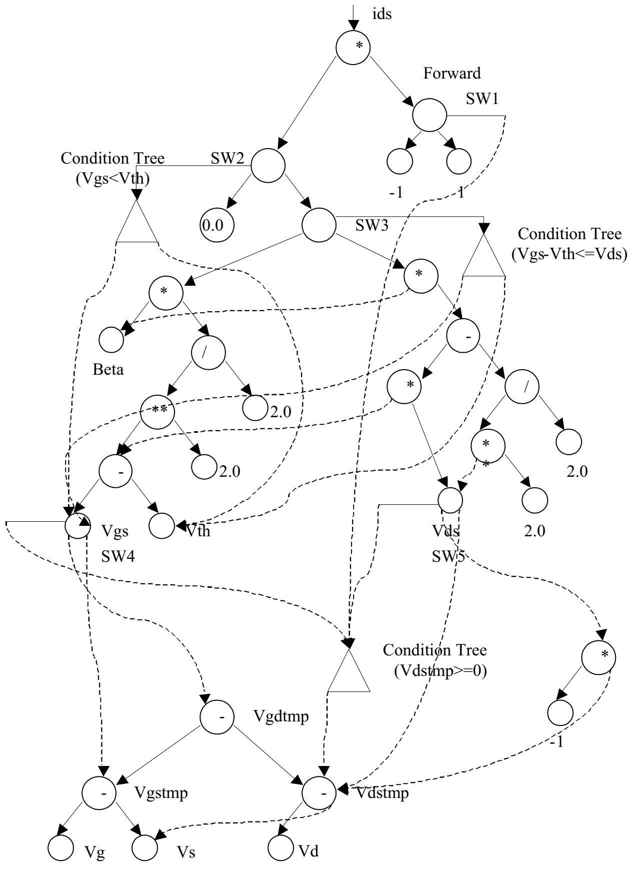


Figure 1. An AST example for MOS Level 1 model.

TABLE 1. REDUCTION ALGORITHM

```
Algorithm: Reduction
Input: AST Tree Node T
This algorithm begins with the root of AST
Output: Reduced AST with tables
1. Reduction for T's left child if exists
2. Reduction for T's right child if exists
3. Set related variables for T
4. Combine, if success, return
5. For T's left and right children, if they have been
   reduced to a table, reset their related variables.
6. Reset T's related variables
7. Set T's calculation cost
8. If T is leaf node, return
9. If T's number of related variables > 2, set T as a
   bottleneck node, return
10. If T's calculation cost > evaluation cost threshold &&
    T's number of related variables == 2, reduce T to a
    table.
```

The details of some steps are explained below:

- A node T's related variables are those node voltages that affect T's evaluation. In step 3, T's related variables are the sum of its children's related variables.
- In order to contain as more operations as possible in the reduced two-dimension table, step 4 has a combination process that helps to build the table upward as high as possible in AST, and thus we can reduce the number of tables. The combination process will try to combine T and its children's tables together if the tables exists and they share the same set of related variables.
- In steps 5 and 6, for T's each child C, if C has been reduced to a table, C's related variables will be reset to contain only one related variable that is C's name. Therefore, we can reduce the number of related variables and can build multi-level tables further based on the new related variable. Accordingly T's related variables are reset based on the children's new related variables.
- In step 7, T's calculation cost is calculated as the sum of T's children's calculation. The calculation cost of leaf nodes, such as the primary device node voltage node, parameter node and constant node, etc., are set to a very small number in practice.
- In step 9, a bottleneck node B is recognized if it has more than two related variables. A bottleneck node can not be reduced to a 2-D table. But B's related variables still have to be reduced to the name of B, and B could become the base related variable of up-level tables.
- Step 10 shows the real condition for T to be reduced to a 2-D table. The evaluation cost threshold is assigned to the evaluation cost from a 2-D table.

Figure 2 illustrates the reduction progress on a MOSFET level 1 AST (simplified for clarity). After the reduction, three tables, A, B and C, are created hierarchically. Table C's relative variables are Vds and B, which itself is also a table.

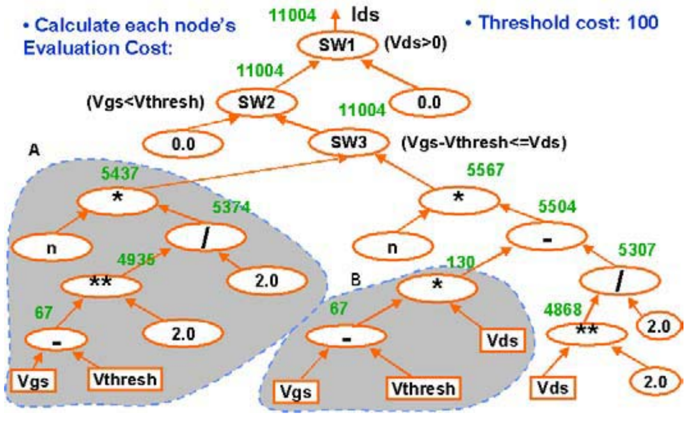


Figure 2(a) AST with evaluation cost.
Assume threshold cost is 100. Sub-tree A and B will be reduced

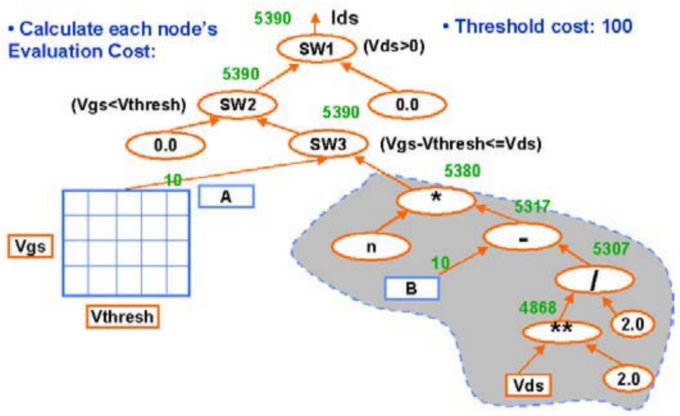


Figure 2(b) Multi-level table reduction.

B. Code generation of the table lookup model

MCAST model compiler generates C/C++ codes for the device model based on the reduced AST. When reaching a table, instead of outputting a block of evaluation codes, a routine of bilinear interpolation [13] for two dimensional table lookup is generated. The computation-intensive block of evaluation codes will also be output but in a separated routine which will be used later on to fill in the table. Bilinear interpolation is adopted since it is computation lightly and it is accurate enough in our process. To locate the four points surrounding the interpolation point, bi-section search is used. One should note that the table spaces are not uniformly separated because dimension variables may change on logarithmic scale and table looked-up variable from the lower level may become clustered or sparse in the dimension for the higher level tables.

C. Evaluation of the table lookup model

The setup routine in a target simulator is modified to fill in the tables for each instance of the device. Compared to the iterative load operation, the running time of the one time setup operation is relatively small [14]. If a circuit to be simulated does not have many new device instances, MCAST has an option to allow the tables to be filled by MCAST and the setup routine in the target simulator only needs to read in the tables, which saves the time for filling the tables.

During the simulation, the computation-intensive blocks are replaced by the computation lightly interpolation processes, therefore, the simulation time is saved.

Huge speedup can be obtained using our proposed hierarchical multi-dimensional table lookup method. But table lookup does introduce errors in the calculation. Simulation result may be wrong if error is not controlled. Beside that, the non-convergence problem may get worse if the circuit is sensitive to the inaccurate calculation of the equivalent conductance (derivative). The additional errors coming from the table lookup may cause the circuit failed to converge. In the following section, we introduce an error-oriented method to control the sizes of the lookup tables.

IV. Error Controlled Table Sizing

As mentioned in the previous section, the table lookup model should have several tables. These tables should be appropriately sized due to the saving requirements of memory capacity and computation time. The aim is to find a set of minimized table sizes such that in the worst case the errors of the interpolated values are less than a given relative error. An error analysis method [15] is used to set the table sizes.

Beginning with a given maximal allowed relative error (Emax), a nonlinear multivariable function is represented by an AST and a given set of intervals for input variables. The AST representing the nonlinear function is decomposed into switch nodes and calculation nodes, each of which is either a double operand operator or a single operand operator, with the restriction for the choice of operators as {+, -, *, /, ^, log, exp}.

For the error analysis, the AST needs to be modified following the rules in Table 2 with an exception that if either A or B is a constant instead of a variable, the modification is unnecessary. The purpose of the modification is making the formal error analysis (will be discussed later) possible.

Table 2. AST modification rule for error analysis.

A * B	Exp(logA + logB)
A / B	Exp(logA – logB)
A ^ B (B is a constant)	Exp (B * logA)
A ^ B	Exp(exp(logB + log(logA)))

Since the logarithm function is undefined for arguments that are smaller than or equal to zero. A transformation of a product of two variables is needed for variables that may have negative values (Fig. 3). Similar transformation are required for / as well as ^.

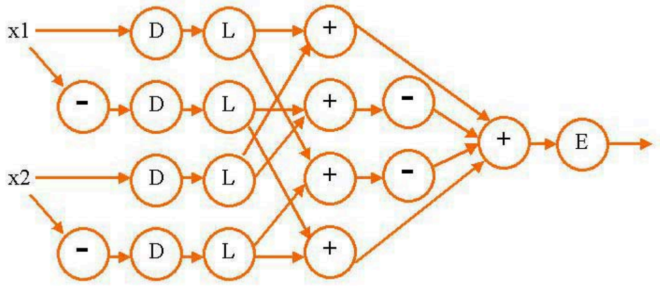
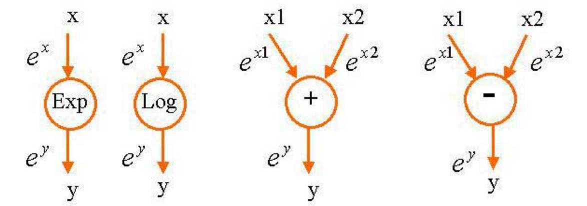


Fig.3. Transformation of variables that may have negative values. Legend: (D) ideal Diode, only positive values can get through.
(L) Log (-) Minus (+) Add (E) Exp

After the modifications and the transformations, the operators like {*, /, ^} will be eliminated from the AST. This modified AST has been isolated as several sub-trees. As mentioned before, each sub-tree is replaced by a two dimensional table. For each of these sub-trees, the error driven sizing algorithm, which consisting of two major steps, is performed to set up an appropriate size of the table. Each of the two steps is a recursive processing along the modified AST.

- First, the intervals of the function and all of the intermediate variables are calculated bottom up rippling from the leaves of the AST. Since the modified AST contains just plus, minus nodes or one incoming node, the intervals are calculated as follows: When a node has one incoming node, its interval is the operation result upon the child's interval. The interval of a plus node is a sum of the intervals of its two children. The interval of a minus node e.g. x1-x2 is (x1min-x2max, x1max-x2min).
- Second, the relative error for each node is calculated top-down staring with the maximal allowed error of the root of the tree and rippling down to the leaf nodes. The error of any node is given by the following equations [15]:



$$e_x = \frac{\ln\left(1+e_y\right)}{\max\left(\left|x_{\min}\right|,\left|x_{\max}\right|\right)}; \qquad \text{(Exp)}$$

and

$$e_x = \left|x_m^{\pm e_y} - 1\right|, x_m = \begin{cases} x_{\min}; x_{\min} > 1 \\ x_{\max}; x_{\max} < 1 \end{cases}; \qquad \text{(Log)}$$

For a plus or minus node y to its children x1 and x2:

$$e_{x1} = \left|\frac{e_y}{2}\cdot\left(1+sign(y)\cdot\frac{\min\left(\left|x_{2\min}\right|,\left|x_{2\max}\right|\right)}{\max\left(\left|x_{1\min}\right|,\left|x_{1\max}\right|\right)}\right)\right|$$

$$e_{x2} = \left|\frac{e_y}{2}\cdot\left(1+sign(y)\cdot\frac{\min\left(\left|x_{1\min}\right|,\left|x_{1\max}\right|\right)}{\max\left(\left|x_{2\min}\right|,\left|x_{2\max}\right|\right)}\right)\right|$$

In this way, all of the nodes will get their largest possible relative errors, which will ensure that in the worst case the overall error will be restricted in the given maximal relative error.

After obtaining the interval and relative error of the variable in the table lookup sub-tree, its table size is simply set to be the interval divided by the relative error.

V. Experimental Results

As an example, MOSFET level 3 model [16] has been implemented by MCAST, linked and built in the open source circuit simulator, Berkeley's SPICE3f5, to compare with human optimized codes (existing built-in device model codes in SPICE3f5). Some notions are used in the comparisons: "Built-in" model is the one manually implemented in SPICE3f5, "Non-optimized" model is the one automatically generated by MCAST but without any optimizations, "Table lookup" model is the one automatically generated by M with optimizations, including table lookup. The accuracy and efficiency of the generated table lookup model are demonstrated by the simulation results.

A. *Accuracy*

The automatic generated table lookup model of the level 3 model from MCAST is very accurate. Figure 4 shows the comparison of the I-V curves. The automatic generated model without table lookup yields exactly the same results from the manually implemented built-in model of level 3 in SPICE3f5. The simulation results also show that the table lookup model is accurate: the errors are constrained below 2% of the built-in model.

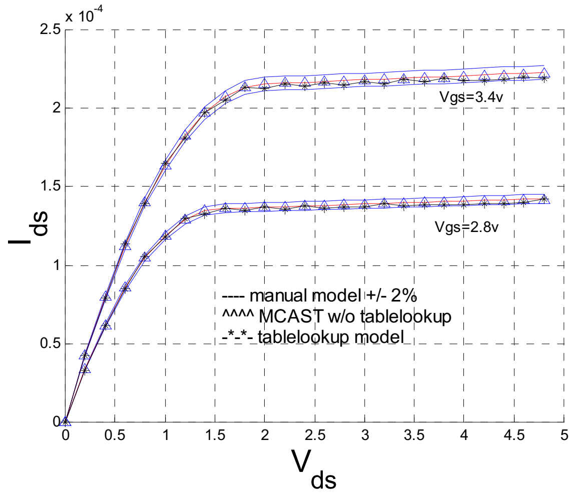


Fig. 4. Accuracy comparison: I-V curves.

Figure 5 shows the transient simulation results of one of our benchmark circuits – power amplifier. The result with table loop up matches well that with analytic evaluation.

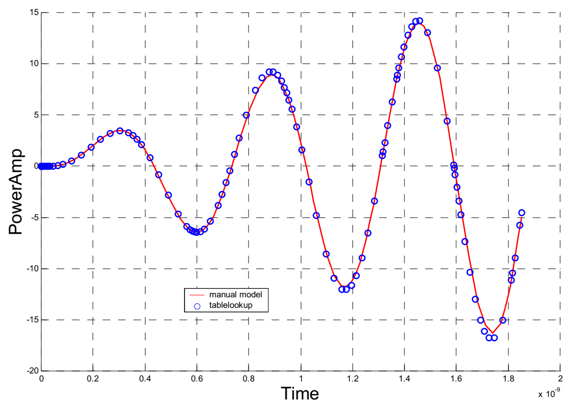


Fig. 5 Accuracy comparison: transient analysis.

B. *Performance*

Figure 6 shows a comparison among different model implementations, including table lookup model, Built-in model and Non-optimized model, of different devices, such as diode, MOSFET level 1 and level 3. The experiment is circuit-independent and only the model evaluation times are compared and normalized. In pure comparison of the evaluation costs of the different models, the table lookup model is at least three times faster than the built-in model and 20-40 times faster than the non-optimized model.

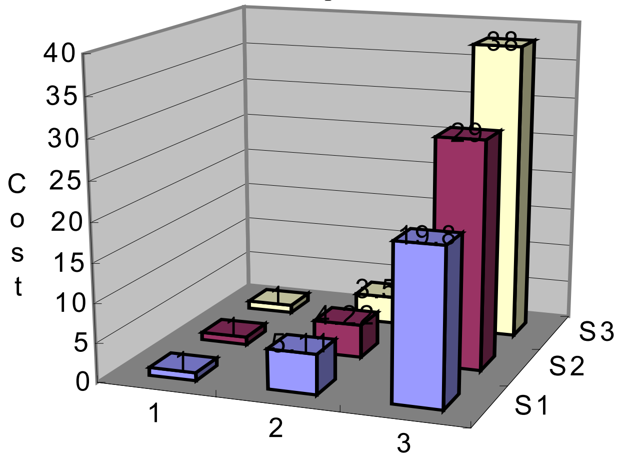


Fig. 6 Normalized model evaluation cost. (1) Table lookup model. (2) Built-in model. (3) Non-optimized model. (S1) Diode model. (S2) MOSFET Level 1 model. (S3) MOSFET Level 3 model.

We also compared the performances in transient analysis. Eight analog and digital benchmark circuits, including Power Amplifier and 8-bit Adder, etc., are used to demonstrate the speed-up results of the table lookup model of the MOSFET model of level 3 versus the built-in model (Fig. 7). We use the device loading time per iteration here for comparison to ignore the convergence effect. The performance of the built-in model is normalized to one. For most of the benchmark circuit, the speed-up is more than two times.

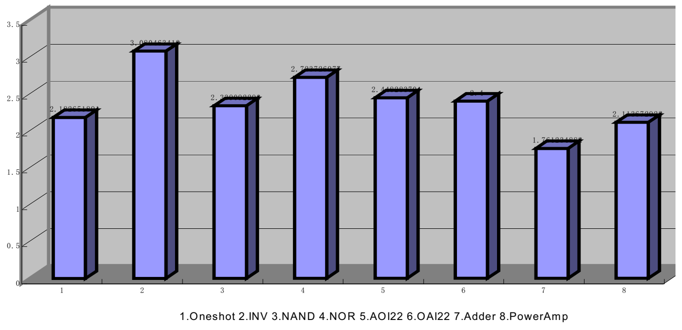


Fig. 7 Speed-up of the table lookup model compared to the built-in model over eight benchmark circuits.

C. *Table Sizing*

To find out the relationship between the accuracy and the memory requirement, a simple CMOS inverter was tested. We swept the capacity of all tables per instance of the device (MOSFET level 3 NMOS transistor) from 500 points to 20,000 points and collected the overall errors of one of the major variables, e.g., Ids of the pull-down transistor.

Figure 8 indicates that when the table size is small, accuracy is almost proportional to the capacity of the tables (errors are small). Accuracy can be easily improved by extending the table sizes. This corresponds to region 1.

But when the capacity of all tables exceeds a limit point, e.g. 4,000 points in this test case, the gain of accuracy is very limited and accuracy will not be improved by increasing the size of the tables. This corresponds to region 2.

The break point will change depending on the type of function that is being tabled. It is higher for function with complex behavior than for simple function. Fortunately, by setting the overall error allowed to be 2% for the major evaluation variables, the proposed table sizing method usually can find the appropriate sizes for all tables.

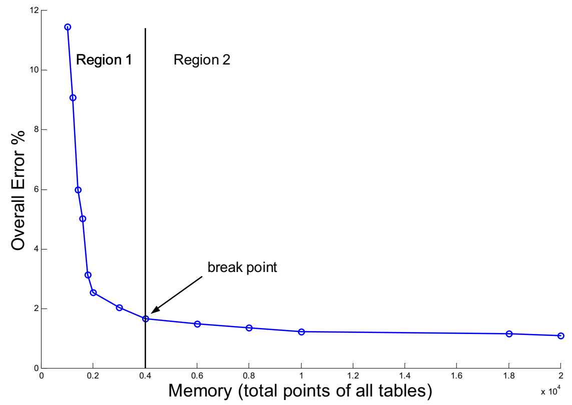


Fig. 8 Error Vs Memory.

VI. Conclusion

We have presented a systematic and automatic method for generating hierarchical multi-dimensional table lookup models for model-compiler-based precise circuit simulation. Any compact device and behavioral model described using high-level languages VHDL-AMS and Verilog-A(MS) can be used. The proposed method is based on an Abstract Syntax Tree representation of behavioral model equations for any devices with arbitrarily number of terminals. A method capable of generating lookup tables subject to a given accuracy requirement but with the minimal amount of memory for storing the data table has been developed.

The proposed method has been implemented in our compact model compiler MCAST and targeted the SPICE3 simulator. Experiment results on a set of standard test circuits have demonstrated that the generated table lookup models are accurate with the error in the range of 1-2%, but at least three times faster than human optimized built-in models, and 30-40 times faster than automatic generated models without optimizations. Furthermore, the proposed error-controlled automatic table sizing method yields nearly minimal table sizes.

References

[1] K. Kundert, “Automatic Model Compilation – An Idea Whose Time Has Come”, *The Designer’s Guide*, May 2002. http://www.designers-guide.com/Opinion/modcomp.pdf

[2] B. Wan, B. P. Hu, L. Zhou and C.-J.Richard Shi, “MCAST: An Abstract-Syntax-Tree based Model Compiler for Circuit Simulation”, *Proc. IEEE Custom Integrated Circuit Conference*, pp. 249-252, Sept. 2003.

[3] S. Liu, K.C.Hsu, P.Subramaniam, “ADMIT-ADVICE Modeling Interface Tool”, *Proc. IEEE Custom Integrated Circuits Conference*, pp. 6.6/1-6.6/4, May 1988.

[4] A.T.Yang, and S.M.Kang, “iSMILE: A Novel Circuit Simulation Program with emphasis on New Device Model Development”, *Proc. IEEE 26th Design Automation Conference,* pp. 630-633, June 1989.

[5] L. Lemaitre, C. McAndrew, and S. Hamm, “ADMS-Automatic Device Model Synthesizer”, *Proc. IEEE Custom Integrated Circuits Conference*, pp. 27-30, May 2002.

[6] R. V. H. Booth, “An Extensible Compact Model Description Language and Compiler”, *Proc. IEEE/ACM BMAS*, pp. 39-44, Oct. 2001.

[7] Hal Carter, “Modeling and Simulating Semiconductor Devices Using VHDL-AMS”, *Proc. IEEE/ACM BMAS*, pp. 22-27, Oct. 2000.

[8] A. Rofougaran and A. A.Abidi, “A Table Lookup FET Model for Accurate Analog Circuit Simulation,” *IEEE Trans. Computer-Aided Design*, vol. 12, pp. 324-335, Feb. 1993.

[9] M.G. Graham and J.J. Paulos, ”Interpolation of MOSFET Table Data In Width, Length, and Temperature”, *IEEE Trans. Computer-Aided Design*, vol. 12, pp. 1880-1884, Dec. 1993.

[10] T. Shima, Ts. Sugawara, S. Moriyama and H. Yamada, “Three-Dimensional Table Look-Up MOSFET Model for Precise Circuit Simulation”, *IEEE J. Solid-State Circuits CS*-17, 3, pp. 449-454, 1982.

[11] T. Shima, H. Yamada and R.L.M. Dang, “Table Look-Up MOSFET Modeling System Using a 2-D Device Simulator and Monotonic Piecewise Cubic Interpolation”, *IEEE Trans.Computer-Aided Design CAD*-2, 2, pp. 121-126, 1983.

[12] B. R. Chawla, et al. “MOTIS-An MOS Timing Simulator”, *IEEE Trans. Circuits and Systems*, Dec. 1975.

[13] H.W. Press, *Numerical Receipt in C*, Cambridge University Press, 1993.

[14] L. W. Nagel, *SPICE2 – A computer program to simulate semiconductor circuits*, Univ. of California, Berkeley, ERL Memo ERL-M520, May 1975.

[15] D.M.W. Leenaerts, W.M.G. van Bokhoven, *Piecewise Linear Modeling and Analysis*, Kluwer Academic Publishers, 1998

[16] HSPICE manual, *Avanti! Corp*. 1999.

[17] Y. Cheng and C. Hu, *MOSFET Modeling & BSIM3 User’s Guide,* Kluwer Academic Publisher, 1999.

[18] MAST/Saber User Manual, *Analogy Inc*.

[19] H. A. Mantooth, http://mixedsignal.eleg.uark.edu/paragon.html

[20] W. J. McCalla, *Fundamentals of Computer-Aided Circuit Simulation,* Kluwer Academic Publishers, 1988.

[21] V. Litovsky and M. Zwolinsky, *VLSI Circuit Simulation and Optimization*, Chapman & Hal*l*, 1997.

[22] Vlach, Jiri and Singhal, *Computer Methods for Circuit Analysis and Design*, Van Nostrand Reinhodl, 1994.

[23] Coleman, Thomas F. and Varma, *The Efficient Computation of Sparse Jacobian Matricies Using Automatic Differentiation*, Cornel Theory Center Technical Report CTC95TR225, 1996.

[24] A. Griewank, et. Al. “ADOL-C, A Package for Automatic Differentiation of Algorithms Written in C/C++”, *ACM Transaction In Mathematics Software*, 1990.

[25] T. L. Quarles, *Analysis of Performance and Convergence Issues for Circuit Simulation*, U. C. Berkeley, Memorandum No. UCB/ERL M89/42, April 1989.

Modeling and simulation of circuit-electromagnetic effects in electronic design flow

Pavel V. Nikitin[1], Vikram Jandhyala[1], Daniel White[2], Nathan Champagne[2], John D. Rockway[2], C.-J. Richard Shi[1], Chuanyi Yang[1], Yong Wang[1], Gong Ouyang[1], Rob Sharpe[2], and John W. Rockway[3]

[1]University of Washington, Department of Electrical Engineering, Seattle, WA 98195
Email: {nikitin, jandhyala, cjshi, cyang1, oyg, yongw}@ee.washington.edu

[2]Lawrence Livermore National Laboratory, 7000 East Ave, Livermore, CA 94550
Email: {dwhite, champagne, rockway2, rsharpe} @llnl.gov

[3]Space and Naval Warfare Systems Command, 4301 Pacific Highway San Diego, CA 92110
Email: rockway@spawar.navy.mil

Abstract

The goal of this paper is to describe a methodology for modeling and simulation of circuit-electromagnetic (EM) effects that fits into a current electronic design flow. Our methodology is based on using time-domain macromodels implemented in a hardware description language (HDL). Simulation of the entire coupled circuit-EM system can be carried out either entirely in HDL simulator or in SPICE-type circuit simulator (using model compiler for macromodel import). We also describe in detail a circuit-EM contact interface and a neutral mesh format necessary to allow for flexibility in choice of EM simulators. At each step of our methodology, we provide an overview of current problems and solutions with reference to existing publications.

As a demonstration example, we consider a simple coupled system (MEMS resonator connected to a lumped circuit) and show that simulations using VHDL-AMS macromodel match full-wave EM results but easily fit in the design flow and take significantly less time. Our methodology is straightforward and permits the use of various EM simulators and macromodel identification algorithms[1].

1 This research was supported by DARPA NeoCAD Program

1. Introduction

Electromagnetic effects have always been important in microwave circuits but now they have become an increasingly significant factor that affects the performance of modern integrated circuit (IC) systems, especially at multi-gigahertz frequencies [18]. Such systems include very large scale integrated (VLSI) chips as well systems-on-chips (SoC), and the examples of objects exhibiting EM behavior are interconnects, spiral inductors, traces, etc. This leads to a necessity of using accurate computer-automated design (CAD) tools for EM modeling and efficient use of those models in circuit simulation [6].

A variety of numerical electromagnetic field solving tools have been developed in the past, all of which have different limitations, capabilities, input and output formats, and computational costs. Choosing the best tool for a particular task and successfully employing and integrating it into an IC CAD design flow are challenging tasks.

Both circuit and EM simulations can be carried out either in time domain or frequency domain but mixed-signal circuit simulations are mostly performed in time domain (due to nonlinearity of analog circuits and sharp rise and fall times of digital signals) whereas EM simulations are mostly performed in frequency domain (due to well developed frequency domain EM methods).

There are three main approaches to incorporate EM simulation results in SPICE-type time-domain circuit simulators. First approach is to extract an equivalent RLC circuit [1], which can be very large (i.e. for substrate coupling) and cumbersome to deal with (model order reduction is often needed). Second approach is to concurrently couple a circuit and EM simulator. While adding lumped passives to a full-wave EM simulation is straightforward, coupling a full-wave EM solver with a non-linear circuit solver is not a routine procedure (e.g. FDTD-SPICE coupling has been done but on case-by-case basis [15]). Third approach, which we describe in this paper, is to develop compact linear EM macromodels [10].

The last approach is very convenient because macromodels can be implemented in high-level hardware description languages used for design (such as VHDL-AMS [3] or Verilog-A [12]), easily interfaced to non-linear circuits, and re-used. Macromodeling permits significant speed-up of simulations and thus gains more and more attention in CAD community (e.g., for MEMS [17]). We should note that propositions to extend HDL's to directly support PDE's and hence EM modeling have also appeared in the literature [13] but this work is still in the research stage.

In this paper, we describe a methodology for modeling and simulation of circuit-EM effects on system performance by using compact linear EM macromodels implemented in a hardware description language. We provide an example – a simple circuit-driven MEMS system analyzed using VHDL-AMS macromodel extracted from time-domain EM simulation. We also describe specifics of circuit-EM contact interface and EM mesh format in a way that can be used by different circuit and EM simulators.

2. Methodology

Modern electronic design flow includes such steps as schematic capture and simulation, system layout, parasitic extraction, post-layout simulation, etc. At each stage, different tools and file formats, standard and proprietary, are used [9].

Analog and digital circuitry is typically described using SPICE- or VHDL-type netlists, which specify how lumped components or digital logic blocks are connected together. Layout is typically described using CIF or GDS II format files. These files contain 2D data about structures located at different chip layers and together with technology files (which contain information about thickness, material properties, and stacking of different layers) give a complete 3D description of a chip.

Having an ability to do an accurate post-layout simulation is critical for verification of functionality and performance of the complete system. Fully coupled circuit-electromagnetic simulations are very computationally intensive and are not commonly used. A typical approach used in the design process today is to perform parasitic extraction and include equivalent RLC circuits into a circuit simulator.

The process of RLC extraction from EM simulations is difficult, but works well in many cases, especially for capacitances of interconnects. Complex coupled problems result in large RLC networks and require a subsequent application of model order reduction methods, which are not well integrated into design flow. Thus there is a clear need for new approaches in coupled circuit-EM simulation.

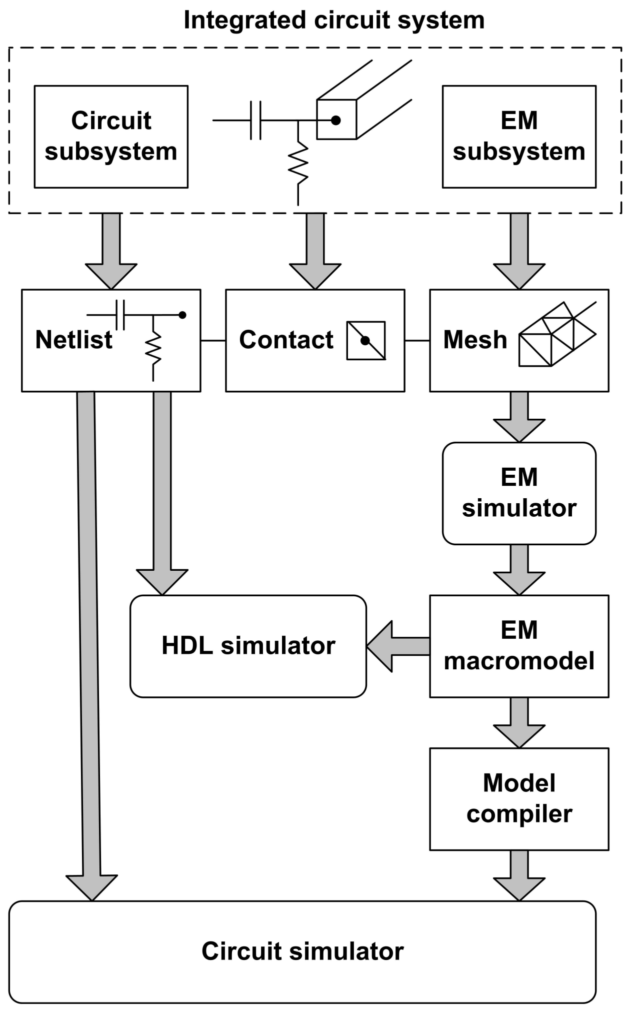


Figure 1. Methodology.

The methodology that we propose is illustrated in Figure 1. An IC system of interest contains lumped circuits connected at certain contact points to geometrical structures that exhibit EM behavior and need to be meshed and accurately modeled. Volumetric or surface mesh is stored in neutral mesh format reusable by various electromagnetic simulators. From frequency- or time-domain simulation data (depending on application and frequency range of interest), time-domain macromodel can be identified and extracted [20]. Such model can easily be implemented in a hardware description language (such as VHDL-AMS) and used either in HDL simulation of the whole system (circuit netlist needs to be converted from SPICE to HDL format) or, with recent advances in model compilers [7, 23], compiled for direct use in a SPICE-type circuit simulator.

2.1. EM simulation, contact interface, mesh format

In circuit simulation, the most popular method is node-based modified nodal analysis (MNA) [16]. In electromagnetic simulation, the variety of methods is richer and includes differential methods (FDTD – finite difference time domain, FEM – finite element method, etc.), integral equation methods (MoM – method of moments, BEM – boundary element method, etc.), hybrid methods [21], etc. Many of these methods can be utilized both in frequency or time domain but traditionally only FDTD has been used for time-domain modeling, and FEM and MoM have been used in frequency domain. Recently, new time-domain methods (TD-FEM [24], TD-MoM [26]) have been developed and successfully applied to a variety of problems. An excellent survey of existing EM methods can be found in [11].

Each method listed above has many variations and deserves a separate overview but most EM commercial tools are based on three major methods and their flavors – method of moments (e.g., *Sonnet* by Sonnet Technologies), finite element method (e.g., *HFSS* by Ansoft Corporation), and finite-difference time domain method (e.g., *XFDTD* by Remcom, Inc.). All electromagnetic solvers require creation of some sort of grid or mesh: either volumetric one that includes all problem space (FEM and FDTD) or surface mesh that covers only certain surfaces (MoM).

An electromagnetic solver applied to coupled circuit-EM problem must recognize the existence of ports or terminals that connect circuit and EM subsystems and through which the interaction happens [22]. Exact definition is different for different EM solving techniques [2]. Examples of specifying such interaction for FDTD can be found in [15] and for MoM in [26, 5]. Circuit world understands currents and voltages, and thus latter serve as common shared quantities at the points of circuit-EM interaction.

Assume that we have identified EM objects and lumped circuit elements connected to them (identification of IC package parts that must be modeled as EM objects is a separate challenging problems that we do not address here). Then circuit-EM contact interface can be defined as an area of the EM object surface to which a circuit element is attached. This concept is shown in Figure 2 (two contacts may form a microwave port).

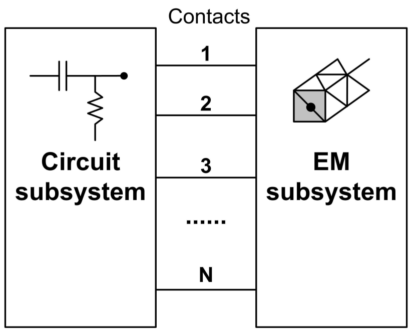


Figure 2. Circuit-EM contact interface.

The contact interface area can be specified in two ways: *mesh-dependent* and *mesh-independent*. *Mesh-dependent* method can be defined as specifying mesh elements that belong to the contact interface. *Mesh-independent* method can be defined as specifying 3D coordinates of contact points (using either x, y, z coordinates in the integrated chip reference frame or text labels in layout/technology files). After the mesh is created, mesh faces in the vicinity of that point (e.g, a spherical region of a certain radius) are recognized as part of contact interface.

Both ways described above have advantages and disadvantages. Mesh-dependent method is less portable as it requires the existence of prior mesh but is better for accurate coupled simulations. Mesh-independent method does not require prior mesh existence and has better portability but may suffer from potential problems related to mesh refinement in the process of EM solution.

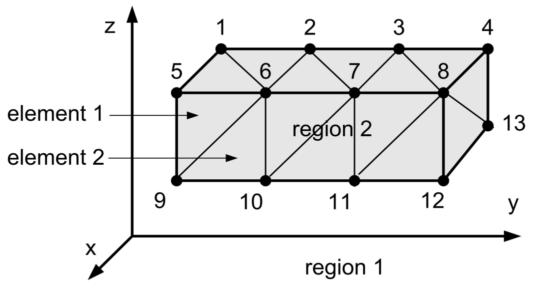


Figure 3. Mesh format.

Mesh itself can also be stored in a variety of ways. Currently, many different mesh formats for EM simulation exist. Unfortunately, there is no standard analogous to netlist standard for circuits. We propose to use the following neutral mesh format, simple and intuitive. To completely define a mesh, three files are needed: *node file*, *element file*, and *material file*. The format of those files can be illustrated with the example shown in Figure 3, where a surface of a perfectly conducting object positioned in free-space is meshed with triangles.

Node file lists coordinates of all nodes (in units selected by user) in the cartesian coordinate system. The node file for the example shown in Figure 3 is:

```
Node x  y  z
1    x1 y1 z1
2    x2 y2 z2
...
```

Element file lists all surface and volume elements (triangles, tetrahedra, etc.) formed by nodes which serve as element vertices. If an element belongs to a surface dividing two regions with different properties, those regions must be specified by their numbers. In the example shown in Figure 3 the elements are triangles on the surface dividing region 1 and region 2, and the node file is:

```
Element n1 n2 n3 region1 region2
1       5  6  9  1       2
2       6  9  10 1       2
...
```

Material file lists all regions (by number), their type (volume, surface, layer), and their properties (permittivity, permeability, and conductivity). Infinite conductivity for perfect electric conductors can be denoted as PEC. The example shown in Figure 3 contains free-space (region 1) and a PEC object (region 2). The material file for this example is:

```
Region eps mu sigma type
1      1   1  0     volume
2      1   1  PEC   volume
...
```

The mesh format, described above, can be used for different EM simulators and translated into mesh formats understood by any of the commercial tools. Once an EM simulation of the multi-port structure is completed, a macromodel needs to be extracted. This process is described in the next subsection.

2.2. Macromodeling

Macromodeling is extremely important for speeding up simulations of complex systems, such as coupled circuit-electromagnetic systems. In order to be easily implementable in a hardware description language, a macromodel must be casted into a time-domain differential equation form. Such model can be obtained from either frequency- or time-domain EM simulation.

A number of different algorithms for extracting macromodels and reduced order models from data are available [14, 8]. An advantage of using time-domain data is that in most cases passivity and stability of obtained macromodel are easier to guarantee than when working with frequency-domain data. Thus, for illustration of our methodology, we choose an approach where a linear compact macromodel is identified from a time-domain electromagnetic response as described in [25].

All possible information about system dynamics is theoretically contained in an impulse response – a system response to a delta-function excitation. System response to any input can be found as a convolution of the impulse response with the input signal. This process is very computationally expensive, especially for highly-resonant devices with long impulse responses. In addition, delta-function causes numerical problems in time-domain EM solvers, and more commonly used excitation is Gaussian pulse:

$$u(t) = u_o \, e^{-\frac{(t-\tau)^2}{2T^2}} \tag{1}$$

with -3dB bandwidth of 0.13/T.

System response to a Gaussian pulse can allow one to identify a continuous time-domain macromodel in its classical state-space form:

$$\begin{aligned} \dot{\vec{x}} &= \hat{A}\,\vec{x} + \hat{B}\,\vec{u} + \hat{K}\,\vec{e}, \\ \vec{y} &= \hat{C}\,\vec{x} + \hat{D}\,\vec{u} + \vec{e}, \end{aligned} \tag{2}$$

where $\vec{x}(t)$ is the vector of state variables, $\vec{u}(t)$ is the excitation, $\vec{y}(t)$ is the output, and $\vec{e}(t)$ is the noise signal. The process of identification can be described as finding $\hat{A}$, $\hat{B}$, $\hat{C}$, $\hat{D}$, and $\hat{K}$ from given $\vec{u}(t)$ and $\vec{y}(t)$.

There exists a large number of different methods and tools for system identification (see, e.g., *MATLAB*[2] system identification toolbox). The order of the model (dimension of the $\hat{A}$ matrix) can be determined from the data. The accuracy and other issues associated with macromodel identification, such as passivity and stability, are not discussed here since they are well covered in the literature (see, e.g., [4, 19]) and lie outside the scope of this paper.

The time-domain state-space model (2) is essentially a set of ordinary differential equations that can easily be implemented in a hardware description language for later use in circuit simulation, as it is shown in the next section.

3. Example

For demonstration of modeling flow methodology described above, consider a simple example: MEMS resonator (micromachined comb structure, approximately 1.5 mm × 0.5 mm in size, and positioned in free-space) driven by an external voltage source as shown in Figure 4. This MEMS structure represents an electromagnetic subsystem and can be thought of as part of a larger integrated package. The voltage source and the resistor represent a lumped circuit subsystem (which can be any transistor circuit).

2 Trademark of *Mathworks, Inc.*

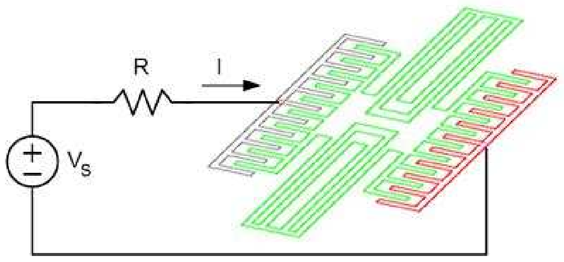


Figure 4. Circuit-driven MEMS resonator.

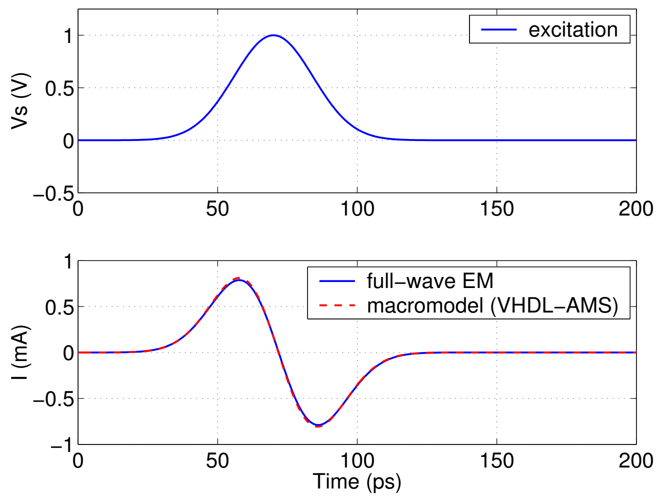


Figure 5. Results of the simulations for the system shown in Figure 4.

The voltage source generates a Gaussian pulse of the form (1) with $u_o = 1$ V, $\tau = 70$ ps, and $T = 14$ ps (bandwidth $\approx$ 10 GHz). The resistor is $R = 100$ Ohm. The mesh for MEMS structure was generated and stored in the neutral format described in the previous section. The problem was solved using a full-wave time-domain integral-equation method [26]. It contained about 1000 triangles (approximately 1500 unknowns) and took approximately 1 minute of runtime on a 1 GHz PC.

Consider a macromodel of the system that includes MEMS resonator in series with 100 Ohm resistor. The input $u(t)$ to this system is the excitation voltage from the source V_s and the output $y(t)$ is the current I through the system. For identifying the continuous state-space system model of the form (2), we used `'pem'` and `'d2c'` functions in *MATLAB* system identification toolbox. The response $y(t)$ was well approximated with the 3rd order model, where noise component was set to zero. The model was implemented in VHDL-AMS as shown below and simulated using VHDL-AMS simulator *Hamster*[3]. The runtime was 0.2 s on 2.5 GHz PC. As one can see from Figure 5, macromodel simulation results match the results of full-wave EM simulation very well.

```
----- Macromodel of MEMS resonator ---
-------- in series with resistor -----
ENTITY macromodel IS
 PORT (TERMINAL a, b : ELECTRICAL);
END;
 ARCHITECTURE behav OF macromodel IS
 QUANTITY u ACROSS i THROUGH a TO b;
 QUANTITY x1,x2,x3: real;
 CONSTANT A11 : real := -4.929E11;
 .....
 CONSTANT C3 : real := -2.04e-8;
 CONSTANT D : real := 0.00518;
BEGIN
x1'dot == A11*x1+A12*x2+A13*x3+B1*u;
x2'dot == A21*x1+A22*x2+A23*x3+B2*u;
x3'dot == A31*x1+A32*x2+A33*x3+B3*u;
-i == C1*x1+C2*x2+C3*x3+D*u;
END ARCHITECTURE;

------ System description ----------
ENTITY system IS END;
ARCHITECTURE behav OF system IS
    TERMINAL n1: ELECTRICAL;
BEGIN
Vs: ENTITY gaussian_source (behav)
    GENERIC MAP (1.0,70.0E-12,14.0E-12)
    PORT MAP (n1,electrical_ground);
Mm: ENTITY macromodel (behav)
    PORT MAP (n1,electrical_ground);
END behav;
```

This example demonstrates that macromodels are an accurate and efficient way of simulating coupled circuit-electromagnetic systems in time-domain. Macromodels in general contain much fewer internal variables than full EM problems (in our example, 3 vs. 1500) and thus provide a significant simulation speedup. They are easy to implement in HDL and can be used in today's design flow.

4. Conclusions

In this paper, we described in detail the methodology of modeling and simulation of coupled circuit-electromagnetic effects using time-domain EM macromodels implemented in a hardware description language. This methodology fits well into electronic design flow existing today. Simulation of complete integrated circuit system can be carried out either entirely in HDL or in SPICE-type circuit simulator

[3] Now part of *Simplorer*, trademark of *Ansoft Corp.*

(using HDL-to-SPICE model compiler). We have also defined a circuit-EM contact interface and a neutral geometry meshing format that can be used by various electromagnetic solvers used in the design process.

For demonstration, we considered a simple coupled system (MEMS resonator connected to a lumped circuit) and showed that VHDL-AMS macromodel simulation results match full-wave EM results but take significantly less time to obtain. This shows that EM macromodeling is a very effective way to include circuit-electromagnetic effects into simulation. Implementing macromodels in a hardware description language allows one to use them in the current IC design flow.

References

[1] R. Achar and M. S. Nakhla. Simulation of high-speed interconnects. *Proceedings of IEEE*, 89(5):693–728, May 2001.

[2] N. J. Champagne. On attaching a wire to a triangulated surface. *IEEE Antennas and Propagation Symposium Digest*, 1:54–57, June 2002.

[3] E. Christen and K. Bakalar. VHDL-AMS – a hardware description language for analog and mixed-signal applications. *IEEE Transactions on Circuits and Systems*, 46(10):1263–1272, October 1999.

[4] S. Grivet-Talocia, I. S. Stievano, I. A. Maio, and F. Canavero. Time-domain and frequency-domain macromodeling: application to package structures. *IEEE International Symposium on Electromagnetic Compatibility*, 2:570–574, August 2003.

[5] V. Jandhyala, Y. Wang, D. Gope, and C.-J. Shi. A surface-based integral-equation formulation for coupled electromagnetic and circuit simulation. *IEEE Microwave and Optical Technology Letters*, 34(2):103–106, July 2002.

[6] K. Kundert, H. Chang, D. Jefferies, G. Lamant, E. Malavasi, and F. Sendig. Design of mixed-signal systems-on-a-chip. *IEEE Transactions on CAD of Integrated Circuits and Systems*, 19(12):1561 –1571, December 2000.

[7] L. Lemaitre, C. McAndrew, and S. Hamm. ADMS – automatic device model synthesizer. *Proceedings of the IEEE Custom Integrated Circuits Conference*, pages 27–30, 2002.

[8] Y. Liu, L. T. Pileggi, and A. J. Strojwas. Ftd: frequency to time domain conversion for reduced-order interconnect simulation. *IEEE Transactions on Circuits and Systems*, 48(4):500–506, April 2001.

[9] D. MacMillen, R. Camposano, D. Hill, and T. W. Williams. An industrial view of electronic design automation. *IEEE Transactions on CAD of Integrated Circuits and Systems*, 19(12):1428–1448, December 2000.

[10] G. Marrocco and F. Bardati. Time-domain macromodel of planar microwave devices by FDTD and moment expansion. *IEEE Transactions on Microwave Theory and Techniques*, 49(7):1321–1328, July 2001.

[11] E. K. Miller. A selective survey of computational electromagnetics. *IEEE Transactions on Antennas and Propagation*, 36(9):1281–1305, September 1988.

[12] I. Miller and T. Cassagnes. Verilog-A and Verilog-AMS provide a new dimension in modeling and simulation. *Proceedings of the 2000 Third IEEE International Caracas Conference on Devices, Circuits and Systems*, pages C49/1–c49/6, March 2000.

[13] P. V. Nikitin, C. J.-R. Shi, and B. Wan. Modeling partial differential equations in VHDL-AMS. *IEEE System-on-Chip Conference*, pages 345–348, September 2003.

[14] A. Odabasioglu, M. Celik, and L. T. Pileggi. PRIMA: passive reduced-order interconnect macromodeling algorithm. *IEEE Transactions on CAD of Integrated Circuits and Systems*, 17(8):645–654, August 1998.

[15] N. Orhanovic and N. Matsui. FDTD-SPICE analysis of high-speed cells in silicon integrated circuits. *Proceedings of Electronic Components and Technology Conference*, pages 347–352, 2002.

[16] D. Pederson. A historical review of circuit simulation. *IEEE Transactions on Circuits and Systems*, 31(1):103–111, January 1984.

[17] B. F. Romanowicz. Methodology for the modeling and simulation of microsystems. *Kluwer Academic Publishers*, 1998.

[18] A. E. Ruehli and A. Cangellaris. Progress in the methodologies for the electrical modeling of interconnects and electronic packages. *Proceedings of IEEE*, 89(5):740–771, May 2001.

[19] J. J. Sanchez-Gasca, K. Clark, N. W. Miller, H. Okamoto, A. Kurita, and J. Chow. Identifying linear models from time domain simulations. *IEEE Computer Applications in Power*, 10(2):26–30, April 1997.

[20] K. Seok-Yoon, N. Gopal, and L. T. Pillage. Time-domain macromodels for VLSI interconnect analysis. *IEEE Transactions on CAD of Integrated Circuits and Systems*, 13(10):1257–1270, October 1994.

[21] R. Sharpe, J. B. Grant, N. J. Champagne, W. A. Johnson, R. E. Jorgenson, D. R. Wilton, W. J. Brown, and J. W. Rockway. EIGER: Electromagnetic Interactions GEneRalized. *IEEE Antennas and Propagation Symposium Digest*, 4(12):2366–2369, July 1997.

[22] I. A. Tsukerman, A. Konrad, G. Meunier, and J. C. Sabonnadiere. Coupled field-circuit problems: trends and accomplishments. *IEEE Transactions on Magnetics*, 29(2):1701–1704, August 1992.

[23] B. Wan, B. Hu, L. Zhou, and C.-J. R. Shi. MCAST: an abstract-syntax-tree based model compiler for circuit simulation. *Proceedings of IEEE Custom Integrated Circuits Conference*, 2003.

[24] D. A. White. Orthogonal vector basis functions for time domain finite element solution of the vector wave equation. *IEEE Transactions on Magnetics*, 35(3):1458–1461, May 1999.

[25] D. A. White and M. Stowell. Full wave simulation of electromagnetic coupling effects in RF and mixed-signal IC's using time domain finite element method. *IEEE Transactions on Microwave Theory and Techniques*, submitted.

[26] C. Yang and V. Jandhyala. A time domain surface integral technique for mixed electromagnetic and circuit simulation. *Electrical Performance of Electronic Packaging Conference*, pages 41–44, 2002.

Distributed Electrothermal Modeling in VHDL-AMS

Pavel V. Nikitin, Erik Normark, and C.-J. Richard Shi
Department of Electrical Engineering
University of Washington
Seattle, WA 98195-2500, USA.
Email: {nikitin, ecn1, cjshi}@ee.washington.edu

Abstract— **This paper demonstrates how to apply VHDL-AMS to modeling a coupled distributed electrothermal problem. We present an example of a simple system where two resistors on top of a silicon substrate are thermally coupled. Thermal exchange is described with a heat balance equation discretized and solved using finite difference method. The entire system is modeled in VHDL-AMS. Our work is a tutorial demonstration of VHDL-AMS capability to model coupled electrothermal systems beyond the traditional equivalent thermal network representation.** [1]

I. Introduction

A general thermal problem can be modeled by using an appropriate simulator that provides a numerical solution of the heat balance (or thermal diffusion) equation for a given set of geometry, sources, and boundary conditions [1]. Modeling a coupled electrothermal problem is more challenging due to the interaction between electrical devices and thermal processes.

Necessity to carry out a concurrent circuit simulation leads to coupling of a thermal simulator and a circuit simulator [2], [3]. A good example of such work can be found in [4], [5]. An alternative approach is to use an equivalent thermal network that consists of sources, capacitors, and resistors. These elements represent the effects of heatflow, self-heating, and mutual coupling between electrothermal devices that dissipate power or are sensitive to temperature. In this approach, the number of variables is drastically reduced. However, since all other thermal effects are reduced to equivalent thermal network representation [6], only the temperature at some spatial points can be found.

While in many cases the second approach provides an adequate accuracy, reducing a 3-D thermal problem to an equivalent thermal circuit network of reasonable size is not trivial, especially for complex systems. At the same time, in the first approach, the process of connecting two simulators is challenging and custom in each case due to the lack of commonly accepted modeling environment. The fact that the time scales for thermal and electric processes are usually quite different presents an additional difficulty in coordinating the operation and interaction of circuit and thermal simulators.

VHDL-AMS (an IEEE standard hardware description language [7]) is an example of the natural environment where modeling coupled electrothermal problems becomes straightforward. A thermal node can be added to any element in the circuit. Two quantities associated with each thermal node, temperature and heatflow, adequately describe electrothermal effects of the device. Thermal nodes of various elements are plugged into the thermal environment. The heat exchange can be modeled either using an equivalent thermal network or by solving the heat balance equation in the volume of interest.

Due to the fact that VHDL-AMS multi-physics capability is currently limited to differential and algebraic equations (DAE's) or equivalent circuits [8], an equivalent thermal network method remains the most popular approach for VHDL-AMS modeling of electrothermal problems. Representative works using this approach (also referred to as thermal impedance matrix approach [9]) include modeling of MMIC array [10], MOST transistor [11], self-heating diode [12], and resistor [13].

Up to date, there has been little work related to distributed electrothermal modeling in VHDL-AMS because of the lack of support for partial differential equations (PDE's) in the current language standard [14]. For example, [15] deals only with a 2-D thermal problem without any electrical circuits.

In this paper, we present an example of a distributed problem: a simple circuit where two resistors are electrically independent but thermally coupled via a silicon substrate. Both resistors dissipate power and are sensitive to temperature changes. Thermal exchange is described with a 2-D heat balance equation solved using a finite difference spatial discretization method, similar to [16], where a fully coupled 3-D electrothermal simulation has been carried in MAST language using *SABER*[2] This approach can also been applied to modeling PDE's in VHDL-AMS [17]. The entire system in our example is modeled purely in VHDL-AMS using freely available simulator *Hamster*[3].

The remainder of the paper is organized as follows. Sections II and III present an example and describe thermal modeling. VHDL-AMS implementation is presented in Section IV. Sections V and VI contain results and discussion. Conclusions are given in Section VII.

[1]This research was supported by DARPA NeoCAD Program under Grant No. N66001-01-8920 and NSF CAREER Award under Grant No. 9985507

[2]*SABER* is a trademark of *Synopsys Inc.*

[3]*Hamster* is now part of *Simplorer*, trademark of *Ansoft Corp.*

II. Example

Consider a simple circuit system shown in Fig. 1. The system consists of two circuits, which are decoupled electrically but coupled thermally due to the fact that both R_2 and R_3 are located on the same silicon substrate. Circuit 1 can represent a trigger line in the digital part of a mixed-signal system whereas circuit 2 can represent a transistor amplifier biasing circuit in the analog part of a mixed-signal system.

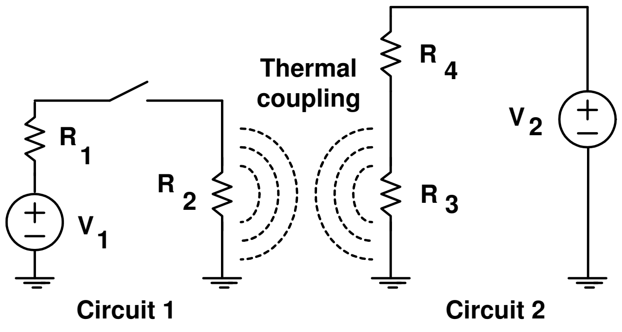


Fig. 1. Example of a circuit system with electrothermal coupling.

The electrothermal interaction that takes place in this system is illustrated in Fig. 2, where P is dissipated power, T is temperature, I is current, and V is voltage. Other devices are either not temperature sensitive, dissipate negligent power, or located off-chip. For electrothermal resistors R_2 and R_3, we will assume that their resistance is given by the following function of temperature [13]:

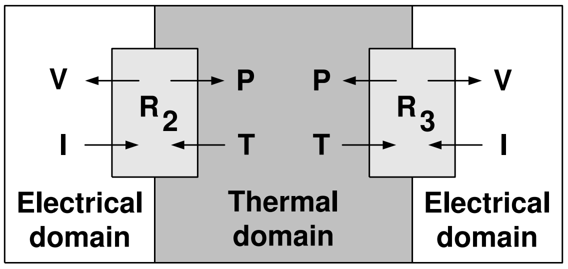


Fig. 2. Electrothermal interaction in the system shown in Fig. 1.

$$R = R_o \left[1 + \alpha(T_r - T_o)\right] \ , \qquad (1)$$

where R_o is the nominal resistance at a normal temperature of $T_o = 300$ K, T_r is the resistor temperature, and α is the temperature coefficient of resistance.

While the example presented here is rather trivial and somewhat artificial, it is a good conceptual demonstration of distributed electrothermal modeling in VHDL-AMS. We intentionally keep it simple to demonstrate clearly all steps involved into application of VHDL-AMS to an electrothermal problem. More complicated examples that include electrothermal semiconductor devices (capacitors, diodes, or transistors) and complex 3-D geometries but can be treated in a similar fashion.

III. Thermal Modeling

A. Geometry

Consider a rectangular substrate shown in Fig. 3 with two resistors R_2 and R_3 located on top of it. The width of the substrate is W, and the height of the substrate is H. Both resistors dissipate power and generate heat flux into the substrate. The currents through these resistors depend on their temperatures and change as the substrate temperature changes.

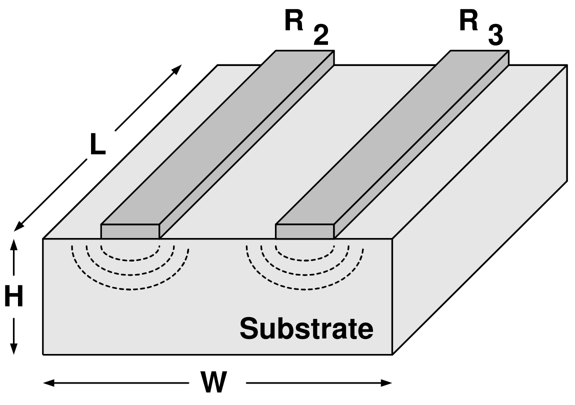


Fig. 3. Geometry of the thermal problem: resistors on top of a substrate.

Note that if the resistor length L is much greater than their width and mutual separation, this 3-D problem can be treated as 2-D.

B. Heat Balance Equation

Assuming that no heat is generated inside the substrate material, the temperature T inside the substrate can be found using the following heat balance equation:

$$\rho C \frac{\partial T}{\partial t} = \vec{\nabla} \cdot \left(k \vec{\nabla} T\right) \ , \qquad (2)$$

where ρ is the material density, C is the specific heat capacity, and k is the thermal conductivity.

Assume that the substrate material is uniform, and its properties are temperature-independent. Then we can rewrite (2) in Cartesian coordinates as:

$$\frac{\partial T}{\partial t} = \frac{k}{\rho C} \left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right) \ , \qquad (3)$$

where x and y are the transversal coordinates.

Boundary conditions on conductive boundary (contact surfaces between the substrate and resistors) and convective boundary (all other surfaces) are given by [18]:

$$k \frac{\partial T}{\partial \vec{n}} = \begin{cases} \dfrac{P}{A} \, , & \text{conductive} \\ h\left(T_a - T\right) , & \text{convective} \end{cases} \ , \qquad (4)$$

where $\vec{n}$ is the vector normal to the boundary surface, P is the power dissipated in a resistor, A is the area occupied by a resistor, and T_a is the ambient temperature.

C. PDE Discretization

A variety of numerical methods are available for solving thermal diffusion PDE'. For simplicity of implementation and clarity of illustration, we use a finite difference discretization with classical central difference formula Since VHDL-AMS does not currently support PDE's, we must discretize partial derivatives with respect to x and y.

Consider a rectangular $N \times M$ mesh, whose nodes are the points where the temperature needs to be determined. The mesh and the node labels are shown in Fig. 4. The horizontal and vertical spacings between the mesh points are $\Delta x = W/(N-1)$ and $\Delta y = H/(M-1)$.

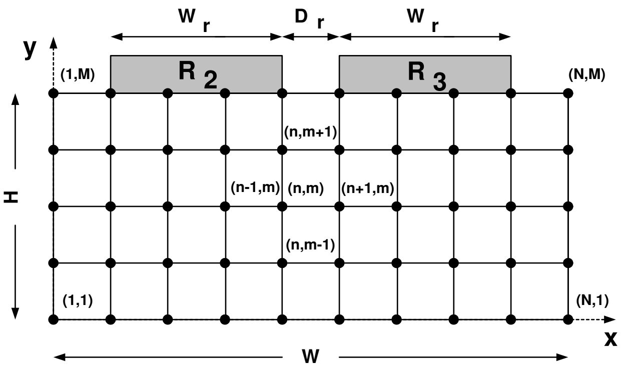


Fig. 4. Rectangular finite difference mesh.

Inside the material, (3) can be discretized to obtain:

$$\frac{\partial T}{\partial t} = \frac{k}{\rho C}\left[\frac{T_{n+1,m} - 2T_{n,m} + T_{n-1,m}}{\Delta x^2} + \frac{T_{n,m+1} - 2T_{n,m} + T_{n,m-1}}{\Delta y^2}\right]. \quad (5)$$

One can rewrite (5) as:

$$\frac{\partial T}{\partial t} = \frac{k}{\rho C\,\Delta x\,\Delta y} \times \left[\frac{T_{n+1,m} - T_{n,m}}{\Delta x}\Delta y - \frac{T_{n,m} - T_{n-1,m}}{\Delta x}\Delta y + \frac{T_{n,m+1} - T_{n,m}}{\Delta y}\Delta x - \frac{T_{n,m} - T_{n,m-1}}{\Delta y}\Delta x\right]. \quad (6)$$

In (6), one can identify inside the brackets several derivative terms which correspond to the horizontal and vertical temperature gradients. To take into account boundary conditions, the derivative terms of (6), which contain mode indices outside of $[1..N, 1..M]$ range are replaced with

$$\pm\frac{h(T_a - T_{n,m})}{k} \quad (7)$$

on convective boundaries or with

$$\pm\frac{P}{kA} \quad (8)$$

on conductive boundaries. The sign (plus or minus) depends on boundary location (left or right, top or bottom).

IV. VHDL-AMS Implementation

The discretization described in the previous section results in a system of $N \times M$ ODE's that can be solved in VHDL-AMS concurrently with the circuit equations. The temperature of each electrothermal resistor is computed by averaging the temperature over all grid points that lie on the resistor contact surface area. Below, we provide VHDL-AMS codes for our circuit system, the electrothermal resistor, and the silicon substrate material.

The values of the circuit elements were chosen such that the thermal coupling effect can be seen in the simulation. These values are: $V_1 = 10$ V, $R_1 = 10$ Ohm, $R_2 = 10$ Ohm (at 300 K), $V_2 = 2$ V, $R_3 = 10$ Ohm (at 300 K), $R_4 = 10$ Ohm. The VHDL-AMS model of the circuit system is shown below.

```
LIBRARY DISCIPLINES; LIBRARY IEEE;
USE DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL;
USE DISCIPLINES.THERMAL_SYSTEM.ALL;
USE IEEE.MATH_REAL.ALL;

ENTITY system IS END;

ARCHITECTURE behav OF system IS
    TERMINAL n1,n2,n3,n4: ELECTRICAL;
    TERMINAL t1,t2: THERMAL;

BEGIN
V1: ENTITY const_source (behav)
     GENERIC MAP (2.0)
     PORT MAP (n1,electrical_ground);
R1: ENTITY resistor     (behav)
     GENERIC MAP (10.0)
     PORT MAP (n1,n2);
R2: ENTITY th_resistor  (behav)
     GENERIC MAP (10.0,0.1)
     PORT MAP (n2,electrical_ground,t1);
R3: ENTITY th_resistor  (behav)
     GENERIC MAP (10.0,0.1)
     PORT MAP (n3,electrical_ground,t2);
R4: ENTITY resistor     (behav)
     GENERIC MAP (10.0)
     PORT MAP (n3,n4);
V2: ENTITY pulse_source (behav)
     GENERIC MAP (10.0,25.0e-6,50.0e-6)
     PORT MAP (n4,electrical_ground);
Si: ENTITY material     (behav)
     GENERIC MAP (300.0)
     PORT MAP (t1,t2);
END behav;
```

The electrothermal resistor model is similar to [13] and is shown below. While in reality the temperature coefficient of resistance is on the order of 0.001-0.01 (CMOS poly and n-well resistors), we chose the value $\alpha = 0.1$ K^{-1} to display stronger temperature dependence and to emphasize the electrothermal effect.

```
LIBRARY DISCIPLINES; LIBRARY IEEE;
USE DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL;
USE DISCIPLINES.THERMAL_SYSTEM.ALL;
USE IEEE.MATH_REAL.ALL;

ENTITY th_resistor IS
 GENERIC  (r_o, alpha: REAL);
 PORT (TERMINAL a, b  : ELECTRICAL)
 PORT     (TERMINAL t: THERMAL);
END;

 ARCHITECTURE behav OF th_resistor IS
 QUANTITY v_r ACROSS i_r THROUGH a TO b;
 QUANTITY t_r ACROSS h_r THROUGH t
                  TO thermal_ground;
 CONSTANT t_o : REAL := 300.0;

BEGIN
 i_r == v_r/(r_o*(1.0+alpha*(t_r - t_o)));
 h_r == i_r * v_r;
END behav;
```

The dimensions of the problem geometry were:

$$W = 45\,\mathrm{mil}\,, \tag{9}$$

$$H = 20\,\mathrm{mil}\,, \tag{10}$$

$$L = 150\,\mathrm{mil}\,, \tag{11}$$

$$W_r = 15\,\mathrm{mil}\,, \tag{12}$$

$$D_r = 5\,\mathrm{mil}\,, \tag{13}$$

where W_r is resistor width, D_r is resistor spacing, and 1 mil = 0.0254 mm. The mesh had the dimensions of $N = 10$ and $M = 5$.

We used the following values for the properties of the silicon substrate and its boundaries:

$$k = 1.412\left[\frac{W}{K\cdot cm}\right], \tag{14}$$

$$\rho = 2.33\left[\frac{g}{cm^3}\right], \tag{15}$$

$$C = 0.7\left[\frac{J}{g\cdot K}\right], \tag{16}$$

$$h = 1000\left[\frac{W}{cm^2\cdot K}\right]. \tag{17}$$

Above, large geometry size and high value of the convective heat transfer coefficient were chosen for illustrative purpose.

Resistor value can be estimated from:

$$R = R_s L / W_r\,, \tag{18}$$

where R_s is the resistance per square, or sheet resistance. In our example, the sheet resistance was set to be $R_s =$ 1 Ohm. In reality, sheet resistance in CMOS process would be approximately $R_s \approx$ 20 Ohm for poly and $R_s \approx$ 2000 Ohm for n-well resistors and the resistor dimensions would be smaller. For example, the size of 200 Ohm poly resistor could be 10 μm $\times$ 100 μm.

The VHDL-AMS model of the substrate is shown below. Because *Hamster* simulator does not support simultaneous *generate* statement, a separate *Matlab*[4] code was used to generate the VHDL-AMS code for the system of equations (6).

```
LIBRARY DISCIPLINES; LIBRARY IEEE;
USE DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL;
USE DISCIPLINES.THERMAL_SYSTEM.ALL;
USE IEEE.MATH_REAL.ALL;

ENTITY material IS
    GENERIC  (Ta: REAL);
    PORT     (TERMINAL t1,t2: THERMAL);
END;

ARCHITECTURE behav OF material IS
  QUANTITY t_1 ACROSS h_1 THROUGH t1
                 TO thermal_ground;
  QUANTITY t_2 ACROSS h_2 THROUGH t2
                 TO thermal_ground;
  CONSTANT k      : REAL := 1.412;
  CONSTANT rho    : REAL := 2.33;
  CONSTANT C      : REAL := 0.7;
  CONSTANT mil    : REAL := 0.001*2.54;
  CONSTANT Wr     : REAL := 15.0*mil;
  CONSTANT L      : REAL := 150.0*mil;
  CONSTANT Dr     : REAL := 5.0*mil;
  CONSTANT A1     : REAL := Wr*L;
  CONSTANT A2     : REAL := Wr*L;
  CONSTANT Width  : REAL := 45.0*mil;
  CONSTANT Height: REAL := 20.0*mil;
  CONSTANT h      : REAL := 1000.0;
  CONSTANT N      : REAL := 10.0;
  CONSTANT M      : REAL := 5.0;
  CONSTANT dx     : REAL := Width/(N-1.0);
  CONSTANT dy     : REAL := Height/(M-1.0);
  CONSTANT Const  : REAL := k/(rho*C*dx*dy);
  CONSTANT hk     : REAL := h*k;

  QUANTITY PkA1, PkA2: REAL;
  QUANTITY T11, T12, ... T104, T105:REAL;

BEGIN
BREAK T11 => Ta;
...
BREAK T105 => Ta;

PkA1 == - h_1 / (k * A1);
PkA2 == - h_2 / (k * A2);

T15'dot==Const*((T25-T15)*dy/dx-dy*hk*
   (T15-Ta)+dx*hk*(Ta-T15)-(T15-T14)*dx/dy);
T25'dot==Const*((T35-T25)*dy/dx-(T25-T15)*
             dy/dx+dx*PkA1-(T25-T24)*dx/dy);
...
T22'dot==Const*((T32-T22)*dy/dx-(T22-T12)*
     dy/dx+(T23-T22)*dx/dy-(T22-T21)*dx/dy);
...
T101'dot==Const*(dy*hk*(Ta-T101)-(T101-T91)*
   dy/dx+(T102-T101)*dx/dy-dx*hk*(Ta-T101));

t_1 == (T25 +T35 +T45 +T55 )/4.0;
t_2 == (T65 +T75 +T85 +T95 )/4.0;

END behav;
```

[4]Trademark of *Mathworks, Inc.*

V. Results

The example presented above was simulated using *Hamster*. The time step was chosen to be 0.1 μs to satisfy the stability criterion [19]:

$$\Delta t \le \frac{1}{2}\frac{\rho C}{k}\left(\frac{\Delta x^2 \Delta y^2}{\Delta x^2 + \Delta y^2}\right). \tag{19}$$

To see the effects of thermal coupling between the two circuits, a 10 V rectangular pulse of 50 μs duration was generated by turning switch on at $t = 25$ μs and then off at $t = 75$ μs.

Fig. 5 shows voltages on resistors R_2 and R_3. In the absence of the excitation pulse from V_1, as V_2 is turned on at $t = 0$ μs, the voltage on R_3 slowly rises due to self-heating until it reaches an equilibrium. With the pulse, the voltage of 10 V, generated by V_1, causes initially a voltage of 5 V to appear across R_2. The current heats R_2 up, causing the rise of its temperature and, respectively, voltage. Generated heat propagates through the substrate to R_3, increasing its temperature and voltage.

As one can see from Fig. 5, in both cases (with and without excitation pulse) curves for V_3 would look identical until a certain moment of time (≈ 30 μs), when the heat wave from R_2 reaches R_3.

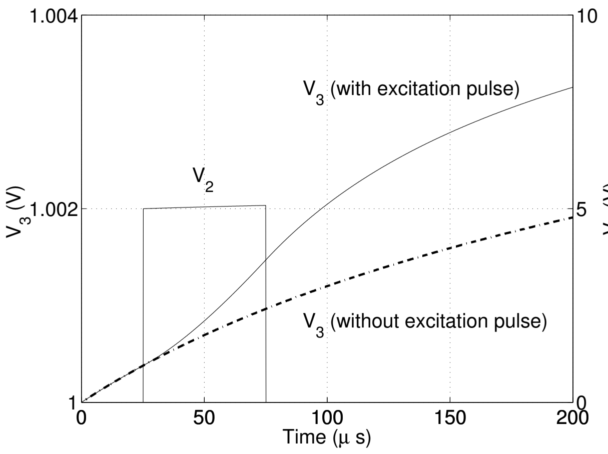


Fig. 5. Voltages on resistors R_2 and R_3 as functions of time.

Fig. 6 shows the calculated temperature distribution in the silicon substrate in the vicinity of resistor R_3 at the time moment $t = 200$ μs. Because of the higher current, resistor R_2 dissipates significantly more heat than R_3 and thus defines the temperature distribution inside the substrate.

VI. Discussion

One can see from Fig. 5 that electrothermal effect in our example is quite small: the change in R_3 voltage caused by it is less than 1%. Semiconductor devices, such as diodes or transistors, typically exhibit much stronger dependence on the temperature and result in more interesting electrothermal behavior.

The example described here can also be modeled with an equivalent thermal network, shown in Fig. 7, where temperature and heatflow play roles of voltage and current (capacitive storage effects are neglected). The heatflow

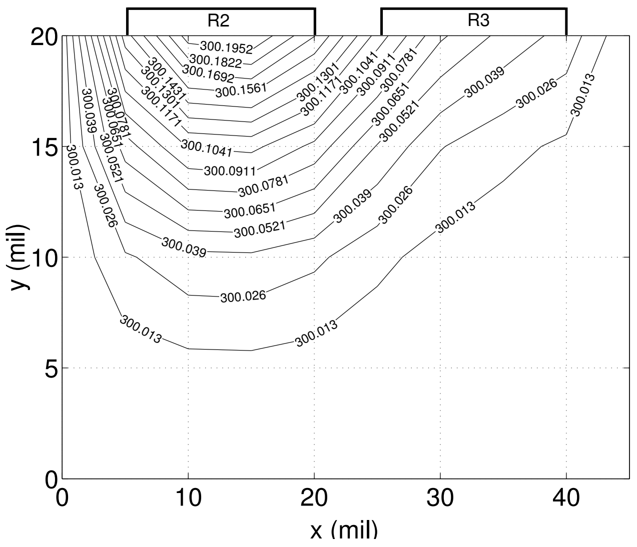


Fig. 6. Temperature distribution in the substrate at $t = 200$ μs.

of thermal current source associated with electrothermal resistor is equal to the power dissipated in the resistor. A thermal voltage source accounts for the temperature of T_a. Thermal resistors R_{t1} and R_{t2} represent mutual coupling between the resistors R_2 and R_3 and their heat dissipation.

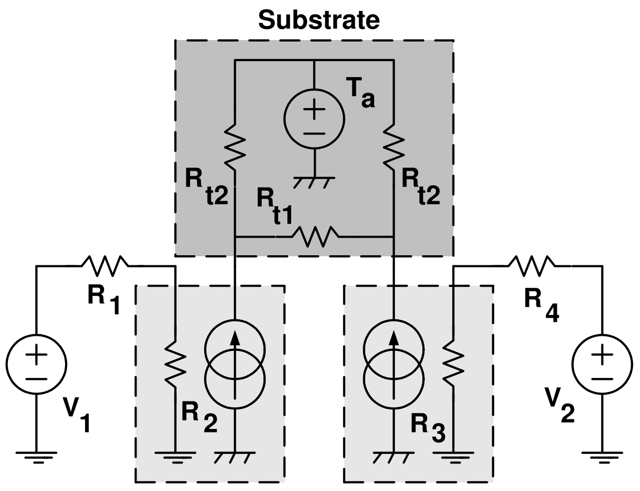


Fig. 7. Equivalent thermal network.

The system shown in Fig. 7 can easily be simulated, if the equivalent thermal resistor values R_{t1} and R_{t2} are known. For a large network, extracting those resistances is not trivial [20] and may require an application of model order reduction techniques [18].

In our example, we manually specified the geometry and the material properties, created a mesh, discretized the equations, and set boundary conditions. Ultimately, these tasks need to be automated, as it is done in many domain-specific simulators, such as *ANSYS*[5] or *FEMLAB*[6]. One can envision a graphical user interface that allows user to perform these functions for standard IC packages and to generate an appropriate set of VHDL-AMS codes.

Having PDE support in VHDL-AMS [17] would further help to simplify the process of modeling electrothermal problems. For example, with such support, heat balance equation (2) and its boundary conditions for one electrothermal element on top of a substrate could be directly implemented in VHDL-AMS and potentially look as follows:

```
BEGIN pde
T'dot==(k/rho*C)*(T''dot(x)+T''dot(y));
END pde;

BEGIN boundary1
T'dot(x) == (k*h/(rho*C))*(Ta-T);
T'dot(y) == (k*h/(rho*C))*(Ta-T);
END boundary1;

BEGIN boundary2
T'dot(y) == k*heatflow/(rho*C*A);
temp == average(T);
END boundary2;
```

where `boundary1` is convective boundary, and `boundary2` is conductive boundary. Note that boundary conditions for conductive boundary `boundary2` include a specification of how an electrothermal element interacts with the substrate, i.e. how quantities `heatflow` and `temp` are related to `T` (function `average(T)` gives the integrated average of the temperature on the contact surface of electrothermal element).

VII. Conclusions

In this paper, we demonstrated how to perform VHDL-AMS modeling of circuit systems with distributed electrothermal interaction. We presented an example where two resistors were thermally coupled via a silicon substrate. The heat balance PDE was discretized using a finite difference method, and the resulting system of ODE's was solved in VHDL-AMS together with the circuit equations using *Hamster* simulator.

While the example presented here may be simple, it nevertheless proves the concept: complex coupled electrothermal systems can in principle be modeled and simulated in a single unified language environment, such as VHDL-AMS. One major advantage of using the standard language is the fact that the model does not depend on the underlying simulator. This can lead to high portability of electrothermal models and ease of sharing them among the designers.

[5]Trademark of *ANSYS, Inc.*

[6]Trademark of *The COMSOL Group*

References

[1] T.-Y. Wang and C. C.-P. Chen, "3-D thermal-ADI: a linear-time chip level transient thermal simulator," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 21, pp. 1434–1445, December 2002.

[2] K. Fukahori and P. R. Gray, "Computer simulation of integrated circuits in the presence of electrothermal interaction," *IEEE Journal of Solid-State Circuits*, vol. 11, pp. 834–846, December 1976.

[3] R. K. Williams, M. Rodamaker, and L. Sevilla, "Electrothermal circuit simulation of power ICs combining SPICE and 3D finite element analysis," *Proceedings of the 4th International Symposium on Power Semiconductor Devices and ICs*, pp. 282–287, May 1992.

[4] L. Sang-Soo and D. J. Allstot, "Electrothermal simulation of integrated circuits," *IEEE Journal of Solid-State Circuits*, vol. 28, pp. 1283–1293, December 1993.

[5] S. Wunsche, C. Clauss, P. Schwarz, and F. Winkler, "Electro-thermal circuit simulation using simulator coupling," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 5, pp. 277–282, September 1997.

[6] A. R. Hefner and D. L. Blackburn, "Simulating the dynamic electrothermal behavior of power electronic circuits and systems," *IEEE Transactions on Power Electronics*, vol. 8, pp. 376–385, October 1993.

[7] E. Christen and K. Bakalar, "VHDL-AMS – a hardware description language for analog and mixed-signal applications," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 46, pp. 1263–1272, October 1999.

[8] B. F. Romanowicz, "Methodology for the modeling and simulation of microsystems," *Kluwer Academic Publishers*, 1998.

[9] W. Batty, C. E. Christoffersen, A. J. Panks, S. David, C. M. Snowden, and M. B. Steer, "Electrothermal CAD of power devices and circuits with fully physical time-dependent compact thermal modeling of complex nonlinear 3-d systems," *IEEE Transactions on Components and Packaging Technologies*, vol. 24, pp. 566–590, December 2001.

[10] W. Batty, C. E. Christoffersen, A. B. Yakovlev, J. F. Whitaker, A. Mortazawi, A. Al-Zayed, M. Ozkar, S. C. Ortiz, R. M. Reano, K. Yang, L. P. B. Katehi, C. M. Snowden, and M. B. Steer, "Global coupled EM-electrical-thermal simulation and experimental validation for a spatial power combining MMIC array," *IEEE Transactions on Microwave Theory and Techniques*, vol. 50, pp. 2820–2833, December 2002.

[11] C. Lallement, F. Pecheux, and Y. Herve, "VHDL-AMS design of a MOST model including deep submicron and thermal-electronic effects," *Proceedings of the Fifth IEEE International Workshop on Behavioral Modeling and Simulation*, pp. 91–96, 2001.

[12] L. Starzak, M. Zubert, and A. Napieralski, "The new approach to the power semiconductor devices modeling," *Proceedings of Fifth International Conference on Modeling and Simulation of Microsystems*, pp. 640–644, 2002.

[13] X. Huang and H. A. Mantooth, "Event-driven electrothermal modeling of mixed-signal circuits," *Proceedings of IEEE/ACM International Workshop on Behavioral Modeling and Simulation*, pp. 10–15, 2000.

[14] C.-J. Shi and A. Vachoux, "VHDL-AMS design objectives and rationale," *Current Issues in Electronic Modeling, Kluwer Academic Publishers*, vol. 2, pp. 1–30, 1995.

[15] X. Zhou, B. Liu, and B. Jammes, "Solving steady-state partial differential equation with neural network," *Proceedings of 8th International Conference on Neural Information Processing*, November 2001.

[16] G. Digele, S. Lindenkreuz, and E. Kasper, "Fully coupled dynamic electro-thermal simulation," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 5, pp. 250–257, September 1997.

[17] P. V. Nikitin, C. J.-R. Shi, and B. Wan, "Modeling partial differential equations in VHDL-AMS," *IEEE SoC Conference*, 2003, in press.

[18] J. T. Hsu and L. Vu-Quoc, "A rational formulation of thermal circuit models for electrothermal simulation - part I: Finite element method; part II: Model reduction techniques," *IEEE Transactions on Circuits and Systems*, vol. 43, no. 9, pp. 721–744, 1996.

[19] M. N. Ozisik, "Finite difference methods in heat transfer," *CRC Press*, 1994.

[20] D. J. Walkey, T. J. Smy, R. G. Dickson, J. S. Brodsky, D. T. Zweidinger, and R. M. Fox, "Equivalent circuit modeling of static substrate thermal coupling using VCVS representation," *IEEE Journal of Solid-State Circuits*, vol. 37, pp. 1198–1206, September 2002.

Parametric Equivalent Circuit Extraction for VLSI Structures

Pavel V. Nikitin, Winnie Yam, and C.-J. Richard Shi
Department of Electrical Engineering
University of Washington
Seattle, WA 98195-2500, USA
Email: {nikitin, wyam, cjshi}@ee.washington.edu

Abstract

Equivalent circuit modelling is a powerful technique widely used for time-domain simulation of complex electromagnetic VLSI structures. Surprisingly, parametric aspect of equivalent circuit modelling has not received much attention until recently (although the need for it has been previously advocated in several publications). Having a circuit with element values given as functions of the structure geometrical parameters eliminates the need to recalculate S-parameters and extract an equivalent circuit again whenever the geometry is modified.

The purpose of this paper is to discuss a concept of parametric equivalent circuit modelling for VLSI structures, to systematically describe a methodology of extracting such circuit from the given set of S-parameters, and to provide an overview of methods and problems arising at each step with referring to existing publications. For demonstration of the parametric equivalent circuit extraction, we use a classical example of a microstrip interconnect represented as an RLCG circuit[1].

1. Introduction

Various structures that exhibit electromagnetic (EM) behavior (inductors, connectors, interconnects, etc.) have always been an important part of microwave circuits. Now they play an important role in many modern VLSI systems-on-chips and seriously affect their performance, especially at multi-gigahertz frequencies. Typically, such structures are measured [1] or simulated in frequency domain using various EM simulators [2].

There exist a great variety of numerical electromagnetic field solvers that allow modelling of on- and off-chip structures. However, electromagnetic simulations are usually computationally intensive and are mostly used for verification rather than for design and synthesis, when one needs to vary parameters many times in the process of optimization.

There are two ways to use S-parameters (or other frequency domain parameters) obtained from EM simulations in SPICE-like time-domain circuit simulators. Most common approach is to extract an equivalent circuit whose S-parameters match those obtained from EM modelling [3, 4]. Many different techniques on implementing this approach exist in the literature [5]. An alternative approach is to perform an inverse Fourier transform on the S-parameters and then do a recursive convolution with the circuit time-domain response [6].

When structure's geometry changes (e.g. in the process of parasitic-aware layout optimization) new S-parameters must be obtained from electromagnetic simulation and new circuit values must be extracted. If the range in which EM structure parameters can vary is known, EM simulations can be carried beforehand to create a parametric table of values for equivalent circuit. This parametric tabulating capability is already present in several commercial EM software products (e.g., Ansoft's *Optimetrics* engine). One can expect that the next logical step is to use extracted parameter values to obtain circuit elements in a functional form for later use by a circuit designer.

Surprisingly, this subject has not received much attention in the CAD literature until recently. A good representative paper on the subject has been published by Sercu and Demuynck [7], who emphasized an integration of circuit simulation, EM simulation, and optimization tool. There exist several other more narrow-focused publications that address, e.g. parametric modelling of microstrip discontinuities [8].

As it is known, analytical models that relate, e.g. capacitance of a microstrip interconnect to its width and dielectric thickness, exist only for simple geometries [9, 10]. Especially when the parasitics effects become significant, no systematic methods exist to extract models that incorporate parasitics for general

[1]This research was supported by DARPA NEOCAD Program under Grant No. N66001-01-8920 and NSF CAREER Award under Grant No. 9985507

structures. Having an equivalent circuit whose element values are functions of the structure parameters does not only allow one to perform a faster circuit simulation and optimization but also provides a designer with a physical insight into the structure's behavior.

In this paper, we give a systematic description of the parametric equivalent circuit extraction process and discuss all related advantages and difficulties. We illustrate this process with a classical example of a microstrip interconnect with a variable strip width.

2. Parametric Equivalent Circuit Extraction Methodology

The process of parametric equivalent circuit extraction is illustrated in Figure . Geometrical parameters of the structure of interest are specified in the range of interest determined by the layout design rules (parameter step must be small enough not to miss important frequency response features, such as resonances). The structure is then modelled with an appropriate EM simulation tool in the frequency band of interest for each parameter value. An equivalent circuit is then extracted from these data. Parametric aspect of this process that we propose to explore is the last step, when circuit element values are approximated as functions of the structure parameters. The last three stages of the process shown in Figure are described below with more details.

2.2. EM modelling and S-parameters

As mentioned before, a variety of numerical electromagnetic field solving tools have been developed in the past, all of which have different limitations, capabilities, input and output formats, and computational costs. Choosing the best tool for a particular task and successfully employing and integrating it into a VLSI CAD design flow are challenging tasks. Most EM tools are based on three major methods and their flavors – method of moments (e.g., *Sonnet* by Sonnet Technologies), finite element method (e.g., *HFSS* by Ansoft Corporation), and finite-difference time domain method (e.g., *XFDTD* by Remcom, Inc.).

A number of equivalent parameters can be used to describe an arbitrary N-port device, such as S, Z, Y, $ABCD$, etc. Frickey [11] provides an excellent overview of various parameters and relationships between them: impedance matrix Z, admittance matrix Y, hybrid matrix h, chain matrix $ABCD$, scattering matrix S, and chain transfer matrix T.

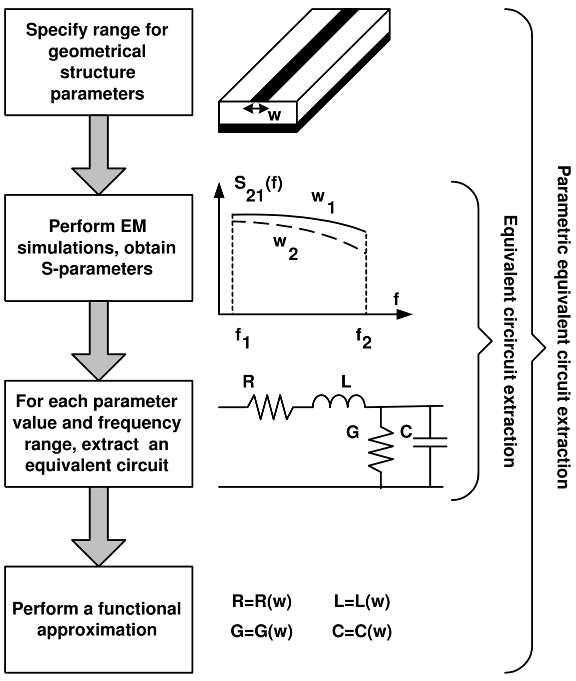

Specify range for geometrical structure parameters
w
Perform EM simulations, obtain S-parameters
S21(f)
w1
w2
f
f1
f2
For each parameter value and frequency range, extract an equivalent circuit
R
L
G
C
Perform a functional approximation
R=R(w)
L=L(w)
G=G(w)
C=C(w)
Equivalent circircuit extraction
Parametric equivalent circuit extraction

Figure 1: Parametric equivalent circuit extraction methodology.

S-parameters are the most popular way of modelling a device in frequency domain. They can be obtained directly from EM simulations and are typically computed for a device terminated with 50 Ohm loads.

2.2. Circuit structure

A crucial assumption for any equivalent circuit extraction approach is the knowledge of the circuit structure whose element values are to be extracted. This knowledge usually comes from a physical insight [8] or from the shape of the S-parameter frequency response. A large number of equivalent circuits are known and used for common structures like spiral inductors or bent interconnects. For example, a two port structure device can generally be modelled with a π-circuit [12].

Another approach to finding an equivalent circuit structure is genetic algorithm-based search [13], but its speed and convergence to correct circuit structures are currently the limiting factors of its applicability.

An assumed circuit structure is usually valid only for a certain frequency range. For example, the number of sections in an equivalent ladder circuit for an interconnect depends on the interconnect length with respect to the minimum wavelength of interest. The validity of one-stage lumped-circuit approximation breaks when the interconnect length becomes comparable to the quarter of a wavelength [14]. As the frequency increases, more stages need to be added.

Many existing EM tools have a built-in equivalent circuit extraction capability that is applied separately to each frequency point. As a result, circuit element values are frequency-dependent and change from one frequency point to another.

2.3. Objective function minimization

A general objective of equivalent circuit extraction is to find a set of circuit element values that results in a good match between the S-parameters of the circuit and the S-parameters of the given structure [13, 15].

For a two-port structure, the simplest objective function whose minimization delivers this match is [13]:

$$F = \sum_{i,j=1}^{2} \sum_{n=1}^{N} \left| S_{ij}^{M}(\omega_n) - S_{ij}(\omega_n) \right|^2 \,, \tag{1}$$

where S_{ij}^{M} are the S-parameters obtained by EM modelling, S_{ij} are the S-parameters of an equivalent circuit, and $\omega_1 ... \omega_N$ are the discrete frequency points. Since the structure of the circuit is assumed to be known, its S-parameters, and, hence, the objective function can also be found in analytical form either by hand or using symbolical methods.

To find the set of circuit parameters that minimizes the objective function, various optimization methods can be used [16]. Most of the methods are gradient-based. All methods require initial values for circuit parameters to be specified.

One of the most popular gradient-based methods is steepest descent method [5]. This method is based on moving in the direction opposite to the gradient of the objective function. The process is repeated at the new point and the algorithm continues until a minimum is found.

One commonly used parameter updating algorithm for steepest descent method is the linear algorithm:

$$\vec{\alpha}_{n+1} = \vec{\alpha}_n - \eta \, \vec{g} \,, \tag{2}$$

where $\vec{\alpha}$ is the multi-dimensional vector of circuit parameters, $\vec{g}$ is the gradient vector and η is commonly referred to as the learning rate and determines the convergence of the process.

The gradient itself can be computed in two ways: symbolically and numerically. Numerical approach to gradient computation is the most popular one as it does not require symbolic derivatives computation for the given circuit. If a symbolical expression is available for the objective function, the gradient vector $\vec{g}$ can also be found in an analytical form.

One should keep on mind that a general problem of optimization in a multi-parameter space is the presence of multiple minima in the objective function. There may be several possible circuit parameter combinations that result in a very small value of objective function and lead to solution ambiguity. This problem is well known and has been discussed in literature [15].

For parametric circuit extraction the uniqueness of solution is especially important: the same conformal objective function minimum must be used for each S-parameter set in order to ensure a proper functional behavior of the circuit parameters vs. structure parameters.

2.4. Parametrics

As mentioned before, tabular parametric ability is present in many commercial simulators. For parametric analysis, a range of structure parameters is specified beforehand. For each parameter set, S-parameters are obtained from EM simulations and equivalent circuit extraction process (objective function minimization) starts. Once the minimum of objective function is found, the obtained equivalent circuit element values are tabulated and the process is repeated for all parameter sets in the given range.

The next logical step that can be performed is a functional approximation – to have circuit element values approximated as analytical functions of the structure parameters. This would give a designer an insight into a structure's physical behavior and eliminate the need to recalculate S-parameters and extract an equivalent circuit again whenever the geometry is modified.

To perform a functional approximation, the type of basis analytical functions, such as polynomials, has to be specified. While the behavior of some structures may be complicated and involve logarithms and exponents, polynomial basis is useful and often sufficient to fairly approximate the first few terms of the Taylor expansion of unknown functions. Other basis functions, such as exponentials, can also be used.

3. Example

For demonstration of parametric equivalent circuit concept, we consider a simple microstrip interconnect line example shown in Figure . The interconnect is 160 mil long (1 mil=0.0254 mm) and consists of

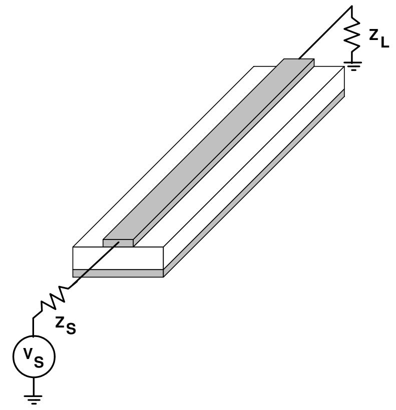


Figure 2: Physical structure of microstrip interconnect.

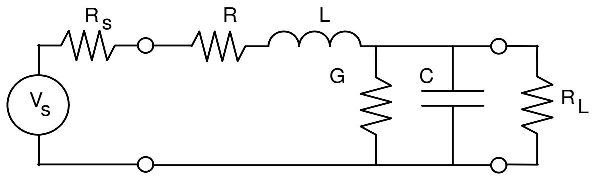


Figure 3: RLCG equivalent circuit representation of microstrip interconnect.

aluminum trace of height $t = 1$ mil and width w on top of alumina substrate of thickness $h = 25$ mil and relative dielectric permittivity $\varepsilon_r = 9.9$. The line is connected to a voltage source V_s, and the source and load impedances are $R_s = R_L = 50$ Ohm.

Such interconnect can typically be modelled as an RLCG ladder network shown in Figure . If the length of an interconnect is small compared to a wavelength, it can be represented as one lumped RLCG section. The transfer function (S_{21}) of the RLCG circuit shown in Figure is given by:

$$S_{21}(\omega) = \frac{1}{1 + (R + j\omega L)\left(R_L^{-1} + G + j\omega C\right)} . \quad (3)$$

Let us use *Sonnet* as a designer's EM tool of choice. *Sonnet* uses a 1/20 λ criteria: if the structure is larger than this size, it has to be modelled by parts which are then cascaded To satisfy this criteria for our interconnect length of 160 mil, we will limit the frequency range of consideration to 1 GHz.

Let us choose the width w as the geometrical parameter to be varied and use S_{21} for equivalent circuit objective function minimization. For demonstration purposes, we will select the following range of interconnect widths: 20 - 50 mil (with a step of 10 mil). Using *Sonnet*, we can perform EM simulation and obtain S_{21} responses, which are shown in Figure .

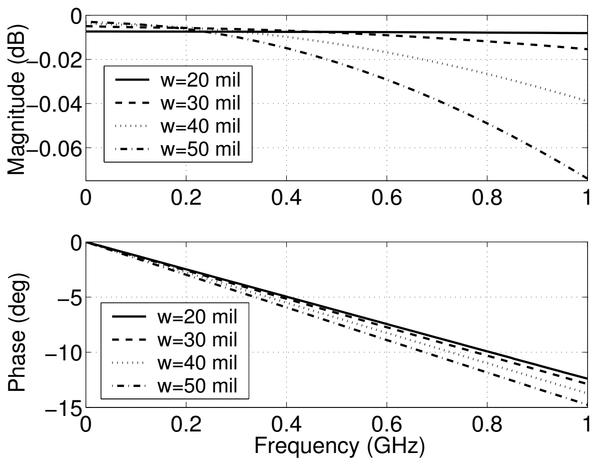


Figure 4: Magnitude and phase of S_{21} obtained with *Sonnet* for different interconnect widths.

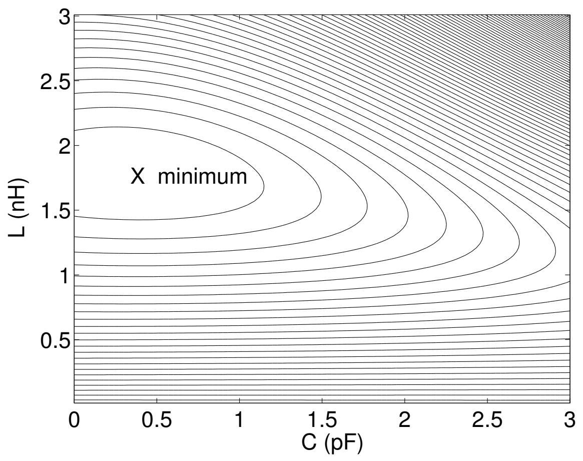


Figure 5: Contour plot of the objective function vs. L and C.

For illustration of the methodology, we created a simple optimization tool based on the steepest descent method, where the gradient of the objective function is computed numerically. Assume that two circuit parameters (L and C) are unknown and need to be extracted. The objective function is given by (1). Figure shows the contour plot of the objective function vs. L and C for $w = 30$ mil. One can see that there is a minimum. The exact at values of L and C at minimum, found from running an optimization tool, are L =1.76 nH and C =0.34 pF.

As mentioned before, initial values are needed for optimization process. Usually, a designer has an approximate idea of the order of magnitude of initial values. We use as initial estimate the following intuitive values: $L = 1$ nH, $C = 1$ pF. We also assume that conductance G and resistance R are small

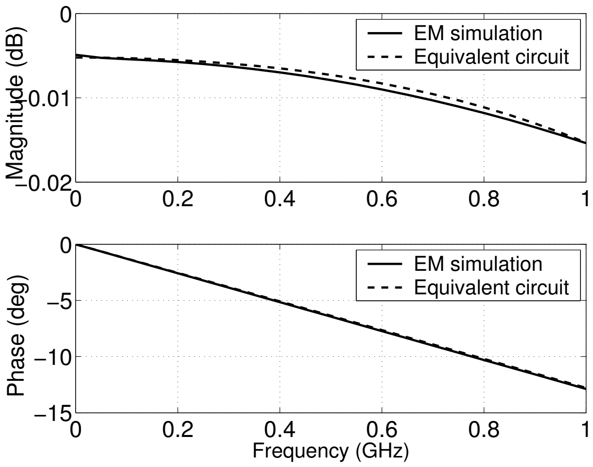


Figure 6: Magnitude and phase of S_{21} obtained with *Sonnet* ($w = 30$ mil) and approximated with an equivalent circuit .

but approximately known (for 30 mil wide trace they are: $R = 0.03$ and $G = 1e-7$). In this specific case, our optimization tool, written in *Matlab*, takes about 1.5 s to run on a 2.5 GHz PC and stops after 180 iterations, when the objective function becomes less than $4 \cdot 10^{-5}$. For each of the four frequency responses, the steepest descent algorithm converged to the same conformal minimum of the objective function.

A typical comparison between the original frequency response, obtained with *Sonnet*, and the frequency response of the equivalent circuit is given in Figure (the width of the interconnect is 30 mil). One can see that the agreement between frequency responses is very good.

Extracted L and C were approximated as functions of width w in the vicinity of $w_o = 30$ mil using the *polyfit* function in MATLAB for first-order polynomials, which gave the following dependence of L and C (per unit length) on the interconnect width w:

$$L \approx L_o + a\,(w_o - w),\; C \approx C_o + b\,(w - w_o), \quad (4)$$

where L_o and C_o are inductance and capacitance for the width w_o and a and b are constant coefficients.

In our specific case, the geometry was simple and well known. Analytical expressions given by (4) and the related coefficients could be obtained directly by writing the first two terms in the Taylor expansion of established analytical formulas for microstrip impedance and capacitance (see e.g. [17]). However, as mentioned before, for more complex geometries analytical formulas do not exist and the advantage of extracting a parametric equivalent circuit is obvious.

4. Discussion

The main advantage of the presented methodology is that functional approximation gives a designer access to equivalent circuit parameters as functions of geometrical and material parameters of the structure, even for those structures for which analytical model is not available. Another advantage is that it provides values for a continuum of geometrical parameters of the structure, not only for a discrete set available from the table or library.

The main drawback of the proposed methodology is the necessity of carrying multiple EM simulations beforehand, which means that a designer must specify or adaptively change the range of structure parameters that he is interested in and the number of points to be used, which determines the runtime. The accuracy of parametric equivalent circuit extraction strongly depends on the accuracy of EM simulator.

One issue to be aware of is that for wideband structures, the equivalent circuit is frequency-dependent. Having one circuit structure that is valid throughout the whole multi-gigahertz band would be ideal but may not be possible due to different frequency-dependent physical effects that take place in a structure (e.g., skin effect, proximity effect, etc.). Many of existing EM simulators are conservative in that regard. As mentioned before, *Sonnet* uses a 1/20 λ criteria to determine the maximum size of structure that can be approximated with one lumped circuit section. This can be viewed as an inter-dependence of frequency and geometrical parameters of VLSI structure.

The parametric methodology can be applied to all variables associated with a VLSI structure (geometrical parameters, electrical parameters, and frequency) to create, e.g., a library of equivalent circuits whose electrical parameters and frequency range of validity are given as functions of geometrical parameters and vice versa.

If a VLSI structure contains multiple ports, the minimization of the objective function has to be performed over a multi-port S-parameter matrix. The circuit structure in this case is usually more involved and equivalent circuit extraction process is more challenging compared to two-port devices.

The described functional parametric methodology can easily be integrated into most existing commercial EM simulators that have a built-in equivalent circuit export option. It can also be linked to a parametric and optimization engine already present in a simulator (and typically used for optimizing power, efficiency, reflection coefficient, or other system parameters).

5. Conclusion

In this paper, we described a methodology of extracting a functional parametric equivalent circuit from the set of S-parameters obtained via EM simulation for a VLSI structure with variable geometric parameters. We presented an overview of this methodology, discussed associated advantages and problems, and referred to existing publications in this area.

We demonstrated the methodology with the classical example of a straight microstrip interconnect, for which analytical capacitance and inductance models are available. The interconnect was modelled in *Sonnet* and represented as an RLCG network for equivalent circuit extraction using steepest descent optimization. The equivalent circuit element values were obtained as functions of the microstrip width.

The presented parametric approach gives a designer an insight into a physical behavior of the structure and can easily be integrated into existing EM simulators which have an equivalent circuit export option. It eliminates the need to recalculate the S-parameters whenever the layout is modified and can be very valuable for time-domain simulation and optimization of VLSI systems that include both circuit and EM structures coupled together.

6. References

[1] W. R. Eisenstadt and Y. Eo, "Determination of IC interconnect line parameters by S-parameter measurements," *Proceedings of IEEE Eighth International VLSI Multilevel Interconnection Conference*, pp. 356–258, June 1991.

[2] T. Wittig, T. Weiland, F. Hirtenfelder, and W. Eurskens, "Efficient parameter extraction of high-speed IC-interconnects based on 3D field simulation using fit," *Proceedings of International EMC Zurich Symposium*, 2001.

[3] A. E. Ruehli and A. Cangellaris, "Progress in the methodologies for the electrical modeling of interconnects and electronic packages," *Proceedings of IEEE*, vol. 89, pp. 740–771, May 2001.

[4] R. Achar and M. S. Nakhla, "Simulation of high-speed interconnects," *Proceedings of IEEE*, vol. 89, pp. 693–728, May 2001.

[5] J. Zhao, R. C. Frye, W. Dai, and K. L. Tai, "S parameter-based experimental modeling of high Q MCM inductor with exponential gradient learning algorithm," *IEEE Transactions on Components, Packaging, and Manufacturing Technology*, vol. 20, pp. 202–210, August 1997.

[6] N. Chang, L. Barford, and B. Troyanovsky, "Fast time domain simulation in SPICE with frequency domain data," *Proceedings of Electronic Components and Technology Conference*, pp. 689 –695, May 2001.

[7] J. Sercu and F. Demuynck, "Electromagnetic/circuit co-optimization of lumped component and physical layout parameters using generalized layout components," *2002 IEEE MTT-S International Microwave Symposium Digest*, vol. 3, pp. 2073 –2076, 2002.

[8] L. Roy, S. Labonte, and M. Li, "Microstrip discontinuity modeling for high-speed interconnects," *Canadian Conference on Electrical and Computer Engineering*, vol. 1, pp. 374–376, September 1995.

[9] E. Bogatin, "A closed form analytical model for the electrical properties of microstrip interconnects," *IEEE Transactions on Components, Hybrids, and Manufacturing Technology*, vol. 13, pp. 258 –266, June 1990.

[10] H. A. Atwater, "Tests of microstrip dispersion formulas," *IEEE Transactions on Microwave Theory and Techniques*, vol. 36, pp. 619–621, March 1988.

[11] D. A. Frickey, "Conversions between S, Z, Y, H, ABCD, and T parameters which are valid for complex source and load impedances," *IEEE Transactions on Microwave Theory and Techniques*, vol. 42, pp. 205–211, February 1994.

[12] Y. Cao, R. A. Groves, H. Xuejue, N. D. Zamdmer, J.-O. Plouchart, R. A. Wachnik, K. Tsu-Jae, and H. Chenming, "Frequency-independent equivalent-circuit model for on-chip spiral inductors," *IEEE Journal of Solid-State Circuits*, vol. 38, pp. 419 –426, March 2003.

[13] P. L. Werner, R. Mittra, and D. H. Werner, "Extraction of equivalent circuits for microstrip components and discontinuities using the genetic algorithm," *IEEE Microwave and Guided Wave Letters*, vol. 8, pp. 333–335, March 1998.

[14] J. Morsey and A. Cangellaris, "PRIME: passive realization of interconnect models from measured data," *Proceedings of IEEE Conference on Electrical Performance of Electronic Packaging*, pp. 47–50, 2001.

[15] M. H. Bakr, J. W. Bandler, and N. Georgieva, "An aggressive approach to parameter extraction," *IEEE Transactions on Microwave Theory and Techniques*, vol. 47, pp. 2428–2439, December 1999.

[16] J. W. Bandler, "Optimization methods for computer-aided design," *IEEE Transactions on Microwave Theory and Techniques*, vol. 17, pp. 533–552, August 1969.

[17] D. Brooks, *Signal Integrity Issues and Printed Circuit Board Design.* Prentice-Hall, 2003.

Modeling Partial Differential Equations in VHDL-AMS

Pavel V. Nikitin, C.-J. Richard Shi, and Bo Wan
Department of Electrical Engineering
University of Washington
Seattle, WA 98195-2500, USA.
Email: {nikitin, cjshi, wanbo}@ee.washington.edu

Abstract— **This paper discusses a problem of modeling distributed physics effects described by partial differential equations (PDE's) in VHDL-AMS, a powerful modeling language for mixed-signal systems. First, we summarize the requirements for PDE support. Second, we demonstrate with the example of a distributed transmission line how to model PDE's in an existing VHDL-AMS by applying spatial discretization to system equations. Third, we propose a language extension needed to support PDE's. Our work should be perceived as a first step towards an accurate description and modeling of coupled multi-physics systems in VHDL-AMS.** [1]

I. Introduction

An IEEE standard, VHDL-AMS is a powerful hardware description language that allows one to model the behavior of mixed-signal (analog and digital) and multi-physics (mixed electrical, electromagnetic, thermal, mechanical, etc.) systems [1], [2], [3]. VHDL-AMS specifies what system of equations is to be used at each simulation time but the choice of a solution technique is left to an implementor. Continuous parts of the system can currently be described in VHDL-AMS using differential and algebraic equations (DAE's). Due to the complexity, the support for partial differential equations (PDE's) was intentionally left out in VHDL-AMS [4]. This limits the accurate modeling of system blocks that include distributed physics effects.

Such blocks are currently modeled in VHDL-AMS exclusively via equivalent circuit approach [5] or reduced order models [6] imported from an accurate solution obtained by an external domain-specific simulator [7].

A proposition to extend the capability of VHDL-AMS to support full-wave modeling of distributed RF and microwave components has recently appeared in the literature [8]. This is a challenging task and to the best of our knowledge no other publications have followed yet. The only other published work in this direction was an earlier paper by Zhou et al. [9] who solved a steady-state PDE in VHDL-AMS with a neural network algorithm.

The purpose of this paper is to define the first step towards modeling distributed physics effects in VHDL-AMS – to introduce a language support for PDE's. The importance of such support in a universal hardware description language cannot be overestimated and has been discussed earlier during the development of a microwave hardware description language (MHDL) [10], [11].

[1]This research was supported by DARPA NeoCAD Program under Grant No. N66001-01-8920 and NSF CAREER Award under Grant No. 9985507

We present an example of a distributed transmission line connected to a circuit and show how to model such system in an existing language. We also demonstrate how it can be modeled using a language extension proposed by us.

The remainder of the paper is organized as follows. Section II summarizes the requirements for PDE support. Implementation of PDE's in an existing VHDL-AMS standard for the transmission line example is shown in Section III. Section IV discusses a VHDL-AMS extension needed for PDE support. Conclusions are given in Section V.

II. Requirements for PDE support

To include a block described by PDE's into a VHDL-AMS system simulation, one needs to define:

1) PDE's that describe the physics of a problem
2) Parameters of the PDE's
3) Boundary conditions
4) Contact interface with the rest of the system

For example, a one-dimensional PDE can have a form:

$$\frac{\partial A}{\partial x} + \alpha(x,t)\frac{\partial A}{\partial t} = f(x,t)\,, \qquad (1)$$

where $A(x,t)$ is the quantity of interest, $\alpha(x,t)$ is the parameter, $f(x,t)$ is the excitation, x is a spatial variable, and t is time. To solve (1), we need to know $\alpha(x,t)$, which contains the information about material properties and geometry of the system, and the boundary conditions for $A(x,t)$, which also include the initial conditions.

If the system described by (1) is connected to a circuit, we need to define how the quantity $A(x,t)$ interacts with circuit quantities. Exact definition of the contact interface depends on the physics of the problem and may involve a translation, e.g., between electric and magnetic fields and voltages and currents [12], [13]. In VHDL-AMS, such interaction can be implemented using port and terminal definitions.

III. PDE's in existing VHDL-AMS

Since current VHDL-AMS does not support partial derivatives, the only way to implement PDE's in existing language is to discretize the equations with respect to spatial variables and leave the time derivatives to be handled by VHDL-AMS. The idea of a stand-alone spatial discretization has been used by several researchers before for solving PDE problems by creating and then solving equivalent circuits with SPICE and its likes [14], [15]. Using VHDL-AMS approach allows one to bypass the equivalent circuit step. It also makes possible a concurrent simulation of mixed-technology multi-physics problems, where PDE's and lumped circuits are mixed together. Below, we present an example that demonstrates this concept.

A. Transmission Line Example

Consider a system that consists of a distributed transmission line connected to a circuit as shown in Fig. 1.

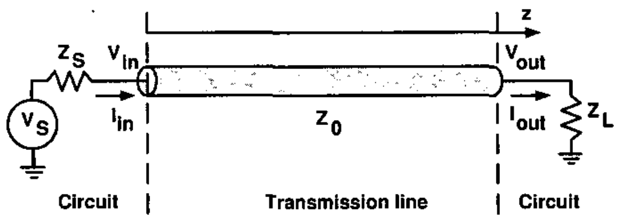


Fig. 1. Transmission line connected to a circuit.

The transmission line can represent an integrated circuit interconnect. The signal propagation on a transmission line can be described with the wave equation, which is a second-order PDE. The circuit is described by Kirchoff's current and voltage law equations.

Coupled problems similar to this one are usually treated by extracting an equivalent port model network for the transmission line and then using SPICE-like circuit simulator to solve for the whole system as a circuit [16] or by interfacing electromagnetic solver and circuit simulator [17].

The interaction between the transmission line and the circuit happens through the terminal voltages and currents: I_{in}, V_{in}, I_{out}, and V_{out}. In this example, the internal quantities of the distributed physics part (voltages and currents) are the same as circuit variables; thus no translation is needed. Boundary conditions that describe an interface to the circuit are:

$$V_{in} = V_S - I_{in} Z_S \tag{2}$$

$$V_{out} = I_{out} Z_L \, . \tag{3}$$

If the line is lossless, the wave equation has the form:

$$-\frac{\partial^2 V}{\partial z^2} - \beta^2 \frac{\partial^2 V}{\partial t^2} = 0 \, , \tag{4}$$

where V is the voltage on the transmission line and $\beta = \sqrt{l\,c}$ is the propagation constant (l and c are the inductance and the capacitance per unit length). The same problem can be equivalently formulated in terms of two Telegrapher's equations [18]:

$$\begin{cases} -\dfrac{\partial V}{\partial z} = l \dfrac{\partial I}{\partial t} \, , \\ -\dfrac{\partial I}{\partial z} = c \dfrac{\partial V}{\partial t} \, . \end{cases} \tag{5}$$

To discretize these equations with respect to z, one can use a classical central difference formula used in many finite difference techniques [19]. If the length of the transmission line is d, a spatial step of Δz results in $N+1$ points where $N = d/\Delta z$ and the voltage and the current need to be determined at each point. A set of two PDE's given by (5) can be converted into the following set of $2N$ ODE's:

$$\begin{cases} -\dfrac{V_n - V_{n-1}}{\Delta z} = l \, I_n' \, , \quad n = 1 ... N \\ -\dfrac{I_{n+1} - I_n}{\Delta z} = c \, V_n' \, , \quad n = 1 ... N \end{cases} \tag{6}$$

where V_n and I_n are currents and voltages at spatial points as shown in Fig. 2 and prime (') denotes a derivative with respect to time. Two additional equations are given by (3) and (3).

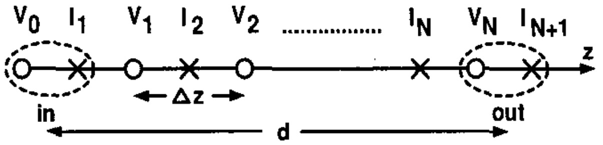


Fig. 2. Spatial finite difference grid.

Note that in this discretization scheme voltages and currents are not defined at the same points in space. This causes matching errors and may give rise to reflections on the transmission line even when all impedances are perfectly matched. The magnitude of the error depends on the discretization step. This effect is known [16] and usually requires an introduction of correction elements to eliminate the errors.

One can see that discretized equations for V and I on the transmission line are equivalent to circuit equations describing the equivalent N-section LC-ladder network shown in Fig. 3, where $L = l\Delta z$ and $C = c\Delta z$ are the inductance and the capacitance of each segment of the transmission line. N-section ladder network is usually valid only for a certain frequency range. The number of sections in the network depends on the transmission line length with respect to the minimum wavelength of interest. As the frequency increases, more stages need to be added. For digital circuits with sharp signal transitions, N needs to be large to accurately reproduce a wide-band response of the transmission line.

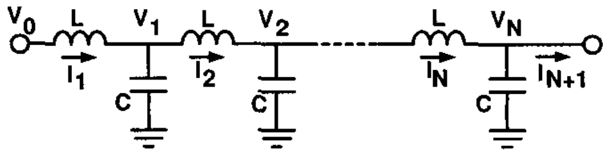


Fig. 3. Equivalent LC-ladder network.

B. VHDL-AMS Implementation

For the transmission line system of equations shown above, we have used two different VHDL-AMS implementations with two different VHDL-AMS simulators: freely available *Hamster*[2] and our own in-house *MCAST* [20].

[2]*Hamster* is now part of *Simplorer*, trademark of *Ansoft Corp.*

Current VHDL-AMS standard includes a language construct called "GENERATE" that in theory allows one to create a large set of simultaneous equations whose terms are array elements and their time derivatives. Boundary conditions and PDE parameter dependence on variables can be defined using array initialization. This is critical for efficient VHDL-AMS implementation of spatial discretization algorithms, which involve a transformation of a small set of PDE's into a large set of ODE's. Unfortunately, to the best of our knowledge the support for the simultaneous statements loop ("GENERATE" construct) is currently missing in many existing VHDL-AMS simulators.

Below we show the VHDL-AMS implementation of the transmission line model, which consists of an entity and an architecture. The entity and the first part of the architecture contain the description of line ports and parameters and are the same for both *Hamster* and *MCAST*. The part of the architecture that describes discretized transmission line equations is different for *Hamster* and *MCAST*.

In our example, PDE parameters l and c are constant, which corresponds to a homogeneous medium. The boundary conditions are given by (2) and (3), and no quantity conversion is needed because quantities of interest (voltages and currents) are the same for both circuit and transmission line.

```
-------- Transmission line begins ---------
LIBRARY DISCIPLINES; LIBRARY IEEE;
 USE DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL;
 USE IEEE.MATH_REAL.ALL;
ENTITY transmission_line IS
 PORT (TERMINAL a, b, g : ELECTRICAL);
END;
ARCHITECTURE behav OF transmission_line IS
 CONSTANT Lz     : REAL := 1.0;
 CONSTANT Cz     : REAL := 1.0;
 CONSTANT Length: REAL := 0.1;
 CONSTANT N      : REAL := 5.0;
 CONSTANT dz     : REAL := Length / N;
 CONSTANT C      : REAL := Lz • dz;
 CONSTANT L      : REAL := Cz *dz;
 QUANTITY Vin  ACROSS Iin  THROUGH a TO g;
 QUANTITY Vout ACROSS Iout THROUGH b TO g;
.........
```

One can see that the transmission line has three terminals: a, b, and g, which correspond to input, output, and ground. The terminal across and through quantities are voltages and currents: $V_{in} = V_0$, $I_{in} = I_1$, $V_{out} = V_N$, and $I_{out} = I_{N+1}$, where $N = 5$. For simplicity, the source and the load resistors were assumed to be $Z_S = 1$ Ohm and $Z_L = 1$ Ohm. The transmission line was taken to be 0.1 m long and have the parameter values $l = 1.0$ H/m and $c = 1$ F/m, which resulted in a characteristic impedance of $Z_0 = 1$ Ohm.

Below we present two implementations of the equations part of the transmission line problem. For *Hamster*, each equation had to be explicitly written out. *MCAST* supports simultaneous loops, which is advantageous for large N. Note that in both implementations I_{out} has a negative sign in front of it because of the VHDL-AMS definition of current flowing into a terminal.

Hamster implementation:

```
.........
 QUANTITY V1, V2, V3, V4: REAL;
 QUANTITY I2, I3, I4, I5: REAL;
BEGIN
 -(V1-Vin) == L * Iin'dot;
 -(I2-Iin) == C * V1'dot;
 -(V2-V1) == L * I2'dot;
 -(I3-I2) == C * V2'dot;
 -(V3-V2) == L * I3'dot;
 -(I4-I3) == C * V3'dot;
 -(V4-V3) == L • I4'dot;
 -(I5-I4) == C * V4'dot;
 -(Vout-V4) == L *I5'dot;
 -(-Iout-I5) == C * Vout'dot;
END;
-------- Transmission line ends ---------
```

MCAST implementation:

```
..........
 QUANTITY V:real_vector(0 to N-1);
 QUANTITY I:real_vector(2 to N);
BEGIN
 -(V(1)-Vin) == L * Iin'dot;
 -(I(2)-Iin) == C * V(1)'dot;
FOR i IN 2 TO N GENERATE
  -(V(i)-V(i-1)) == L * I(i)'dot;
  -(I(i+1)-I(i)) == C * V(i)'dot;
END GENERATE;
 -(Vout-V(N-1)) == L * I(N)'dot;
 -(-Iout-I(N)) == C * Vout'dot;
END;
-------- Transmission line ends ---------
```

Fig. 4 shows the input voltage V_{in} and the output voltage V_{out} for the transmission lines with $N = 5$ and $N = 20$ (other parameters are the same as described before) simulated both in *Hamster* and *MCAST*. The differences are due to truncation errors and the fact that two simulators use different integration methods. One can also see that the average delay of the response is approximately the same for both $N = 5$ and $N = 20$, but more high frequencies are present in the transient for $N = 20$, as one would expect.

IV. Extension for PDE support

Based on the example considered above, an extension for PDE support in VHDL-AMS can be considered. The extension would include a language operator $'dot(x)$, where x is a spatial variable. Such operator is currently non-existent in VHDL-AMS language standard. Choice of spatial discretization technique will be left to an implementor of the VHDL-AMS simulator, as it is now the case with time discretization. Equations for the transmission line example using such language extension would look as follows:

```
-V'dot(z) == Lz * I'dot
-I'dot(z) == Cz * V'dot
```

In perspective, one can also consider including in VHDL-AMS some generic operators, such as nabla operator ($\vec{\nabla}$). Together with vector and scalar multiplication operations and a coordinate system specification, this would enable one to cast

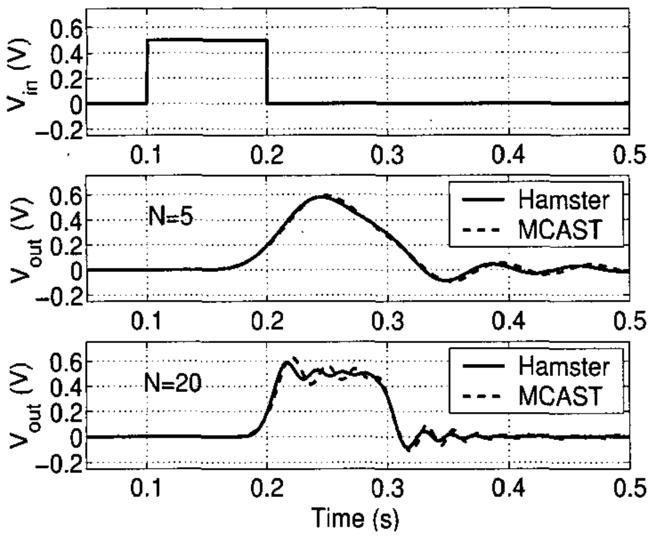


Fig. 4. Input and output voltages for the transmission line shown in Fig. 1 with $N = 5$ and $N = 20$ excited by a rectangular pulse.

many physical equations in VHDL-AMS in a very intuitive form.

Boundary conditions and PDE parameter dependence on variables can be defined using functional description. The contact interface to circuit in VHDL-AMS part can be defined via ports and terminals. Depending on the physics of the problem, the internal quantities of distributed physics blocks may need to be converted into voltages and currents.

The challenge in VHDL-AMS PDE modeling is going to be a realization of a simulator with built-in discretization schemes and solution techniques for different PDE's. Many numerical PDE solvers have already been developed for PDE's of different types and in different physical domains such as *FEMLAB*[3]. The results of PDE solvers research should definitely be used in development of the simulator. In order to be useful to the CAD industry, such simulator must be accurate and fast when applied to large scale multi-physics problems with many unknowns.

V. Conclusion

In this paper, we discussed the problem of modeling and simulation of distributed physics systems described by PDE's in VHDL-AMS. We summarized the requirements for PDE support, demonstrated with the example how to model PDE's in the existing language, and proposed a language extension to support simple PDE's.

This work should be perceived as a first step towards further extension of VHDL-AMS, which would allow one to model and simulate mixed-technology multi-physics systems, consisting of both distributed-physics and lumped-circuit parts. Such capability would facilitate portability, distribution, and exchange of various models between different designers even if a designer is not an expert in, e.g., electromagnetic or thermal modeling. This should greatly speed up an automated synthesis of complex systems-on-chips and can hopefully lead to a new language standard in the CAD industry.

[3]Trademark of *The COMSOL group*

References

[1] E. Christen and K. Bakalar, "VHDL-AMS – a hardware description language for analog and mixed-signal applications," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 46, pp. 1263–1272, October 1999.

[2] B. F. Romanowicz, "Methodology for the modeling and simulation of microsystems," *Kluwer Academic Publishers*, 1998.

[3] R. Shi, E. Christen, P. Liebmann, S. Krolikoski, and W. Zhou, "VHDL-A: analog extension to VHDL," *Proceedings of Seventh Annual IEEE International ASIC Conference and Exhibit*, pp. 19–23, September 1994.

[4] C.-J. Shi and A. Vachoux, "VHDL-AMS design objectives and rationale," *Current Issues in Electronic Modeling, Kluwer Academic Publishers*, vol. 2, pp. 1-30, 1995.

[5] C. Lallement, F. Pecheux, and Y. Herve, "VHDL-AMS design of a MOST model including deep submicron and thermal-electronic effects," *Proceedings of the Fifth IEEE International Workshop on Behavioral Modeling and Simulation*, pp. 91–96, 2001.

[6] X. Huang and H. A. Mantooth, "Event-driven electrothermal modeling of mixed-signal circuits," *Proceedings of IEEE/ACM International Workshop on Behavioral Modeling and Simulation*, pp. 10–15, 2000.

[7] L. Starzak, M. Zubert, and A. Napieralski, "The new approach to the power semiconductor devices modeling," *Proceedings of Fifth International Conference on Modeling and Simulation of Microsystems*, pp. 640–644, 2002.

[8] J. Willis and J. Johnson, "Language design requirements for VHDL-RF/MW," *IEEE MTT-S International Microwave Symposium Digest*, vol. 3, pp. 2093–2095, 2002.

[9] X. Zhou, B. Liu, and B. Jammes, "Solving steady-state partial differential equation with neural network," *Proceedings of 8th International Conference on Neural Information Processing*, November 2001.

[10] B. Cohen, "Partial differential equations in an MHDL," *MHDL requirements document*, 1991.

[11] D. L. Barton and D. D. Dunlop, "An introduction to MHDL," *IEEE MTT-S International Microwave Symposium Digest*, vol. 3, pp. 1487–1490, 1993.

[12] I. A. Tsukerman, A. Konrad, G. Meunier, and J. C. Sabonnadiere, "Coupled field-circuit problems: trends and accomplishments," *IEEE Transactions on Magnetics*, vol. 29, pp. 1701–1704, August 1992.

[13] N. Orhanovic and N. Matsui, "FDTD-SPICE analysis of high-speed cells in silicon integrated circuits," *Proceedings of Electronic Components and Technology Conference*, pp. 347–352, 2002.

[14] G. Kron, "Numerical solution of ordinary and partial differential equations by means of equivalent circuits," *Journal of Applied Physics*, vol. 16, pp. 172–186, March 1945.

[15] W. R. Zimmerman, "Time domain solutions to partial differential equations using SPICE," *IEEE Transactions on Education*, vol. 39, pp. 563–573, November 1996.

[16] W. Gwarek, "Analysis of arbitrarily shaped two-dimensional microwave circuits by finite-difference time-domain method," *IEEE Transactions on Microwave Theory and Techniques*, vol. 36, pp. 738–744, April 1988.

[17] V. Jandhyala, Y. Wang, D. Gope, and C.-J. Shi, "A surface-based integral-equation formulation for coupled electromagnetic and circuit simulation," *IEEE Microwave and Optical Technology Letters*, vol. 34, pp. 103–106, July 2002.

[18] M. N. Sadiku and L. C. Agba, "A simple introduction to the transmission-line modeling," *IEEE Transactions on Circuits and Systems*, vol. 37, pp. 991–999, August 1990.

[19] K. S. Yee, "Numerical solution of initial boundary value problems involving Maxwell's equations," *IEEE Transactions on Antennas and Propagation*, vol. 14, pp. 302–307, 1966.

[20] B. Wan, B. Hu, L. Zhou, and C.-J. R. Shi, "MCAST: An abstract-syntax-tree based model compiler for circuit simulation," *Proceedings of IEEE Custom Integrated Circuits Conference*, 2003.

MCAST User's Manual

Bo Hu

June 2004

Mixed-Signal CAD LAB, Department of Electrical Engineering
University of Washington

Version 1.0

System requirement

a. Solaris Operating System
b. gcc compiler
c. perl
d. tcsh shell

Installation on Sun Workstation

a. *mkdir vhdlams*
b. copy mcast.tar.gz and spice3f5.tar.gz to vhdlams/
c. *cd vhdlams/*
d. *tar zxvf mcast.tar.gz*
e. *tar zxvf spice3f5.tar.gz*
f. modify the mcast/.bmac source file to set up the correct environment variables
g. installation completed

Compile VHDL-AMS model and installation of new device model

a. *souce vhdlams/mcast/.bmac*
b. *mkdir run/*
c. *cp modelfile run/*
d. *cd run/*
e. *$BMACROOT/bin/bmac modelfile.vhdl*
f. *./install.spice3f5*

Compile spice3f5 with new model

a. go to spice3f5/
b. *./util/build sun4 gcc*
c. after successful compilation, new spice3* binary will be located in spice3f5/src/bin/, and it will be able to simulate the new device.

Automatic Analog Layout Retargeting for New Processes and Device Sizes

Nuttorn Jangkrajarng, Sambuddha Bhattacharya, Roy Hartono, and C.-J. Richard Shi

Department of Electrical Engineering, University of Washington
Seattle, WA 98195 USA
{njangkra,sbb,rhartono,cjshi}@ee.washington.edu

ABSTRACT – *This paper presents an automatic analog layout resizing tool that can generate a new layout incorporating the target technology process and the target transistor sizes. The tool automatically preserves the analog layout integrity by extracting layout symmetry and matching, and then solving the constrained layout generation problem using a combined linear programming and graph-theoretic approach. The tool has been applied successfully to integrate specified transistor sizes and to migrate layouts for various analog designs from TSMC 0.25um CMOS to TSMC 0.18um CMOS process with comparable performances to re-design.*

1. Introduction

The scaling of feature size in VLSI circuits, both digital and analog, has been one of the strongest driving forces toward the rapid development of electronics technology. For digital circuits, the shrinkage of transistor sizes (for example from 0.25um to 0.18um to 0.13um) is the main reason microprocessors rapidly increase in speed. In the analog or mixed signal layouts, the circuit performances also benefit from smaller minimum feature size.

When there is a change in technology process, digital designers can utilize benefits from the new technology without much effort by using existing high-level VHDL or Verilog designs, scalable cell libraries, and readily available automatic place & route tools to generate a new circuit layout with better performances, or by layout retargeting tools. In contrary, analog designers do not have the comparable ability, which means they have to go through a full time-consuming cycle of redesigning, testing and drawing layouts. Therefore, an automated tool for re-layout of analog circuits will be essential in significantly reducing the design time for mixed-signal circuit technology migration.

In this paper, we present, for the first time, a computer-aided design tool that can automatically resize an existing analog layout for some modestly new processes. The tool is developed based on the existing algorithms [2-5] related to layout compaction. Its interface is shown in figure 1. The automatic analog layout resizer reads an original layout and its technology file, a new target technology file, and new transistor sizes. Then, it automatically generates a new layout that satisfies all the design rules while preserving all the analog layout integrities such as matching and symmetry constraints. The methodology we propose here is based on "recycling" already fine-tuned analog layouts. As high performance analog circuits are layout-sensitive and require device/wiring alignment, matching and symmetry, the method of recycling the layouts will be able to conserve the above requirements. Moreover, it will preserve all the unique aspects intended by engineers on any particular layout.

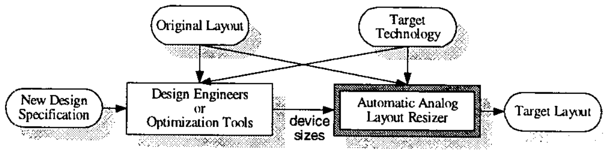


Figure 1: *Interface of the automatic analog layout retargeting tool.*

This paper is structured as follows. Section 2 presents the proposed analog layout retargeting flow and the implementation of important sections. Section 3 describes results using this new retargeting tool. Section 4 concludes the paper.

2. Automatic Layout Resizing Procedure

In order to automatically retarget an analog layout to a new technology process, three main considerations - namely new technology restriction, new device sizes, and layout structure preservation - have to be taken into account. The original design is used as a starting point for our program, with the purpose of maintaining the layout property. Our approach consists of layout representation and extraction [1,2], symmetry detection [3], technology conversion, circuit components resizing, and layout compaction [2,5]. The flow is shown in Figure 2, which important sections are described as follow.

2.1 Layout Representation and Symmetry Detection

First, the layout is represented by the corner stitching data structure [1], which recognizes each rectangle and its neighbors for every layer. The transistors and nets are then extracted from the layout.

The symmetry axes can be detected automatically between transistor pairs, based on the algorithm from [3]. However, this method may create many unnecessary symmetry axes. To overcome this problem, we introduce a user-specified option, which instructs the detection function to check and compare only

* This research was supported by NSF-ITR and DARPA NeoCAD grant.

between specified transistors or blocks. The symmetry axes then become one of the main criteria that the retargeting tool has to maintain.

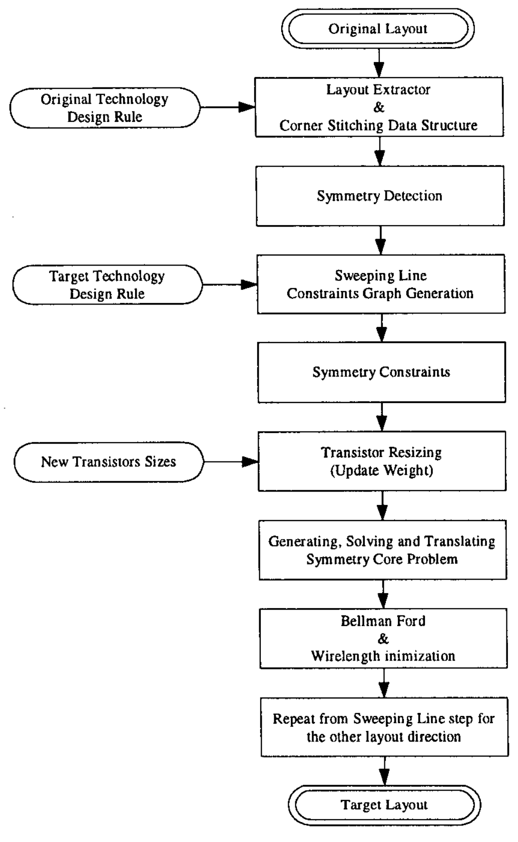

Original Layout
Original Technology Design Rule
Layout Extractor & Corner Stitching Data Structure
Symmetry Detection
Target Technology Design Rule
Sweeping Line Constraints Graph Generation
Symmetry Constraints
New Transistors Sizes
Transistor Resizing (Update Weight)
Generating, Solving and Translating Symmetry Core Problem
Bellman Ford & Wirelength inimization
Repeat from Sweeping Line step for the other layout direction
Target Layout

Figure 2: Proposed analog layout resizing flow.

2.2 Technology Migration Constraints

In order to facilitate the layout technology process migration and device resizing, a constraint graph that consists of nodes (representing the rectangles edges) and directed-weighted arcs (representing the constraints between edges) has to be created. One way of obtaining the graph is by using the sweeping line method [2]. First, the design rules for the target technology have to be acquired. Here, we categorize the design rules into three groups – minimum width, spacing, and extension. The sweeping line will start from the most left edge of the layout. While the line is traversing to the right, all required constraints from the current edge to the visible edges on its left will be added. The sweeping line algorithm also reduces redundant constraints, thus speeding up the solution solving. The example of constraints generated is shown in Figure 3.

As the sweeping line algorithm preserves the layout structure, our resizing tool requires two conditions: the target technology that covers all layers employed in the original layout, and the design that can be retargeted to the new process. Therefore, we shall call such process a *modestly* new process.

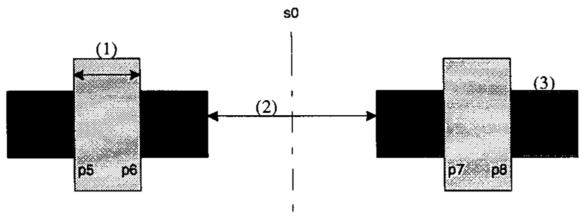


Figure 3: A layout of two transistors of only active layer (stripes) and poly layer (gray). Shown in numbers are examples of design rules: (1) minimum-width (2) spacing and (3) extension. The letters denote edges of active (a) rectangles and poly (p) rectangles, where (s) is a symmetry axis. Here are the constraints generated from this figure in horizontal direction. (Example design rules are taken from TSMC 0.25um)

$a2\text{-}a1 \geq 3$	$a4\text{-}a3 \geq 3$	$p6\text{-}p5 \geq 2$	$p8\text{-}p7 \geq 3$
$p5\text{-}a1 \geq 3$	$a2\text{-}p6 \geq 3$	$p7\text{-}a3 \geq 3$	$a4\text{-}p8 \geq 3$
$a3\text{-}a2 \geq 3$	$s0\text{-}p6 = p7\text{-}s0$	$p6\text{-}p5 = p8\text{-}p7$	

2.3 Transistor Resizing

Typically, it is necessary to resize transistors to accomplish the same or better performance when a design is targeted on a new process. The target transistor sizes can be found either by manual simulations or, preferably, by some automatic transistor sizing tools.

Transistor resizing is accomplished by adding to the constraint graph the fixed-width constraints for each transistor to reflect new widths (added to active rectangles) and lengths (added to poly rectangles). This needs to be done on both horizontal and vertical direction. For symmetric transistor pairs, all the pairs have to be resized with exactly the same dimensions.

While performing transistor resizing, there is one difficulty regarding the number of active to metal-one contacts. Since the transistors sizes can be either tightened or widened, when decreasing transistors width, the reduced active area may not be able to fit all existing contacts. Thus a contact removal scheme has to be executed. After the addition of size-constraints, all active to metal-one contacts that reside by the transistors are removed. We, then, need to add two constraints between the metal-one rectangle edges and active rectangle edges, as shown in Figure 4, in order to preserve the connectivity between the two areas. After the constraint graph problem is solved, rows of contacts will be placed back in the right location.

2.4 Constrained Layout Generation

The new layout can be achieved by solving the constrained linear programming problem, which can be mathematically described as

$\min\ (x_{r,o} - x_{l,o})$	*subject to layout constraint*
$(a)\ x_i - x_j \geq w$	*min-width, spacing, extension*
$(b)\ x_i - x_j = w$	*fixed-width*
$(c)\ x_i - x_j = 0$	*connectivity*
$(d)\ x_i - sym = sym - x_j$	*symmetry*

where variables $x_{r,0}$, $x_{l,0}$, and x_i or x_j are the most-right edge of the layout, the most-left edge of the layout, and any rectangles edges respectively.

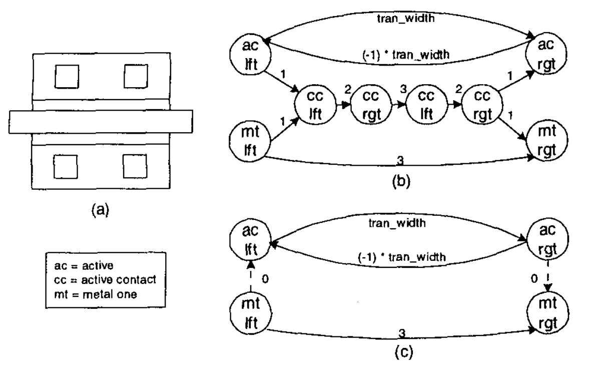


Figure 4: Contact removal in transistor resizing. (a) Transistor layout. (b) Original constraint graphs for only lower drain/source side with active contacts. (c) After removing contacts, two constraints are added for connectivity (metal->active and active->metal). Note: Constraints weights are taken from TSMC 0.25um process and in unit of lambda.

If we ignore the symmetry constraints, the above linear programming problem can be converted to the shortest path problem of the constraint graph represented by nodes as the layout rectangle variables and directed-weighted arcs as the design rule constraints.

From Section 2.2, constraint (a) $x_1 - x_2 \geq w$ can be expressed with a directed arc of weight w from node x_1 to x_2. Constraint (b) and (c) can be divided into two constraints of $x_1 - x_2 \geq w$ and $x_2 - x_1 \geq -w$, which also can be specified in the graph. Without the symmetry constraints, the minimum distance from one side of the layout to the other can be solved quickly with a graph-based shortest path algorithm (i.e. Bellman-Ford [6]).

However, in the presence of symmetry constraints, the linear programming is still necessary. Okuda *et al.* [4] has established an algorithm to solve this problem more efficiently by using advantages from both fast graph-based and linear programming technique. Instead of employing linear programming on a large problem, a smaller equivalent problem consisting of only the layout boundary variables and variables associated with symmetry axes are generated, and then solved with linear programming. The solution will give us the exact location of all variables in the equivalent problem. After that, we can replace each symmetry constraints with two fixed-width constraints and solve the compaction problem with the graph-based shortest path algorithm. Examples are

$$x_1 - sym = sym - x_2 \quad \Rightarrow \quad x_1 - sym = b \text{ and } sym - x_2 = b$$
$$x_3 - x_4 = x_5 - x_6 \quad \Rightarrow \quad x_3 - x_4 = c \text{ and } x_5 - x_6 = c$$

One weakness of the basic shortest path algorithm is that it tries to find the minimum distance from every variable to the starting (most-left) variable, which creates excessive leftward extension in some rectangles. It consequently results in bad performances due to surplus parasitic resistance and capacitance. Therefore, after solving the problem, minimization of individual rectangles as described in [2] or [5] has to be performed to secure good performance.

3. Examples

This section presents the results of applying our resizing tool on a single-ended folded-cascode and a two-stage operational amplifier. Both circuits are initially designed using the TSMC 0.25um CMOS process, and laid-out manually using Cadence's Virtuoso with multi-finger transistor structures. The target technology is the TSMC 0.18um CMOS process.

The CIF format files and the Cadence format technology file are imported from Virtuoso to our program. Once resizing is finished, the target CIF layout is exported back to Virtuoso, and a design-rule checking is performed. Finally, the netlists from both layouts are simulated by Hspice to compare their functionalities and performances.

3.1 Folded Cascode Operational Amplifier

Figure 5 shows the schematic of a folded cascode operational amplifier with 14 transistors (43 transistors if each finger is counted as a separate transistor). The original layout in the TSMC 0.25um process is shown in Figure 6, where the three symmetry axes A, B and C represented are {M1}:{M2}, {M3}:{M13} and {M4,M6,M8,M10}:{M5,M7,M9,M11}.

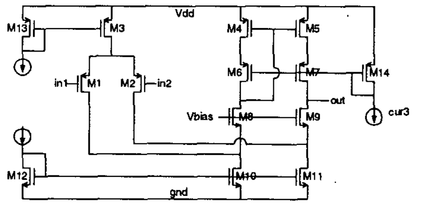


Figure 5: Schematic of a single-output folded-cascode opamp.

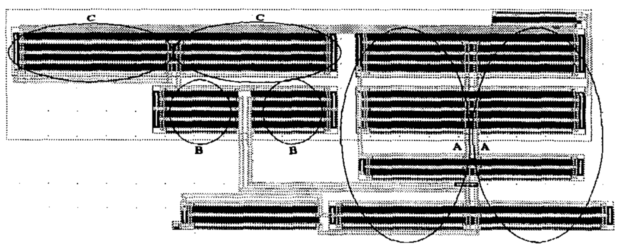


Figure 6: Original layout of the folded cascode operational amplifier in TSMC 0.25um. A, B and C are transistors symmetry blocks.

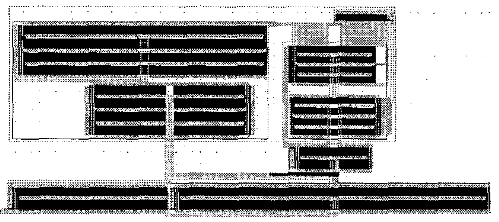


Figure 7: Resized layout of folded cascode opamp in TSMC 0.18um.

Figure 7 shows the resized layout in the TSMC 0.18um process. The transistor sizes are selected such that general operational amplifier specifications are met. The statistics on the performances and silicon area of the resized layout are summarized in Table 1, where "resize" and "no-resize" represent, respectively, the results with resized (the modified width and length from design engineers) transistors and no-resized (the same width and length as in the original layout) transistors.

Table 1: Performances comparison of folded-cascode opamp.

	0.25um	*0.18um no-resize*	*0.18um resize*
Vdd	*2.5 V*	*1.8 V*	*1.8 V*
Load Cap.	*1.0 pF*	*0.7 pF*	*0.7 pF*
Gain	*60.9 dB*	*61.9 dB*	*60.6 dB*
Bandwidth	*51.7 MHz*	*71.7 MHz*	*63.5 MHz*
Phase Margin	*63 deg*	*42 deg*	*71 deg*
Gain Margin	*12.5 dB*	*12.4 dB*	*10.5 dB*
Power	*1.48 mW*	*1.07 mW*	*0.88mW*
Area	*4,813 um^2*	*3,000 um^2*	*2,083 um^2*

3.2 Two-Stage Operational Amplifier

The schematic of a two-stage operational amplifier, shown in Figure 8, comprises of 1 MOS capacitor and 8 transistors (48 as each finger counted). There is one symmetry axis between two pairs of transistors {M1,M4}:{M2:M5}. The original layout (on TSMC 0.25um) is illustrated in Figure 9. The resizing is performed on width and length of all transistors, including the MOS capacitor. The target layout (in TSMC 0.18um) is depicted in Figure 10. The statistics on the performances and silicon area of the resized layout are summarized in Table 2.

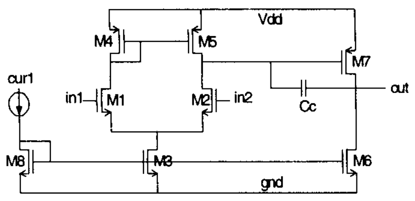


Figure 8: Schematic of a two-stage opamp.

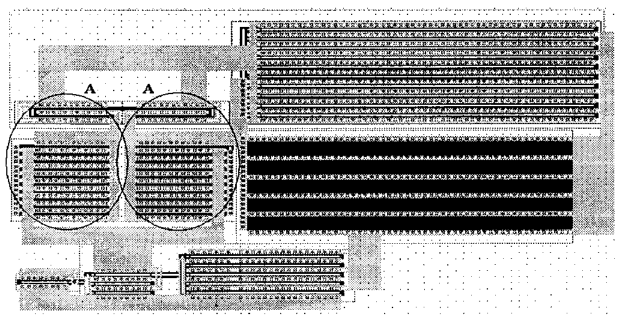


Figure 9: Original layout of the two-stage opamp in TSMC 0.25um.

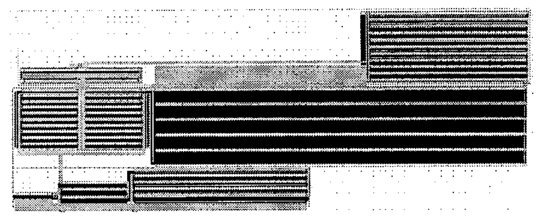


Figure 10: Resized layout of the two-stage opamp in TSMC 0.18um.

Table 2: Performances comparison of two-stage opamp.

	0.25um	*0.18um no-resize*	*0.18um resize*
Vdd	*2.5 V*	*1.8 V*	*1.8 V*
Load Cap.	*1.0pF*	*0.7pF*	*0.7pF*
Gain	*57.7 dB*	*39.6 dB*	*64.4 dB*
Bandwidth	*135 MHz*	*181 MHz*	*104 MHz*
Phase Margin	*50 deg*	*56 deg*	*56 deg*
Gain Margin	*9.6 dB*	*12.5 dB*	*9.2 dB*
Power	*4.82 mW*	*3.56 mW*	*3.46 mW*
Area	*3,648 um^2*	*2,664 um^2*	*2,745 um^2*

The runtime for the folded cascade opamp is 39.2 seconds and the two stage opamp is 37.6 seconds on a 440MHz SUN ultrasparc10 workstation.

4. Conclusion

An automatic tool for re-targeting existing analog layouts to new technology processes and new device sizes is presented. Layout recycling with symmetry detection and layout conservation scheme is applied in order to preserve the properties of analog circuit layout. Additionally, the tool has the ability to consider new transistor sizes to achieve better performances as part of the re-targeting process.

5. References

[1] J. K. Ousterhout, "Corner stitching: A Data-Structuring Technique for VLSI Layout Tools", *IEEE Transactions on Computer Aided-Design of Integrated Circuits and Systems*, pp.87-100, January 1984.

[2] S. L. Lin and J. Allen, "Minplex – A Compactor that Minimizes the Bounding Rectangle and Individual Rectangles in a Layout", *Proceedings of Design Automation Conference*, pp.123-130, 1986.

[3] Y. Bourai and C. J. R. Shi, "Symmetry Detection for Automatic Analog Layout Recycling", *Proceedings of Asian and South Pacific Design Automation Conference*, pp.5-8, 1999.

[4] R. Okuda, T. Sato, H. Onedera and K. Tamaru, "An Efficient Algorithm for Layout Compaction Problem with Symmetry Constraints", *Proceedings of International Conf. on Computer Aided Design*, pp.148-151, Nov. 1989.

[5] G. Lakhani and R. Varadarajan, "A Wire-Length Minimization Algorithm for Circuit Layout Compaction", *Proceedings of International Symposium on Circuits and Systems*, pp.276-279, 1987.

[6] T. H. Cormen, C. E. Leiserson and R. L, Rivest, *Introduction to Algorithms*, MIT Press, 1990.

Acknowledgement: The authors would like to thank Kiyong Choi and Jinho Park, SOC lab, Dept. of Electrical Engineering, University of Washington, for valuable discussions on circuit examples.

Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE @ DIRECT®

INTEGRATION, the VLSI journal 36 (2003) 237–262

INTEGRATION the VLSI journal

www.elsevier.com/locate/vlsi

IPRAIL—intellectual property reuse-based analog IC layout automation[☆]

Nuttorn Jangkrajarng, Sambuddha Bhattacharya, Roy Hartono, C.-J. Richard Shi*

Mixed-Signal CAD Research Laboratory, Department of Electrical Engineering, University of Washington, Seattle, WA 98195, USA

Received 16 March 2003; received in revised form 13 August 2003; accepted 25 August 2003

Abstract

This paper presents a computer-aided design tool, IPRAIL, which automatically retargets existing analog layouts for technology migration and new design specifications. The reuse-based methodology adopted in IPRAIL utilizes expert designer knowledge embedded in analog layouts. IPRAIL automatically extracts analog layout intellectual properties as templates, incorporates new technology design rules and device sizes, and generates fully functional layouts. This is illustrated by retargeting two practical operational amplifier layouts from the TSMC 0.25 μm CMOS process to the TSMC 0.18 μm CMOS process. While manual re-design is known to take days to weeks, IPRAIL only takes minutes and achieves comparable circuit performances.
© 2003 Elsevier B.V. All rights reserved.

Keywords: Analog integrated circuit design; Analog layout automation; Layout symmetry; Analog synthesis and optimization

1. Introduction

With the increasing demand for smaller, cheaper, more portable electronics in wireless communication and consumer electronics, semiconductor industry is moving towards mixed-signal systems-on-chips. Multiple functionalities such as digital, analog, and even radio frequency

[☆] This research has been supported in part by US Defense Advanced Projects Agency NeoCAD Program under Grant No. 66001-01-1-8920, in part by National Science Foundation (NSF) ITR Program under Grant No. 9985507, and in part by a grant from Conexant Systems.

*Corresponding author. Tel.: +1-206-2215291; fax: +1-206-5433842.

E-mail addresses: njankra@ee.washington.edu (N. Jangkrajarng), cjshi@ee.washington.edu (C.-J.R. Shi), .

0167-9260/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/j.vlsi.2003.08.004

(RF), which used to reside on many different chips, are being converged into one or a few chips. Driven also by the need to be more powerful, semiconductor manufacturers continue to innovate technologies towards smaller and smaller *transistor feature sizes* (for example from 0.25 to 0.18 to 0.13 μm). As a result, there is an increasing need in re-designing functioning mixed-signal layouts for new technology processes.

Due to the differences in technology process properties, migrating an available layout to a new process requires an overall re-design and re-creation of a new layout. In order to speed up the design process, design engineers can utilize available *computer-aided-design* (CAD) tools to mitigate part of the jobs. For digital layout, designers can employ the scalable cell libraries and the readily available automatic place and route tools to the existing high-level VHDL or Verilog design, in order to generate a target layout. On the other hand, analog designers do not have a comparable ability, which means they have to go through a full time-consuming cycle of redesigning, testing and drawing layouts. Therefore, an automated re-layout tool for analog circuits will significantly accelerate the mixed-signal circuit technology migration.

In this paper, we present a CAD tool, called Intellectual Property Reuse-based Analog IC Layout (IPRAIL), which automatically retargets an existing analog layout to *modestly new* processes. The methodology we propose here is based on the '*recycling*' scheme. IPRAIL uses an already fined-tuned input layout to automatically create a *structural template*, and then imposes new device sizes and new technology process design rules on the template. From this, IPRAIL generates an output layout that satisfies all the design rules while preserving all the analog layout intellectual properties such as device/wiring alignment, matching and symmetry. IPRAIL also preserves all unique aspects of the input layout intended by designers. Some preliminary results of this work were presented in [1].

This paper is structured as follows. Section 2 exhibits issues and previous work in analog layout automation. Section 3 illustrates the proposed IPRAIL methodology. Section 4 explains the process of layout template extraction from an existing layout. Section 5 describes automatic layout generation from an extracted layout template. Section 6 presents the experimental results of IPRAIL. Section 7 points out the limitations of IPRAIL and suggests some future work. Section 8 concludes the paper.

2. Background

2.1. Issues in analog layout automation

The strong impact of layout geometry on circuit performance makes analog layout design a very complicated task. Device matching and symmetry, parasitics, current density in interconnects, thermal, and substrate effects are of utmost importance in high performance analog circuits [2,3]. Overcoming these challenges is essential to the success of analog layout automation. The important layout issues are briefly discussed below.

2.1.1. Matching and symmetry

Transistors designed to behave identically may exhibit finite mismatch due to asymmetry in their layout structures or locations. The asymmetry in transistor layouts is mainly due to the differences in channel orientations and surrounding environment of the two transistors [2]. Two transistors are deemed symmetric if their layouts are geometric mirror images of each other. For large or stacked transistors, layouts drawn by maintaining simple geometric mirroring may not establish acceptable matching due to spatial variations in process parameters like oxide thickness, mobility etc. In such cases, *common-centroid* configurations are often employed to cancel out the mismatches introduced due to process gradients.

Transistor mismatch can drastically affect analog circuit performance leading to DC offsets, finite even-order distortion and lower common-mode rejection [4]. Ensuring layout symmetry, between transistors identical by design, demands significant designer effort.

2.1.2. Floorplanning and device locations

Circuit performance may be significantly affected by the exact positioning of certain devices and blocks with respect to the rest of the layout. Thermal and substrate effects, together with variations in lithographic processes, demand careful floorplanning of sensitive devices and blocks.

2.1.3. Wiring considerations

Wiring parasitics, cross-talk and coupling can severely affect sensitive signal nets. Conservative layout styles maintaining wire-spacing and wire-shielding are often employed by expert layout designers for proper functioning of analog circuits. Wire-sizing for maintaining prescribed current density and preventing electromigration are of utmost importance in analog layout design.

2.2. Previous work on analog layout automation

2.2.1. Procedural module generation

The earliest approaches to analog layout automation belong to the class of procedural module generation. These schemes usually employ a designer-constructed geometric template that specifies all device-to-device and device-to-wiring spatial relationships. The template generation is accomplished either through a procedural language [5,6] or a graphical user interface [7]. The actual layout generation involves filling up the template with correct device and wire sizes. The drawbacks of these methods are in their limited flexibility and high cost of template generation. Recently, [8] proposed a template-based module generation method that attempts to palliate the flexibility issue by extensive hierarchical templating. Unfortunately, the effort required for construction of such templates exceeds the actual manual creation of custom layouts by almost an order of magnitude.

2.2.2. Macro-cell placement and routing

These approaches [9–12] inherit the basic design methodologies of the digital CAD world and adapt them to analog layout automation by incorporating performance-based optimization. Devices are treated as flexible blocks and may be reshaped and reoriented using slicing-tree floorplans prior to their automatic placement and routing. The layout generation proceeds without any intervention from the layout designer. While these methods are very general in principle, they require extensive computation and, more importantly, fail to incorporate the expertise of analog layout designers into the flow. Unfortunately, the lack of designer input into the automation often results in performance degradation and forestalls the acceptance of these methods in the design community.

3. Proposed automatic analog layout retargeting methodology

The incorporation of designer experience is a major factor to the acceptance of analog layout automation. Therefore, IPRAIL draws its inspiration from the procedural module generation methods. In contrast to [8], IPRAIL facilitates layout reuse by automatically generating the templates from an existing analog layout drawn by the experienced designer. Our objective of reusing an existing layout for new process and/or specifications is accomplished by direct extraction of the knowledge embedded in the layout.

As illustrated in Fig. 1, the *original layout* and its technology information are first fed into IPRAIL. The device sizes under the new specifications are obtained either by manual simulations or from an analog circuit synthesis tool. First, IPRAIL converts the original layout into a *resizable symbolic template*. It then generates the new layout, henceforth called *target layout*, by imposing the target process design rules and new device sizes as constraints on the symbolic template. The entire process of automatic creation of symbolic template and generation of target layout takes a few minutes of CPU time.

The IPRAIL tool-suite consists of two main components: the *layout template extractor* and the *layout generator*. Fig. 2 illustrates the internal structure and interface of IPRAIL in greater detail. First, the template extractor scans in the original layout in *Caltech Intermediate Format* (CIF) [13]. In CIF, a layout is expressed in ASCII format, which describes two-dimensional shapes on each layer based on their coordinates. The technology process design rules associated with the input layout are obtained from the Cadence environment [14].

The layout template extractor identifies the active and passive devices, detects device matching and symmetry, and extracts device connectivity and net-topology from the original layout. Based on the extracted information and the technology process design rules, it transforms the layout into a constraint-based resizable symbolic template representation. The "*symbolic layout template*" is virtually an abstract representation of the extracted layout properties, namely devices and connectivity, technology process design rules, and analog layout integrities.

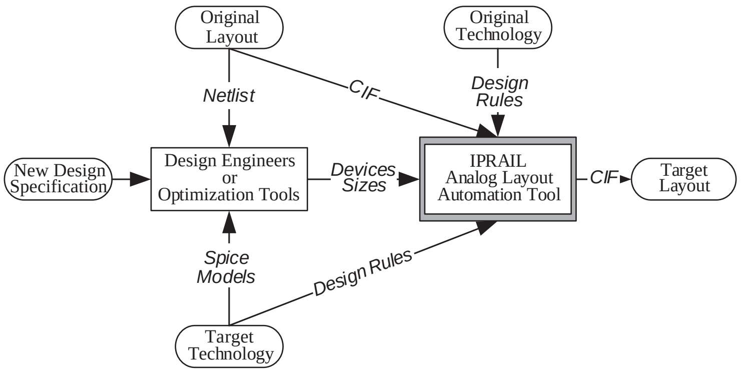


Fig. 1. Analog layout retargeting methodology using IPRAIL.

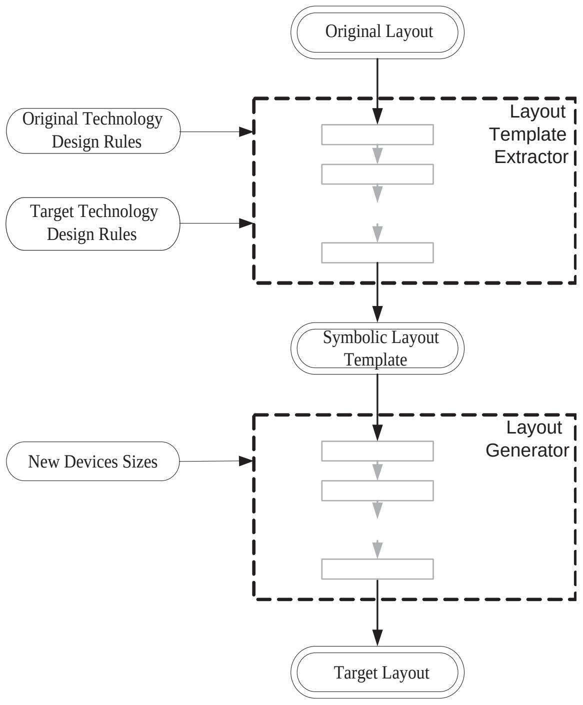

Original Layout
Original Technology Design Rules
Target Technology Design Rules
Layout Template Extractor
Symbolic Layout Template
New Devices Sizes
Layout Generator
Target Layout

Fig. 2. Overview of IPRAIL structure and flow.

The key tasks of the layout generator are to enforce new device sizes and to resolve the symbolic layout representation for rectangle locations. This is accomplished by a combination of linear programming and graph-based methods. The target layout is generated in CIF.

The direct incorporation of the embedded knowledge in the original layout as a template ensures retention of the layout integrity. The target layout generated by IPRAIL is checked for design-rule compliance.

4. Layout template extractor

The detailed flow of template extraction is shown in Fig. 3. It involves representation of the layout in corner-stitching data structure, extraction of transistors, nets and passive devices, generation of layout constraints, and detection of device symmetry. Each sub-task performed by the template extractor is described below.

4.1. Layout representation by corner-stitching data structure

IPRAIL adopts the *corner-stitching* data structure [15] for storing a layout. Our preference for corner-stitching over other potential data structures, for example bins and linked-lists, is dictated by its efficiency in fast localized searches, as described in [16]. An example of the corner-stitching data structure is shown in Fig. 4. In this scheme, the entire plane of each mask layer is represented

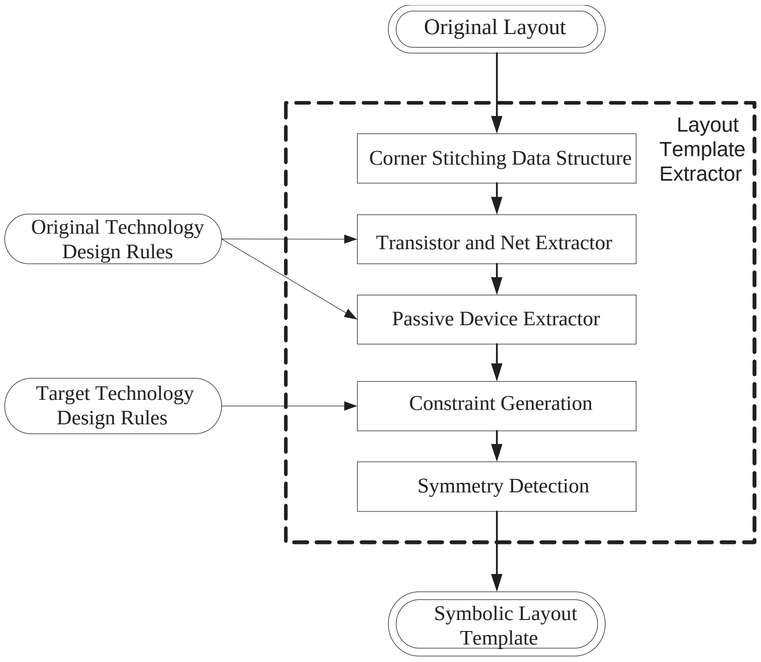

Original Layout
Layout Template Extractor
Corner Stitching Data Structure
Original Technology Design Rules
Transistor and Net Extractor
Passive Device Extractor
Target Technology Design Rules
Constraint Generation
Symmetry Detection
Symbolic Layout Template

Fig. 3. Internal flow of the layout template extractor in IPRAIL.

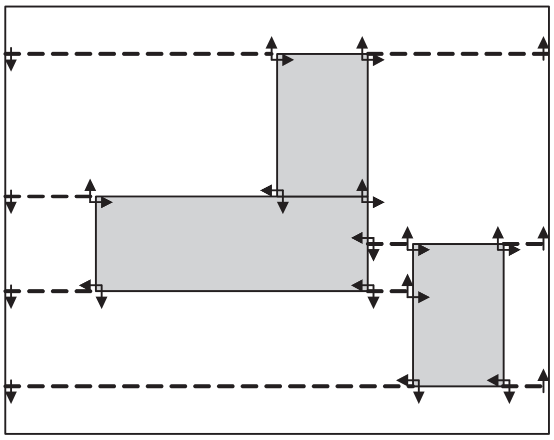


Fig. 4. An example of the representation of a layout (one mask layer) in the corner-stitching data structure.

explicitly in terms of solid (gray) and space (white) rectangles called *tiles*. Each tile in a layer plane is connected to other tiles in the same plane by four *stitches* on its lower-left and upper-right corners, and is organized such that *maximal horizontal strips* is achieved. Some of the basic corner-stitching based operations frequently used in IPRAIL are *area-enumeration* for finding all tiles in a given area, *point-finding* for locating a tile at a given position, and *neighbor-finding* for listing all tiles adjacent to a given tile.

4.2. Transistor and net extraction

A *metal–oxide-semiconductor field-effect-transistor* (*MOSFET*) in a layout is defined as an overlap between two tiles in *poly-silicon* and *diffusion* (*active*) mask layers. A transistor has three terminals, viz., the gate terminal in the poly-silicon layer and the source and drain terminals in the diffusion layer. A net is defined as an electrical connection between the terminals of transistors or external ports. IPRAIL currently does not support *bipolar-junction-transistor* (BJT).

Extraction of transistors and nets from the layout follows the algorithm proposed in the Magic VLSI layout system [17,18]. The usage of the corner-stitching data structure ensures fast extraction of transistors and nets in the circuit layout. Using area enumeration in the corner-stitching database, the extractor detects transistors by looking for overlaps between the poly-silicon and the diffusion layers. The transistor description stored in the database includes its orientation, size, location, and terminal information.

Once the transistors are extracted, a simple recursive algorithm detects nets in the layout using the terminals of the transistors as starting points. The fundamental operation involves marking all tiles that are electrically connected. The tile at the start-point is marked first and all neighbor tiles in the same mask layer are identified. The algorithm proceeds to one of the neighboring tiles and continues through a depth-first-search. If vias or contacts are encountered, the search moves to the next mask layer.

4.3. Extraction of passive devices

Resistors in analog layouts are typically designed in the poly-silicon mask layer as it exhibits high linearity, low capacitance to substrate and relatively small mismatches [3]. In some technologies, resistors are also constructed in the n-well or diffusion layers. The resistor topology supported in IPRAIL consists of single unit or multiple units laid out in parallel and connected in series, as shown in Fig. 5.

Resistors in the layout are detected by searching through the tiles of the nets in the circuit. A single tile or a series of connected tiles of a net are classified as a resistor when the resistive value exceeds a user-defined threshold. Once a resistor is detected, its parent net is split into two.

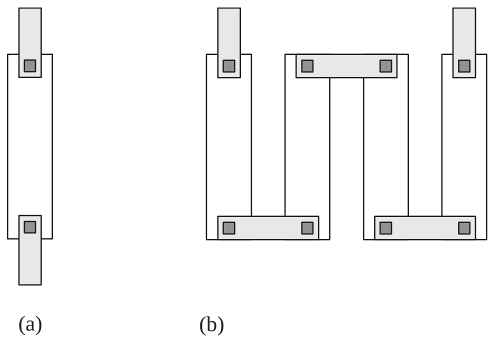


Fig. 5. Layouts of resistors. (a) A resistor constructed from a single tile. (b) A resistor constructed from multiple series-connected tiles.

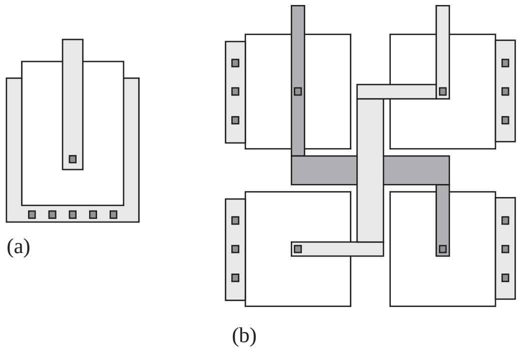


Fig. 6. Layouts of capacitors in a P–P or MIM structure. (a) A capacitor constructed with one pair of tiles. (b) Two capacitors laid out in common-centroid configuration.

Connectivity between a resistor and the nets is maintained through port tiles. The resistor data structure stores geometry information along with connectivity at its ports.

High-density linear capacitors are fabricated using a poly-silicon over another poly-silicon (P–P) layer in a "double poly" process [3]. In absence of such structures, capacitors are fabricated as sandwiches of two or more metal layers over silicon dioxide insulator (M–I–M). Examples of P–P and M–I–M capacitors are shown in Fig. 6. Alternately, MOS transistors can be used as non-linear capacitors (MOSCAP) by shorting their source and drain terminals.

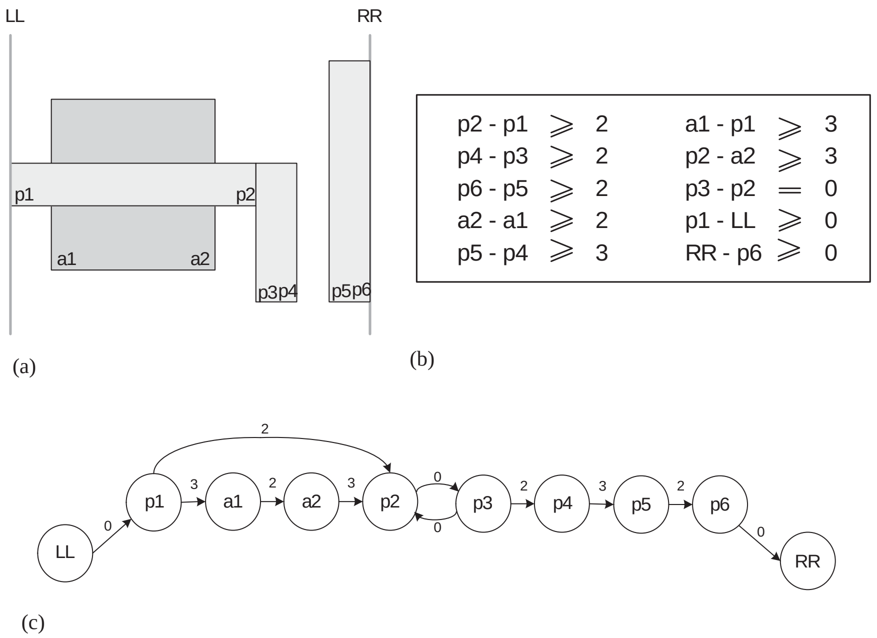
In IPRAIL, capacitors are defined as overlaps of two tiles in different layers that belong to different nets. Searching through the nets, the extractor detects capacitors when the capacitance due to the overlap exceeds a user-defined threshold. The MOSCAPs are detected during the transistor extraction. The geometry information and connectivity are stored in the capacitor data structure.

During the ensuing layout generation phase, if a passive is resized, other devices or wiring tiles may overlap with it. To prevent this, a *temporary dummy tile* is placed over the device location. The tile is defined in a new *dummy mask layer*, and is furnished with spacing constraints to every mask layer. This reserves exclusive space for the passive device.

4.4. Constraint generation for technology migration

The symbolic template is based on a set of *geometric constraints* between the tiles in the layout. The constraints arise due to the connectivity between tiles and the technology design rules. The connectivity-based constraint between a pair of tiles ensures that the tiles remain electrically connected after the layout generation process. The constraints enforced by the technology design rules belong to one of the following three categories: (1) *minimum width* of a tile, (2) *minimum spacing* between two electrically unconnected tiles on the same or different mask layers, and (3) *minimum extension* of two overlapping tiles on different mask layers.

The constraints for the symbolic template are established independently in the horizontal and vertical directions. Here, we describe the method for generating the constraints in the horizontal direction. The template constraints can be formulated in an equation form. Fig. 7(a) and (b) illustrate a layout and its corresponding *constraint-equations* respectively in TSMC 0.25 μm CMOS technology process, where LL is the left-most boundary and RR is the right-most boundary. Here, variables a_i and p_i are associated with the left and right *tile-edges* of each



p2 - p1 $\geqslant$ 2	a1 - p1 $\geqslant$ 3
p4 - p3 $\geqslant$ 2	p2 - a2 $\geqslant$ 3
p6 - p5 $\geqslant$ 2	p3 - p2 $=$ 0
a2 - a1 $\geqslant$ 2	p1 - LL $\geqslant$ 0
p5 - p4 $\geqslant$ 3	RR - p6 $\geqslant$ 0

Fig. 7. Constraint-based template formulation in horizontal direction. (a) An example layout. (b) A layout template in an equation form. (c) A layout template in a constraint graph form.

rectangle and are used in the constraint equations. For this example, the four different types of constraint-equations are:

$p_2 - p_1 \geqslant min\_poly\_width$	minimum width constraint
$p_5 - p_4 \geqslant min\_poly\_space$	minimum spacing constraint
$p_2 - a_2 \geqslant min\_diff\_extension$	minimum extension constraint
$p_3 - p_2 = 0$	connectivity constraint

Generating constraints between every pair of tile-edges in the layout leads to significant redundancy. For example, in Fig. 8, the minimum spacing constraint between the right tile-edge of rectangle *b* (represented by b_r) and the left tile-edge of rectangle *s* (represented by s_l) is not required. This is due to the existence of a minimum width constraint in a tile *d*, and minimum spacing constraints between tile-edges b_r and d_l and between tile-edges d_r and s_l. This is verified from the following constraint equations:

$$d_l - b_r \geqslant min\_space, \tag{1}$$

$$s_l - d_r \geqslant min\_space, \tag{2}$$

$$d_r - d_l \geqslant min\_width, \tag{3}$$

$$(1) + (2) + (3): \quad (d_l - b_r) + (s_l - d_r) + (d_r - d_l) \geqslant 2(min\_space) + min\_width$$

$$\text{or}: \quad s_l - b_r \geqslant 2(min\_space) + min\_width. \tag{4}$$

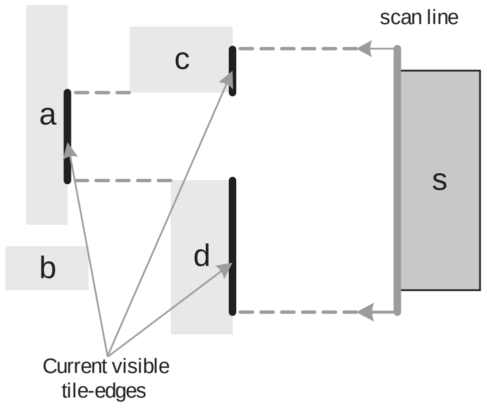


Fig. 8. Example of the scan-line method.

Eq. (4) is the linearly dependent on Eqs. (1–3) and therefore superfluous. To prevent such redundancy, the constraint equations are generated by employing a *scan-line* method [19].

The scan-line used for generating horizontal constraints is a vertical line that sweeps through the layout from left to right. It enumerates all tile-edges in a list sorted by their abscissas. For a tile-edge in the sorted list, constraints are generated from that tile-edge to all other tile-edges to its left that are "visible" (not blocked by other tiles in the same mask layer) from it. For example in Fig. 8, parts of the right tile-edges of the rectangles a, c and d (as marked) are visible from the left tile-edge of rectangle s, whereas the right tile-edge of rectangle b is not visible as it is blocked by rectangle d. Thus, while processing the left tile-edge s_l of rectangle s, separation constraints are generated only for the right tile-edges a_r, c_r and d_r of rectangles a, c and d respectively. After all constraints from rectangle s are generated, the scan-line will move to the next rectangle on the list. The process continues until all edges are handled. The algorithm has a worst-case complexity $O(N^2)$, where N is the number of tiles.

The constraint-equations of the symbolic template, as defined in Fig. 7(a) and (b), can also be represented as a *weighted directed constraint graph* $G = [V, E]$. From the constraint-equations, a graph can be constructed where the equation variables are represented as graph vertices (V) and the equation constants as weights of the graph edges (E). All connectivity and design rule constraints appear in one of the three following forms: *lower bound constraints*, *upper bound constraints* and *fixed weight constraints*. A lower bound constraint-equation of the form $x_i - x_j \geqslant w$ is represented in the graph as a directed edge from vertex x_j to vertex x_i of weight w. An upper bound constrained equation of the form $x_i - x_j \leqslant w$ is represented by a directed edge from vertex x_i to vertex x_j of weight $-w$. A fixed weight constrained equation of the form $x_i - x_j = w$ is first separated into two different equations $x_i - x_j \geqslant w$ and $x_j - x_i \geqslant -w$. These are represented in the graph by one directed edge of weight w from vertex x_j to vertex x_i and another directed edge of weight $-w$ from vertex x_i to vertex x_j respectively. Fig. 7(c) shows an example of the graph constructed from the equations.

Here, if the target layout design involves a migration to a new process technology, the constraint-equations have to be updated with the minimum width, spacing and extension design rules of the new technology.

4.5. Symmetry detection

Detection of matching and symmetry of transistors in the layout is of utmost importance for layout template extraction. Two transistors are considered symmetric if they are "*geometrically mirrored*". This involves equal dimension, similar horizontal or vertical location, uniform orientation and same type.

The symmetry detection in IPRAIL is based on the algorithm described in [20]. Here, we describe the detection for horizontally aligned transistors. Consider the two transistors *M4* and *M5* in Fig. 9. First, all tiles belonging to transistor *M4* are collected in a list in an increasing order of the left tile-edge positions. The tiles that constitute transistor *M5* are collected in another list in a decreasing order of the right tile-edge positions. If the contents of the two lists are matched in terms of locations, sizes, and layers, then the two transistors are deemed symmetric and a symmetry axis is detected. The symmetry axis for vertically aligned transistors can be obtained similarly.

The example of common-centroid topology is shown in Fig. 10 where transistors *M1* and *M2* are matched in a schematic diagram. In a layout, the transistors *M1* and *M2* are each divided into two halves. Each half is laid out diagonally from the other half to cancel out mismatches. Here, two symmetry axes are required to maintain the common-centroid layout. Transistor pairs (m_1:m_4) and (m_2:m_3) are symmetric vertically by sym_1 axis. Transistor pairs (m_1:m_2) and (m_4:m_3) are symmetric horizontally by sym_2 axis.

We extend our algorithm to the detection of symmetry between two groups of transistors. Two groups are symmetric if there exists inter-group pair-wise matching between horizontally or vertically aligned transistors. In some circuits, designer intervention might be necessary for detecting matching between only a few groups of transistors. In this interactive mode, symmetry between two groups of transistors can be detected by drawing bounding-boxes thereby selecting each group. In IPRAIL, similar facility for detecting symmetry in the batch mode is also provided through input text files.

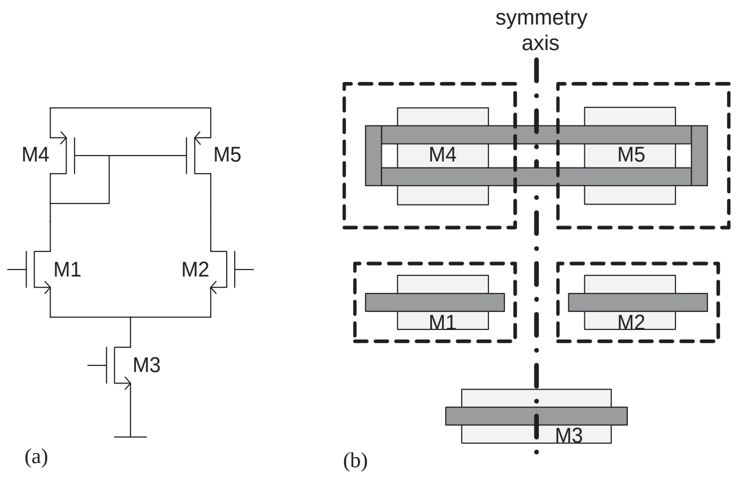


Fig. 9. Example of symmetric transistors. (a) The symmetric pairs are (M1:M2) and (M4:M5). (b) The symmetry axis of transistor pairs M1:M2 and M4:M5.

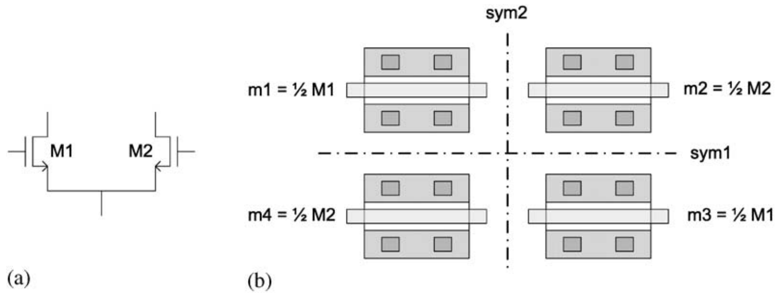


Fig. 10. A common-centroid layout and symmetry axes. (a) Schematic diagram. (b) Layout.

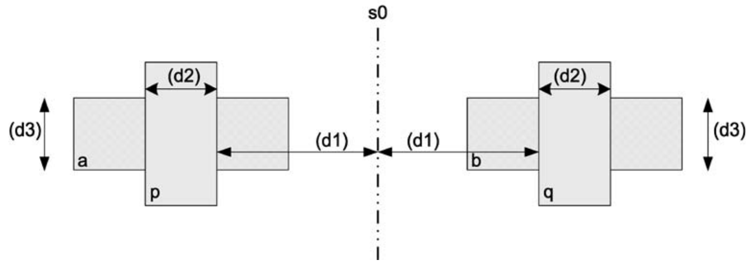


Fig. 11. A simplified layout of two symmetric transistors showing only diffusion (stripes) and poly-silicon (gray) layers. Distance *d1*, *d2* and *d3* are kept equally to maintain symmetry.

For each symmetric transistor pair, three types of constraint-equations are generated as illustrated in Fig. 11. Edges of rectangle a are defined as a_l (left), a_r (right), a_b (bottom) and a_t (top). Edges of rectangles b, p and q are defined similarly. The symmetry axis is denoted by s_0. The constraints generated due to the symmetric transistors are

$$2s_0 - p_r - q_l = 0, \tag{5}$$

$$p_r + q_l - p_l - q_r = 0, \tag{6}$$

$$a_t - b_t = 0 \quad \text{and} \quad a_b - b_b = 0. \tag{7}$$

Eq. (5) preserves the distance *d1* between left and right transistors to the symmetry axis. Eq. (6) matches the lengths *d2* of the two transistors. As these three-variable or four-variable constrained equations maintain the distance between two variable pairs, we shall call them *equi-distance constraints*. Eq. (7) maintains the vertical location of both transistors and matches their widths *d3*.

As for representation in the constraint graph, the fixed-weight equations "$a_t - b_t = 0$" and "$a_b - b_b = 0$" can be included in the graph directly (as described in Section 4.4). However, there are no straightforward ways to represent the equi-distance equations as graph edges. So these constraints have to be set aside in the equation form.

5. Layout generator

The layout generator in IPRAIL creates the target layout from the symbolic template extracted from the original layout. Fig. 12 shows the various steps in the layout generation process. First, the layout generator updates the template with the transistor and passive sizes obtained from a circuit synthesis tool. As the updated template consists of the symmetry, connectivity and design rule constraints, the problem of generation of the target layout from the symbolic template reduces to a *symbolic compaction problem*. This is solved by a combination of linear programming and graph-based shortest-path algorithm.

5.1. Transistor sizing

For a new technology process, the new transistor widths and lengths can be added to the constraint graph as fixed weight constraints on the gate diffusion tile and gate poly-silicon tile, respectively. Due to changes in transistor sizes, the diffusion-metal-one *contacts* at the drain and source terminals require careful handling. If the size of a transistor is smaller in the target layout than in the original, the drain or source area may not be able to accommodate the original number of contacts. To overcome this, the contacts are first removed from the layout, and extra constraints are added between the diffusion and metal-one tile-edges to preserve their overlap. Fig. 13 illustrates the constraints of the transistors laid-out horizontally. After the contacts are removed, two constraints are added from the left-edge of metal-one to the left-edge of diffusion and from the right-edge of diffusion to the right-edge of metal-one for

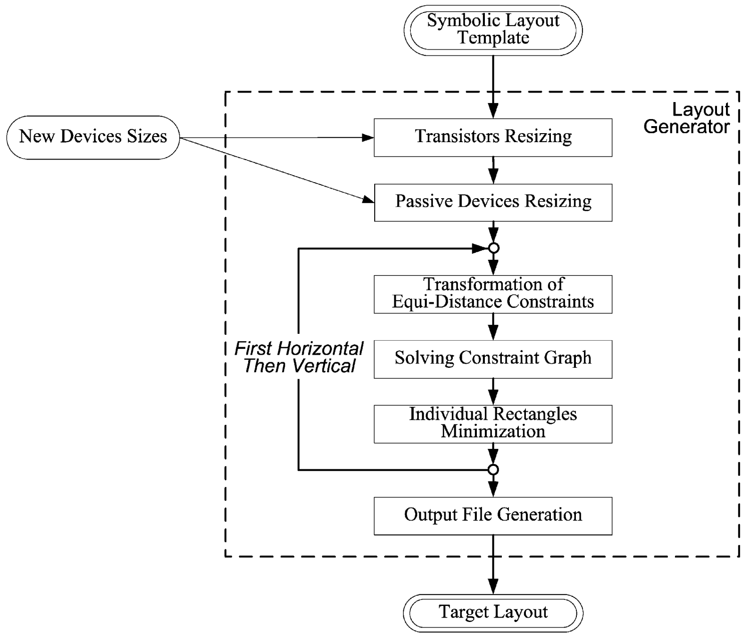


Fig. 12. Internal flow of the layout generator in IPRAIL.

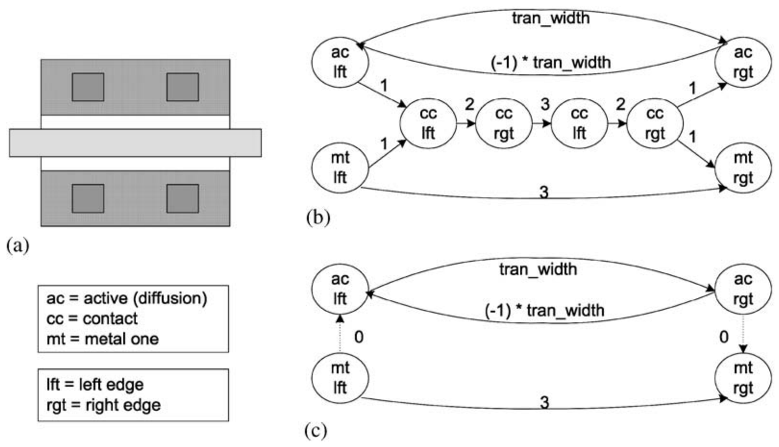


Fig. 13. Contact removal in transistor resizing along the width of transistors. (a) A transistor layout. (b) Original constraint graphs for only lower diffusion side with contacts. (c) Constraint graphs after removing contacts.

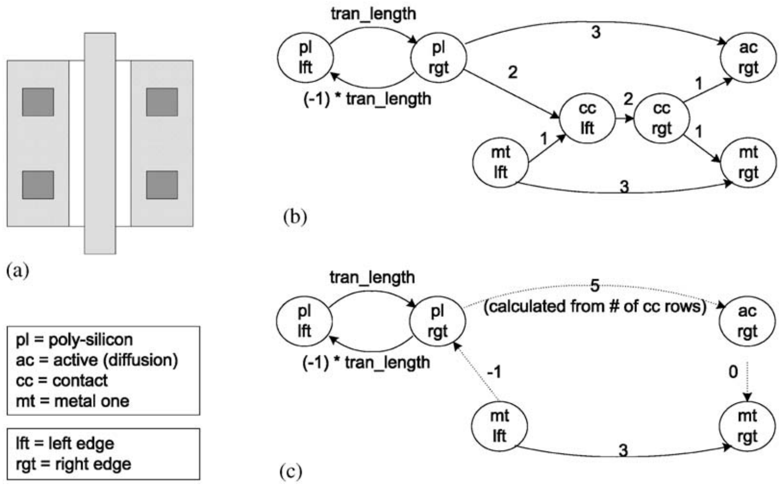


Fig. 14. Contact removal in transistor resizing along the length of transistors. (a) A transistor layout. (b) Original constraint graphs for only right diffusion side with contacts. (c) Constraint graph after removing contacts.

connectivity. Fig. 14 illustrates the constraints of the transistors laid-out vertically. After the contacts are removed, two constraints from the left-edge of metal-one to the right-edge of the poly-silicon and from the right-edge of diffusion to the right-edge of metal-one are added for connectivity. In addition, one constraint from the right-edge of poly-silicon to the right-edge of diffusion is updated, based on the number of contacts. In both cases, during the generation of the target layout, rows of contacts are added to connect such diffusion and metal-one tiles.

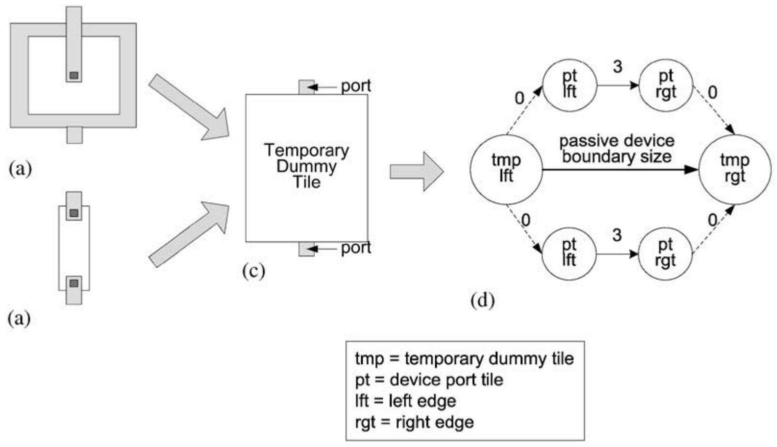


Fig. 15. Passive devices resizing. (a) A P–P or M–I–M capacitor layout. (b) A resistor layout. (c) Replacing the passive device with a temporary dummy tile. (d) Constraint graphs. The minimum width of port layer in this example is arbitrarily chosen as 3 units.

5.2. *Updating passive device sizes*

For a new technology process, the passive device geometries and dimensions can be changed from its original layout. Updating the passive device sizes is done by adding constraints between the temporary dummy tile left and right tile-edge variables. Furthermore, extra constraints have to be incorporated to the constraint graphs between the temporary dummy tile variables and the port tile variables to keep the passive device aligned. This maintains connection between the passive devices and the circuit nets after the compaction process. An example of temporary dummy tile variables, port tile variables and their constraints is described in Fig. 15.

There are two schemes for replacing a passive device. If the new device has the same structure as the original one but is different in size, the fixed-weight constraints are added between corresponding tile variables in the graph. If the new device has different structure, we totally remove the passive device tile variables and their constraints. The dummy tile is resized according to the new device boundary geometries. And the port tiles are restricted so that they are attached to the dummy tile. The ports are positioned in the middle, unless they are located on the same side. In this case, when generating the output layout, the new device layout has to be created based on device geometries and ports location and added in place of the dummy tile.

5.3. *Compaction by combined linear programming and graph-based algorithms*

The assembled template constraint problem can be solved by applying the layout compaction algorithm. The problem in equation form can be solved directly using *linear programming* (LP) [21]. Nevertheless, it is too computationally expensive for VLSI layouts, due to the problem size. Traditionally, graph-based compaction methods [19,22,23] are preferred for their superior computational speed. The presence of the equi-distance constraints resulting from symmetry detection, however, hinders the successful application of the conventional graph-based methods.

Recall the example layout of Fig. 11. The two equi-distance equations for the layout are

$$2s_0 - p_r - q_l = 0, \tag{5}$$

$$p_r + q_l - p_l - q_r = 0. \tag{6}$$

Eq. (5) preserves the distance from the left and the right transistors to the symmetry axis. The matching of the two transistor lengths is ensured by Eq. (6). As the equi-distance equations contain three or four variables, they cannot be directly incorporated as weighted edges into the constraint graph. Therefore, all equi-distance constraints have to be converted into combinations of two-variable constrained equations, which can be added to the graph. For this conversion, the optimum distances for the two-variable equations can be obtained by a combination of linear programming and graph-based algorithms introduced in [24].

First, a *core-graph* $G' = [V', E']$, which is a reduced-sized equivalent graph consisting of variables $\{v_{\mathrm{LL}}, v_{\mathrm{RR}}\}$ U $\{v_i \,|\, v_i$ appears in equi-distance constraints$\}$, is generated based on the original constraint-graph G. The variables v_{LL} and v_{RR} correspond to the left and the right boundaries respectively. In order to keep the core-graph equivalent to the original graph, the constraints between all variables have to be derived from the original constraints. This is accomplished by searching the longest directed path between v_i and v_j in G, for $\forall v_i, v_j \in V'$ and $i \neq j$. If the path exists and no $v_k \in V'$ $(k \neq i,j)$ is on the path, a directed edge from v_j to v_i with weight of the longest path length will be added to G'. At the end of this process, the core-graph G' possesses the same properties as the original graph G. Next, the core-graph is converted into an LP-compatible equations form. Incorporating these LP-equations with the equi-distance constraints, the problem can be solved together by linear programming.

The solution from the LP problem above yields optimal values for the variables related to the equi-distance constraints. This, in effect, converts the equations into a form that can be added to the original constraint graph G. We obtain the following form for Eq. (5), where b is a constant calculated from the optimal linear programming solution.

$$s_0 - p_6 = p_7 - s_0 = b. \tag{8}$$

Eq. (8) can then be further converted into the following form:

$$s_0 - p_6 \geqslant b, \quad p_6 - s_0 \geqslant -b, \quad p_7 - s_0 \geqslant b, \quad s_0 - p_7 \geqslant -b. \tag{9}$$

Directed edges of weight b are then added from p_6 to s_0 and from s_0 to p_7 in the original constraint graph G. Similarly, directed edges of weight $-b$ are added from s_0 to p_6 and from p_7 to s_0 in G. In this way, a complete constraint graph incorporating the equi-distance constraints is constructed. And finally, we solve the compaction problem by applying the shortest path algorithm on the complete constraint graph. In IPRAIL, the *Bellman-Ford algorithm* [25] is chosen due to its ability to handle negative-weight edges. The Bellman-Ford algorithm has the worst-case complexity of $O(|V| \times |E|)$, where $|V|$ is the number of vertices and $|E|$ is the number of edges [26].

5.4. Minimization of individual rectangles

The solution obtained from the shortest path algorithm can cause another problem. The shortest path algorithm finds the minimum distance from every variable to the left-most variable, which is the left boundary. This causes some tiles to extend excessively toward its left. An example

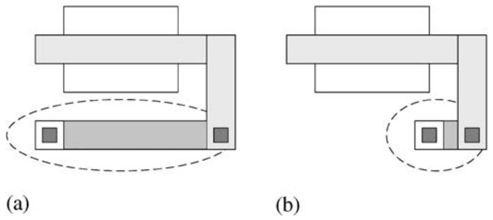


Fig. 16. Example of tile extensions caused by the shortest path algorithm. Shown in circles are two metal-one rectangles: (a) not-minimized and (b) minimized.

is illustrated in Fig. 16. Such extensions introduce unnecessary parasitic resistance and capacitance, which affect the performance of the design adversely. Therefore, after the first feasible solution is obtained, the *individual rectangle minimization* algorithm needs to be performed.

The individual rectangle minimization, applied in IPRAIL, is a modification of the *wire-length minimization* presented in [27]. First, the algorithm locates a *critical path*, which is the shortest path from the left to the right boundary in the constraint graph. The critical path can be determined by employing the shortest path algorithm twice; first from left to right and then from right to left. All variables that possess similar positions belong to the critical path. Tile variables in the path are already minimized and cannot be moved. The rest are indicated as movable tiles, whose areas will be reduced. The algorithm attempts to shift the movable tile variables to their feasible right-most locations, if the moves result in smaller overall tile area. To speed up the process, tile variables are dynamically grouped and ungrouped as they are repositioned towards the right. When the mobility of a tile variable is restricted by another tile variable in the critical path, its optimum position is reached and the variable is not considered for any further operation.

During the process, if moving a tile variable or a group of tile variables decreases the area of one tile and increases that of another, priority is given to the tile in the layer with larger resistance and capacitance. This is accomplished by assigning weight coefficients to all variables in the graph. The algorithm ends if all tile variables are restricted by critical path or if moving the variables does not bring about any further reduction in total tile area.

Similar to the wire-length minimization algorithm, IPRAIL's individual rectangle minimization has the worst-case complexity of $O(N^2 \log N)$, where N is the number of constraint edges. The average-case complexity is almost $O(N)$.

5.5. Output CIF file generation

The shortest path algorithm and the individual rectangle minimization are completed in both horizontal and vertical directions. They provide all rectangles their new positions in the target layout. As the resizing of transistors involves the removal of all diffusion-metal contacts, they are inserted back into the target layout. The number of contacts to be replaced is calculated based on the diffusion-metal overlap area and the design rules for contacts in the new technology process.

As for the passive devices, in case there is a change in device configuration, the original set of tiles is removed from the constraint problem. Thus, new tiles need to be re-constructed based on the new device geometries and device configurations. Finally, the tiles are inserted in their exact positions depending on the temporary dummy tiles. It may be necessary to rotate or flip the tile structures so the devices are aligned with their ports.

6. Examples of layout retargeting using IPRAIL

The IPRAIL-based methodology of retargeting layouts is presented for a single-ended folded-cascode operational amplifier and a two-stage Miller-compensated operational amplifier. Both circuit layouts are initially designed in the TSMC 0.25 μm CMOS process and are retargeted to the TSMC 0.18 μm CMOS process. The new device sizes are obtained from design and simulation of the circuit netlist in the target process.

For both examples, the layout description files in CIF and the original and target technology design rules from Cadence environment are imported to IPRAIL. Once the retargeting process is finished, the regenerated layouts are design-rule checked (DRC). Both original and target layout netlists are extracted and simulated in Hspice to compare their functionalities and performances.

6.1. Single ended folded cascode operational amplifier

Fig. 17 illustrates the schematic of a single-ended folded-cascode operational amplifier. The design consists of 14 transistors. Fig. 18 shows the original layout in TSMC 0.25 μm CMOS process. The transistors are represented in multi-finger structures, which cause the layout to contain 43 distinct unit transistors.

The target layout is generated by IPRAIL, focusing on three main factors. First, three symmetrical axes between transistors are taken into account, depicted as *A*, *B* and *C* in the layout. Second is a set of design rules in TSMC 0.18 μm CMOS process. Last, the new transistor sizes are compiled based on the evaluation and simulation of the schematic netlist in the new process, such that the desired specifications are met.

We employ IPRAIL to retarget the original layout following two different schemes; first, keeping the original device sizes (denoted as original-device-size), and then, imposing the new device sizes (denoted as new-device-size), listed in Table 1. In both schemes, the symmetry axes and the new technology process are supported. The result of original-device-size layout and new-device-size layout are presented in Figs. 19 and 20, respectively. The statistics on the performances and silicon areas are summarized in Table 2.

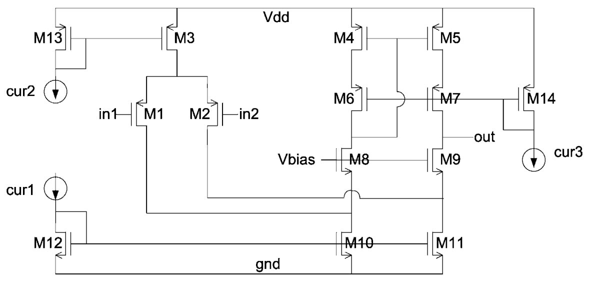


Fig. 17. Schematic of a single-output folded-cascode opamp.

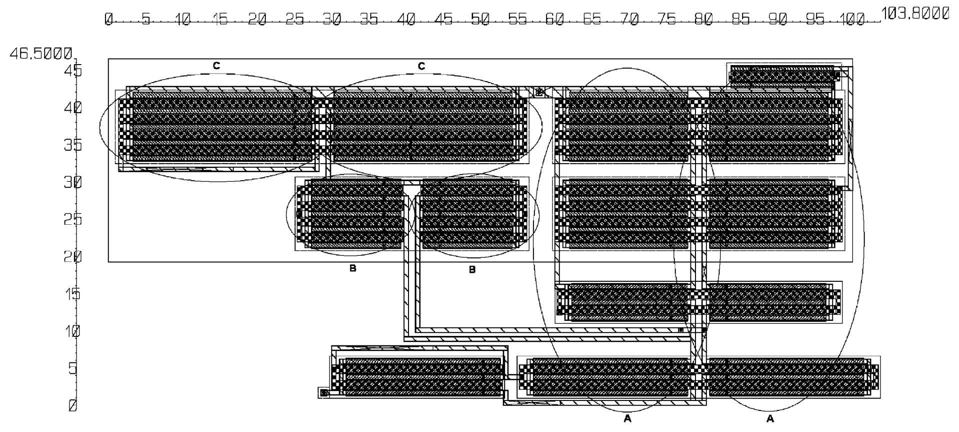


Fig. 18. Original layout of a folded cascode opamp in TSMC 0.25 μm. 'A', 'B' and 'C' are symmetrical transistor block pairs.

Table 1
Transistors sizes of a folded-cascode opamp

Transistors	Total width (μm)/total length (μm)		
	0.25 μm	0.18 μm original-device-size	0.18 μm new-device-size
M1, M2	48.0/1.2	48.0/1.2	33.6/1.4
M3, M13	96.0/1.2	96.0/1.2	56.0/1.4
M4, M5	63.6/1.2	63.6/1.2	15.2/0.9
M6, M7	63.6/1.2	63.6/1.2	15.2/0.9
M8, M9	31.2/1.2	31.2/1.2	6.6/1.3
M10, M11	41.4/1.2	41.4/1.2	36.4/1.3
M12	41.4/1.2	41.4/1.2	36.4/1.3
M14	13.8/1.2	13.8/1.2	6.0/0.9

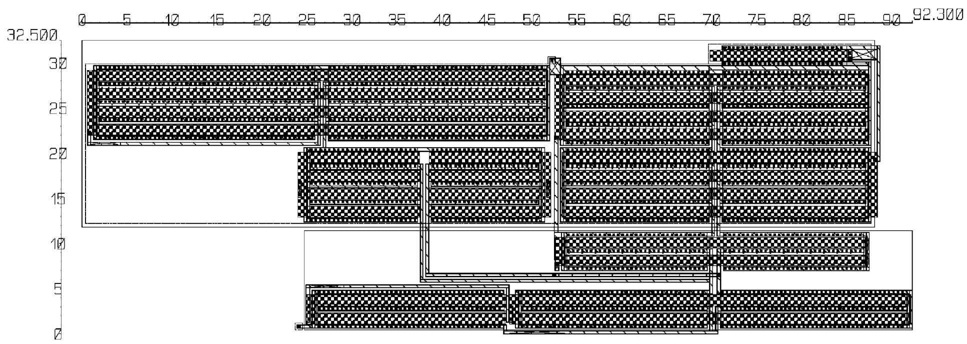


Fig. 19. Target layout of a folded cascode opamp in TSMC 0.18 μm. Original transistor sizes are retained.

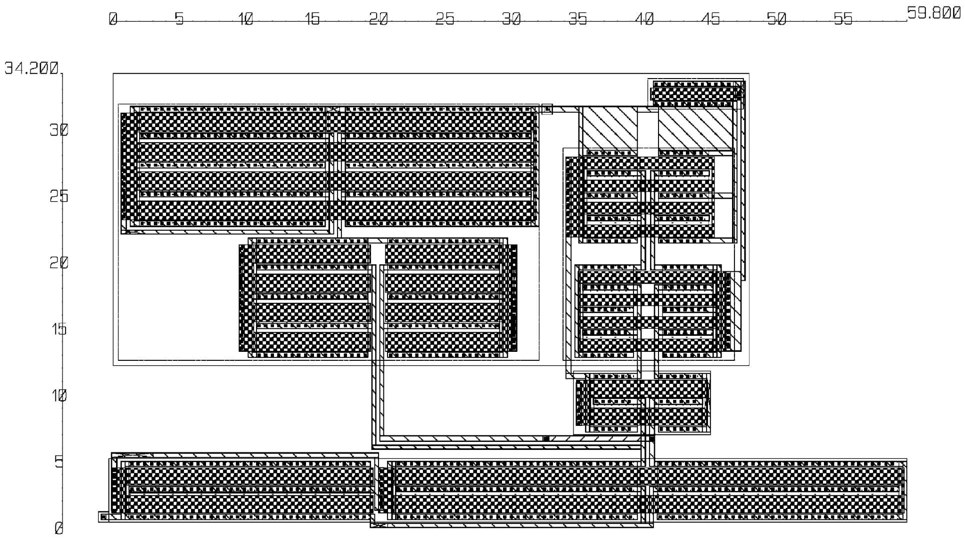


Fig. 20. Target layout of a folded cascode opamp in TSMC 0.18 μm. Transistors are resized according to Table 1.

Table 2
Performances comparison of a folded-cascode opamp

	0.25 μm	0.18 μm original-device-size	0.18 μm new-device-size
Vdd	2.5 V	1.8 V	1.8 V
Load cap.	1.0 pF	0.7 pF	0.7 pF
Gain	60.9 dB	61.9 dB	60.6 dB
Bandwidth	51.7 MHz	71.7 MHz	63.5 MHz
Phase margin	63°	42°	71°
Gain margin	12.5 dB	12.4 dB	10.5 dB
Power	1.48 mW	1.07 mW	0.88 mW
Area	4826.70 μm^2	2995.75 μm^2	2045.16 μm^2

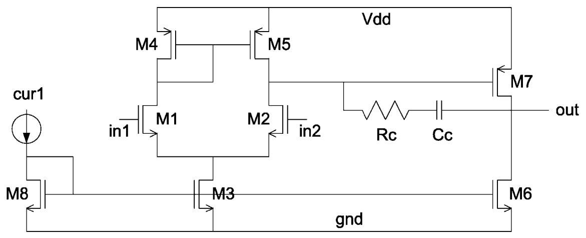


Fig. 21. Schematic of a two-stage Miller-compensated opamp.

6.2. Two-stage miller-compensated operational amplifier

Fig. 21 shows the two-stage Miller-compensated operational amplifier that consists of 8 transistors. Its original layout in TSMC 0.25 μm CMOS process is illustrated in Fig. 22. The compensation capacitor is designed using the MOSCAP and the compensation resistor is laid out

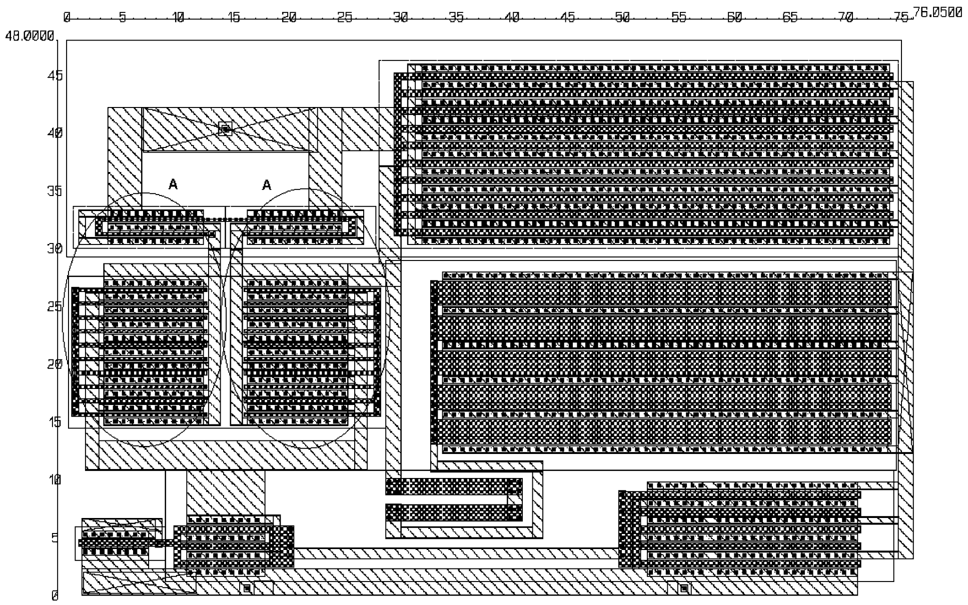


Fig. 22. Original layout of a two-stage opamp in TSMC 0.25 μm. 'A' is a symmetrical transistor block pair.

Table 3
Devices sizes of a two-stage opamp

Transistors or resistors	Total width (μm)/total length (μm)		
	0.25 μm	0.18 μm original-device-size	0.18 μm new-device-size
M1, M2	90.0/0.3	90.0/0.3	80.0/0.3
M3	21.5/0.6	21.5/0.6	21.0/0.6
M4, M5	17.1/0.3	17.1/0.3	17.1/0.3
M6	90.0/0.6	90.0/0.6	90.0/0.6
M7	420.0/0.6	420.0/0.6	210.0/0.4
M8	6.0/0.6	6.0/0.6	6.0/0.6
Cc	200.0/2.1	200.0/2.1	260.0/2.0
Rc	1.35/19.2	1.35/19.2	1.4/45.2

on the poly-silicon mask layer. With the multi-finger structures in both transistors and MOSCAP, the layout contains 48 distinct unit transistors.

Similar to the folded-cascode opamp, we employ IPRAIL twice; first with original-device-size, and then, with new-device-size, whose dimensions are listed in Table 3. The target layouts in TSMC 0.18 μm CMOS process are illustrated in Figs. 23 and 24 for the original-device-size and new-device-size respectively. Table 4 summarizes the performances and area comparison.

The runtime on the folded cascade opamp is 39.2 s and on the two-stage opamp is 37.6 s on a 440 MHz SUN Ultrasparc10 workstation. For each example, the time elapsed on both the original-device-size and new-device-size cases are the same.

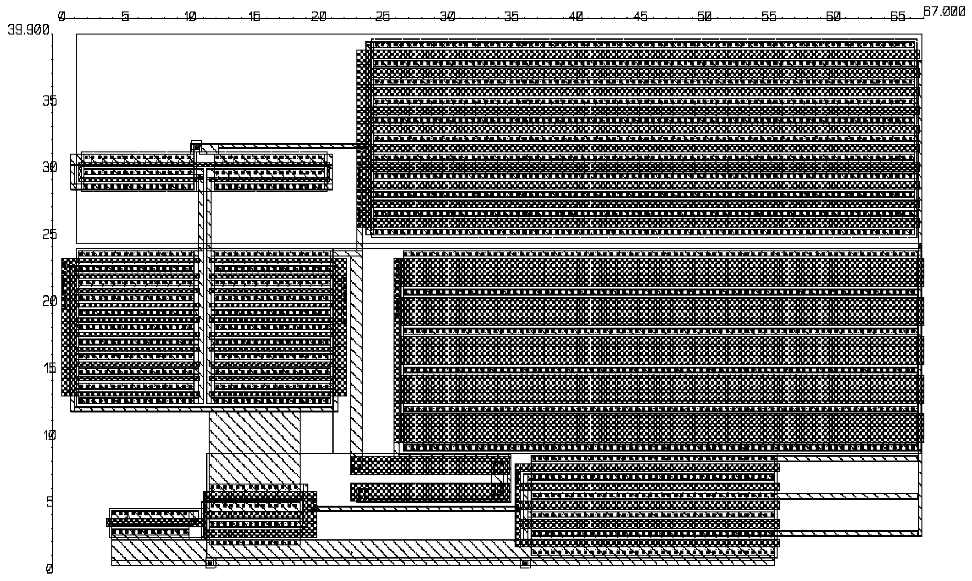


Fig. 23. Target layout of a two-stage opamp in TSMC 0.18 μm. Original transistor sizes are retained.

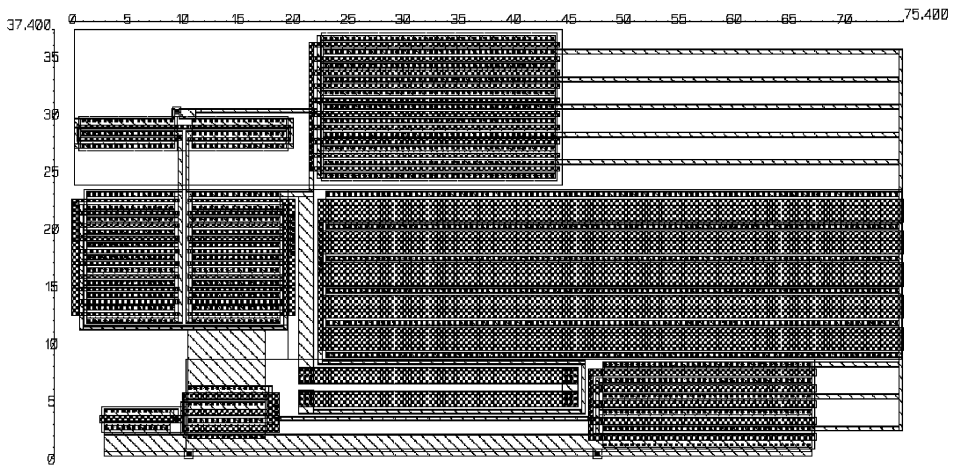


Fig. 24. Target layout of a two-stage opamp in TSMC 0.18 μm. Transistors are resized according to Table 3.

7. Limitations and future work of IPRAIL

As the methodology in the current version of IPRAIL is based on recycling of the original layout, there are a few limitations. First, the target technology process has to cover all the layers presented in the original layout; this is what we called *modestly new* process migration. Second, the new device sizes given to the tool cannot be arbitrary, and has to be resizable on the layout. In particular, if two transistors share the same drain diffusion rectangle, the widths of both transistors have to be the same. Otherwise, it will result in an over-constrained problem. Last,

Table 4
Performances comparison of a two-stage opamp

	0.25 μm	0.18 μm original-device-size	0.18 μm new-device-size
Vdd	2.5 V	1.8 V	1.8 V
Load cap.	1.0 pF	0.7 pF	0.7 pF
Gain	57.7 dB	44.0 dB	64.3 dB
Bandwidth	135 MHz	237 MHz	106 MHz
Phase margin	50°	44°	87°
Gain margin	9.6 dB	9.9 dB	17.4 dB
Power	4.82 mW	3.56 mW	3.46 mW
Area	3650.40 μm^2	2673.30 μm^2	2820.00 μm^2

creating a symbolic template directly from the original layout may limit design configuration. If there is a change in voltage level in the target technology, it may adversely affect the performance of certain design topologies, as well as the compactness of the target layouts. In the current version of IPRAIL, we assume the circuit topology, along with the transistor structures, remains the same when migrating to a modestly new process. Despite these limitations, we have found that IPRAIL is a useful tool for a lot of practical retargeting problems.

While IPRAIL accomplishes the retargeting of the two different operational amplifier layouts fairly easily, several extensions can enhance the benefits of this tool. First, wire-sizing based on current-density and electromigraion can be introduced. Second, an extension to hierarchical retargeting can significantly reduce the solution time for large analog blocks. Third, a multi-finger transistor generation with different number of fingers will allow retargeting the layout with more efficient devices, thereby also improving the overall compactness of the layout. Fourth, the recent progress in representing analog device floorplan and placement with non-slicing topologies [28] can be leveraged in generating efficient templates.

In the current version of IPRAIL, only three categories of design rules—minimum width, minimum spacing, and minimum extension—are considered. Although these design rules are adequate for the technology processes shown in the examples, they may not be sufficient in more recent technology processes. Moreover, there may be technology specific design rules based on structures or population. Thus some modifications might be required. For example, the spacing of vias in one technology is based on the number of vias populated into that particular metal connection. For such case, after generating the structural template, all via-populated areas have to be evaluated and the constraint-weights have to be updated based on the design rules.

8. Conclusions

An automatic analog layout tool, IPRAIL, which is capable of re-targeting the layout to different technology processes, is presented. Layout recycling through symmetry detection and layout integrity conservation scheme is used in order to preserve the analog layout property. Additionally, IPRAIL considers new device sizes to satisfy new specifications as part of the retargeting process. IPRAIL has been applied successfully to migrate some practical CMOS analog circuit layouts.

Acknowledgements

The authors would like to thank Youcef Bourai and Bo Wan for their participation in the early phase of the IPRAIL project, and also Kiyong Choi and Jinho Park for valuable discussions on circuit examples.

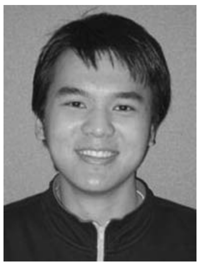
References

[1] N. Jangkrajarng, S. Bhattacharya, R. Hartono, C.-J.R. Shi, Automatic analog layout retargeting for new processes and device sizes, Proceedings of IEEE International Symposium on Circuits and Systems, May 2003, pp. 704–707.
[2] M.J.M. Pelgrom, A.C.J. Duinmaijer, A.P.G. Welbers, Matching properties of MOS transistors, IEEE J. Solid State Circuits 24 (1989) 1433–1440.
[3] A. Hastings, The Art of Analog Layout, Prentice-Hall, Englewood Cliffs, NJ, 2001.
[4] B. Razavi, Design of Analog CMOS Integrated Circuits, McGraw-Hill, New York, 1999.
[5] H. Koh, C. Sequin, P. Gray, OPASYN: a compiler for CMOS operational amplifiers, IEEE Transactions on Computer-Aided-Design of Integrated Circuits and Systems, Vol. 9, February 1990, pp. 113–125.
[6] H. Onodera, H. Kanbara, K. Tamaru, Operational-amplifier compilation with performance optimization, IEEE J. Solid State Circuits 25 (1990) 466–473.
[7] J.D. Conway, G.G. Schrooten, An automatic layout generator for analog circuits, Proceedings of European Design Automation Conference, March 1992, pp. 513–519.
[8] R. Castro-Lopez, F.V. Fernandez, F. Medeiro, A. Rodriguez-Vazquez, Generation of technology-independent retargetable analog blocks, International Journal of Analog Integrated Circuits and Signal Processing, Vol. 33, Kluwer Academic Publishers, Dordrecht, December 2002, pp. 157–170.
[9] M. Aktuna, R.A. Rutenbar, L.R. Carley, Device-level early floorplanning algorithms for RF circuits, IEEE Trans. Comput.-Aided-Design Integrated Circuits Systems 18 (4) (1999) 375–388.
[10] J.M. Cohn, D.J. Garrod, R.A. Rutenbar, L.R. Carley, KOAN/ANAGRAM II: new tools for device-level analog placement and routing, IEEE J. Solid State Circuits 26 (1991) 330–342.
[11] K. Lampaert, G. Gielen, W.M. Sansen, A performance-driven placement tool for analog integrated circuits, IEEE J. Solid State Circuits 30 (1995) 773–780.
[12] E. Malavasi, E. Charbon, E. Felt, A. Sangiovanni-Vincentelli, Automation of IC layout with analog constraints, IEEE Trans. Comput.-Aided-Design Integrated Circuits Systems 15 (1996) 923–942.
[13] S. Rubin, Computer Aids for VLSI Design, Addison-Wesley, Reading, MA, 1987 (Appendix B).
[14] Virtuoso Layout Editor User Guide, Version 4.4.6, Cadence Design Systems Incorporated, 2000.
[15] J.K. Ousterhout, Corner stitching: a data-structuring technique for VLSI layout tools, IEEE Trans. Comput.-Aided-Design Integrated Circuits Systems 3 (1984) 87–100.
[16] D. Marple, M. Smulders, H. Hegen, Tailor: a layout system based on trapezoidal corner stitching, IEEE Trans. Comput.-Aided-Design Integrated Circuits Systems 9 (1) (1990) 66–90.
[17] J.K. Ousterhout, G.T. Hamachi, R.N. Mayo, W.S. Scott, G.S. Taylor, Magic: a VLSI layout system, Proceedings of IEEE/ACM Design Automation Conference, June 1984, pp. 152–159.
[18] W.S. Scott, J.K. Ousterhout, Magic's circuit extractor, Proceedings of IEEE/ACM Design Automation Conference, June 1985, pp. 286–292.
[19] S.L. Lin, J. Allen, Minplex—a compactor that minimizes the bounding rectangle and individual rectangles in a layout, Proceedings of IEEE/ACM Design Automation Conference, June 1986, pp. 123–130.
[20] Y. Bourai, C.J.R. Shi, Symmetry detection for automatic analog layout recycling, Proceedings of Asian and South Pacific Design Automation Conference, January 1999, pp. 5–8.

[21] D. Luenberger, Linear and Nonlinear Programming, 2nd Edition, Addison-Wesley, Reading, MA, 1984.
[22] D. Boyer, Symbolic layout compaction review, Proceedings of IEEE/ACM Design Automation Conference, June 1988, pp. 383–389.
[23] C. Bamji, R. Varadarajan, Leaf Cell and Hierarchical Compaction Techniques, Kluwer Academic Publishers, Dordrecht, 1997.
[24] R. Okuda, T. Sato, H. Onodera, K. Tamaru, An efficient algorithm for layout compaction problem with symmetry constraints, Proceedings of International Conference on Computer-Aided-Design, November 1989, pp. 148–151.
[25] T.H. Cormen, C.E. Leiserson, R.L. Rivest, Introduction to Algorithms, MIT Press, Cambridge, MA, 1990.
[26] N. Sherwani, Algorithms for VLSI Physical Design Automation, Kluwer Academic Publishers, Dordrecht, 1999.
[27] G. Lakhani, R. Varadarajan, A wire-length minimization algorithm for circuit layout compaction, Proceedings of International Symposium on Circuits and Systems, May 1987, pp. 276–279.
[28] F. Balasa, Device-level placement for analog layout: an opportunity for non-slicing topological representations, Proceedings of Asia and South-Pacific Design Automation Conference, January 2001, pp. 281–286.

Nuttorn Jangkrajarng received the B.Eng. degree in Electrical Engineering from Chulalongkorn University, Bangkok, Thailand, in 1997, and the MSEE degree in Electrical Engineering from University of Washington, Seattle, WA, in 1999. He has joined the Mixed-Signal CAD Research Laboratory at the University of Washington and is currently working toward his Ph.D. degree. His research interests are in the field of mixed-signal VLSI and analog IC design automation.

Sambuddha Bhattacharya received his B.Eng. in Electrical Engineering from Birla Institute of Technology and Science, Pilani, India, in 1997. He received his M.S. in Electrical Engineering from the University of Washington, Seattle, in 2002. He has held position as an application engineer with Synopsys, India, and worked on placement, routing and timing convergence issues in digital design. He is currently working towards his Ph.D. degree in Electrical Engineering at the University of Washington. His research interests are in analog design automation techniques and timing and noise issues in digital circuits.



Roy Hartono received the B.S. in Electrical Engineering from University of Washington in 2002. He is currently pursuing the M.S. in Electrical Engineering from University of Washington. Since 2002, he has joined the Mixed-Signal CAD Research Laboratory at the University of Washington. His interests are in VLSI design and automation.

C.-J. Richard Shi (M'91-SM'99) is currently an Associate Professor in Electrical Engineering at the University of Washington. His research interests include several aspects of the computer-aided design and test of integrated circuits and systems, with particular emphasis on analog/mixed-signal and deep-submicron circuit modeling, simulation and design automation. Dr. Shi is a key contributor to IEEE std 1076.1-1999 (VHDL-AMS) language standard for the description and simulation of mixed-signal circuits and systems. He founded IEEE International Workshop on Behavioral Modeling and Simulation (BMAS) in 1997, and has served on the technical program committees of several international conferences. Dr. Shi has authored or co-authored over 100 papers published in international journals and conferences, and has served as the principal investigator of research projects supported by DARPA, SRC and NSF with over $8M funding. Dr. Shi received a Best Paper Award from the IEEE/ACM Design Automation Conference, a Best Paper Award from the IEEE VLSI Test Symposium, a National Science Foundation CAREER Award, and a Doctoral Prize from the Natural Science and Engineering Research Council of Canada. He has been an Associate Editor, as well as a Guest Editor, of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II, ANALOG AND DIGITAL SIGNAL PROCESSING. He is currently an Associate Editor of IEEE Transactions on Computer-Aided Design of Integrate Circuits and Systems.

Hierarchical Extraction and Verification of Symmetry Constraints for Analog Layout Automation[*]

Sambuddha Bhattacharya, Nuttorn Jangkrajarng, Roy Hartono and C-J. Richard Shi

Department of Electrical Engineering, University of Washington
Seattle, WA 98195, USA
{sbb, njangkra, rhartono, shi}@ee.washington.edu

Abstract - Device matching and layout symmetry are of utmost importance to high performance analog and RF circuits. In this paper, we present HiLSD, the first CAD tool for the automatic detection of layout symmetry between two or more devices in a hierarchical manner. HiLSD first extracts the circuit structure from the layout, then applies an efficient pattern-matching algorithm to find all the subcircuits automatically, and finally detects layout symmetry on the portion of the layout that corresponds to extracted subcircuit instances. On a set of practical analog layouts, HiLSD is demonstrated to be much more efficient than direct symmetry detection on a flattened layout. Results from applying HiLSD to automatic analog layout retargeting for technology migration and new specifications are also described.

I Introduction

Variations in the process poly-silicon etch rate, dopant concentration and gradients in temperature, stress and oxide thickness affect the threshold voltage, mobility and current-factors in MOS transistors [1]. These effects on the device characteristics introduce mismatches in transistors that are designed to behave identically. Such mismatches drastically affect analog circuit performance leading to DC offsets, finite even-order distortion and lower common-mode rejection [2]. Symmetric layout of matched transistors alleviates the effects of mismatch in analog/RF circuits.

Device matching and symmetry along with floorplanning, placement and parasitic-driven wiring considerations pose considerable challenge to the automation of analog/RF layouts [2][3]. Over the years, macro-cell based automated placement and routing methodologies have been proposed for analog circuits [4][5]. These layout automation schemes, despite their effectiveness and generality, often fail to incorporate the expertise of the layout designer and are seldom accepted in the industry.

For technology migration and changes in performance specification of analog/RF circuits, a layout reuse methodology promises to be a viable alternative. Such methodologies for analog layout retargeting through layout-template creation by a procedural-language or graphical-user-interface have been proposed in [6][7]. Unfortunately, creation of such templates demands substantial effort from the user. In contrast, [8] recently proposed an automatic layout retargeting methodology for analog circuits, in which an already fined-tuned layout is used to automatically create a *symbolic structural templat*e incorporating floorplan, symmetry and device/wiring alignment information. The new device sizes under retargeting are imposed on the template and the output layout is generated by layout compaction with symmetry constraints [9].

In [8], the axes of symmetry obtained from the existing layout are used as constraints in the structural template. As will be elaborated later, the complexity of such layout retargeting methods is strongly dependent on the number of symmetry axes and corresponding constraints. Therefore, the efficient detection of layout symmetry represents an essential step for the analog layout retargeting process.

An algorithm was proposed in [10] for the detection of layout symmetry. Under this scheme, symmetry detection is accomplished by scanning the entire layout for all horizontally or vertically aligned equi-sized transistors. Unfortunately, this leads to the detection of all unintended axes of symmetry that reside in the layout. Such redundant axes over-constrain the structural template thereby rendering the layout retargeting process computationally expensive.

In this paper, we present a CAD tool, HiLSD (**Hi**erarchical **L**ayout **S**ymmetry **D**etector), which automatically detects layout symmetry based on circuit hierarchy. First, the layout is extracted for the circuit netlist. Then, the circuit hierarchy is established from this flat netlist based on a library of subcircuits that contain device matching information. The detection of the axes of symmetry in the layout is then initiated from the hierarchical netlist. By triggering symmetry detection from the circuit-specific information, HiLSD significantly curtails the search-space and ignores all unintended axes of symmetry that reside in the layout. HiLSD generates a very concise set of symmetry constraints for the automatic layout retargeting process.

Furthermore, in a typical design company, layout and circuit designs are seldom accomplished by the same personnel. For the conscientious circuit designer, HiLSD provides an interactive mode of layout symmetry verification from its graphical user interface.

This paper is organized as follows. Section II discusses the background and the motivation for this work. Section III illustrates the methodology employed for symmetry detection in HiLSD. Section IV explains the process of netlist and hierarchy extraction. Section V describes the actual detection of symmetry from the layout. Section VI presents the experimental results of HiLSD and its application in analog layout automation. Section VII concludes the paper.

II. Background and Motivation

A. Background

A MOS transistor in a layout is defined as an overlap between two rectangles in the *poly-silicon* and *diffusion* mask layers and has three terminals, viz., the gate terminal in the poly-silicon layer and the source and drain terminals in the diffusion layer. Good matching between any pair of transistors is established by laying out the transistors symmetrically. Two transistors are deemed to be symmetric if their layouts are geometric mirror images of each other. As illustrated in Fig. 1, this implies equi-sized channel, drain and source regions, identical orientation and close proximity of the two transistors. For large or multi-fingered transistors, simple geometric mirroring may not establish acceptable matching due to the thermal and process gradients. Such transistor-pairs are often laid-out cross-coupled in one dimension, Fig. 2, or in the two-dimensional cross-coupled form of Fig. 3 also known as the common-centroid layout.

[*] This research has been supported in part by the U.S. Defense Advanced Research Projects Agency's NeoCAD program and in part by the National Science Foundation's ITR program.

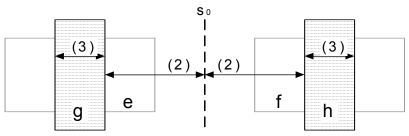


Fig. 1: A simplified layout of two symmetric transistors. Only diffusion and the poly-silicon (dotted) layers are shown. The symmetry axis is denoted by *'s₀'*.

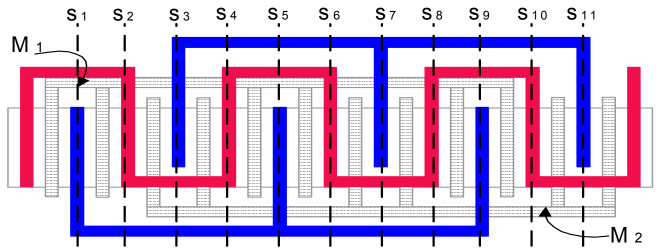


Fig. 2: A one-dimensional cross-coupled symmetric transistor pair. The rectangles with dotted patterns represent the poly-silicon layer.

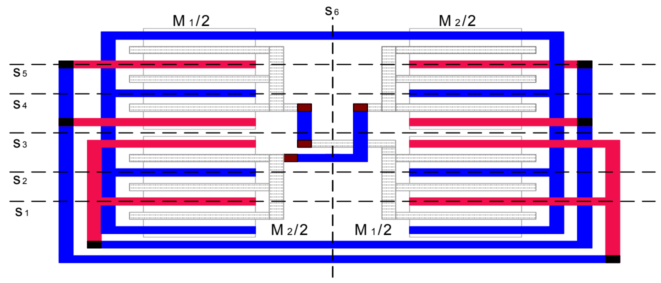


Fig. 3: A common-centroid layout of a symmetric transistor pair. Rectangles with dotted pattern represent the poly-silicon layer.

The layout symmetry detection algorithm presented in [10], henceforth called Direct Layout Symmetry Detection (DLSD), relies on scanning the entire layout for symmetric transistors. First, the nets and transistors in the layout are identified and all transistors are stored in a queue sorted by their bottom-edges. Devices connected by a net and with same ordinate of bottom-edges are then pairwise compared for the existence of geometric mirror images. After detection of all symmetric transistor-pairs, all axes of symmetry with same abscissa or ordinate are merged into a single axis. Under this scheme, the layout of Fig. 2 has eleven axes of symmetry marked by the axes s_1 to s_{11} and sixty-six (selecting 2 from 12) matched transistor pairs. The layout in Fig. 3 has six axes of symmetry as indicated by the axes s_1 to s_6 and thirty matched transistor pairs.

B. *Motivation: Analog Layout Retargeting*

The automatic layout retargeting methodology [8] provides an efficient way of reusing existing fine-tuned analog layouts over changes in technology and design specifications. The re-targeting tool reads in a hand-crafted analog layout, the source and target technology-dependent design rules and automatically creates a symbolic structural template. By imposing the new device sizes pertaining to new specifications on the template, the tool generates a target layout that maintains all the designer expertise embedded in the source layout. The internal flow diagram of the retargeting tool is shown in Fig. 4.

The retargeting tool-suite consists of a template extractor and a layout generator. The symbolic template, extracted from the source layout by the template extractor, comprises the design-rules, connectivity and symmetry constraints. The following equations represent the symmetry constraints generated for the layout of Fig. 1.

$$(e_{top} - f_{top}) = (e_{bottom} - f_{bottom}) = 0 \tag{1}$$

$$(s_0 - g_{right}) - (h_{left} - s_0) = 0 \tag{2}$$

$$(g_{right} - g_{left}) - (h_{right} - h_{left}) = 0 \tag{3}$$

Here, s_0 represents the symmetry axis and all other variables represent the edges of the rectangles. Eq. (1) enforces the alignment at the same ordinate and the equality of the widths of the transistors. The equidistance of the transistors from the symmetry axis is imposed by Eq. (2). The equality of the gate-lengths is enforced by Eq. (3).

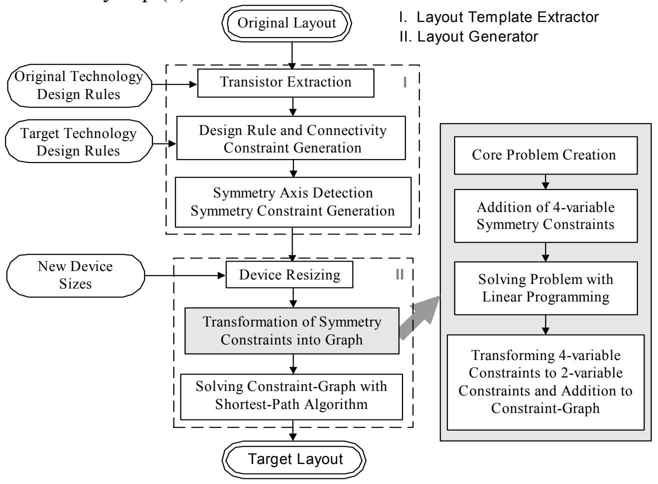

Original Layout
I. Layout Template Extractor
II. Layout Generator
Original Technology Design Rules
Transistor Extraction
I
Target Technology Design Rules
Design Rule and Connectivity Constraint Generation
Symmetry Axis Detection Symmetry Constraint Generation
Core Problem Creation
Addition of 4-variable Symmetry Constraints
New Device Sizes
Device Resizing
II
Solving Problem with Linear Programming
Transformation of Symmetry Constraints into Graph
Transforming 4-variable Constraints to 2-variable Constraints and Addition to Constraint-Graph
Solving Constraint-Graph with Shortest-Path Algorithm
Target Layout

Fig. 4: Internal Flow for template-based layout retargeting.

The problem of the generation of a new layout from the symbolic template reduces to solving a constrained *symbolic compaction* problem [13]. The layout generator tool solves this compaction problem after imposing new device sizes on the symbolic template. While *linear programming* (LP) [14] can be employed to solve this problem, it is computationally intensive and therefore prohibitive for large problems. Therefore, the compaction problem is solved by a combination of linear programming and graph-based shortest-path algorithm [9].

The constraint equations, therefore, need to be transformed into a constraint-graph $G(V,E)$. While the design rule and connectivity constraints can be directly mapped to the constraint-graph, the transformation of the three or four variable symmetry constraints in Eqs. (2) and (3) is rather complex. The steps in the transformation of the symmetry-dictated constraint equations to the graph form have been magnified on the right in Fig. 4. First, the graph $G(V,E)$ obtained from the design rule constraints is reduced to a smaller graph called *core-graph* $G_1(V_1,E_1)$ where $V_1 \subset V$ and $V_1 = \{v_i \,|\, v_i$ corresponds to the variables in the equi-distance constraint-equations$\}$ [9]. The edges of the core-graph are obtained by applying the shortest path algorithm on the main constraint graph $G(V,E)$. A directed edge $e(v_i, v_j)$ is added between the pair of vertices in G_1 if there exists a shortest path between the corresponding vertices in $G(V,E)$.

The LP-compatible equations are generated from the core-graph G_1. The solution of these equations transforms the equidistance constraints in a form that can be directly incorporated into the main constraint graph G. For example, Eq. (2) is transformed into a form

$$s_0 - g_{right} = h_{left} - s_0 = b \tag{4}$$

where b is a constant. Once all the three and four-variable constraint-equations are transformed and added into the main constraint-graph, the symbolic compaction problem is solved using the shortest-path algorithm.

Thus, each symmetry axis introduces numerous variables and necessitates multiple transformations of the constraint-graph into the core-graph [9]. A large number of symmetry axes render the process very computation intensive. Also, as we found during our retargeting experiments, too many redundant symmetry constraints may even render the problem unsolvable. Clearly, reducing the number of symmetry axes and avoiding all redundant constraints is essential for efficient layout retargeting.

III. Hierarchical Symmetry Detection Flow

As discussed in Section II, reduction of symmetry constraints and avoidance of unintentional symmetry is a prime requirement for successful and efficient layout retargeting. The method proposed in this work is based on layout proximity based clustering of netlist and extraction of hierarchy information from the circuit. This is illustrated in Fig. 5.

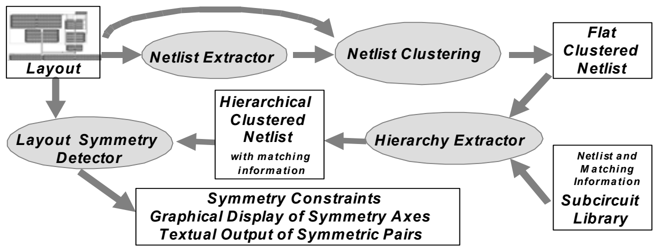

Layout
Netlist Extractor
Netlist Clustering
Flat Clustered Netlist
Hierarchical Clustered Netlist with matching information
Hierarchy Extractor
Layout Symmetry Detector
Netlist and Matching Information
Subcircuit Library
Symmetry Constraints
Graphical Display of Symmetry Axes
Textual Output of Symmetric Pairs

Fig. 5: Hierarchical Symmetry Detection Methodology. The oval blocks are modules of HiLSD.

First, the Netlist Extractor generates the circuit netlist from the layout information. The netlist is then clustered into groups based on physical proximity in the layout. A designer-provided library consists of the netlists of the building blocks, and matching and symmetry information of individual devices. The subcircuits in the library can be any commonly used analog circuit like differential pair, current mirror or larger hierarchical blocks like comparators, operational amplifiers etc. For simple building blocks such as differential pair and current mirror, the matching information is implicitly embedded in the library, whereas for larger complex blocks like operational amplifiers, explicit matching information may be input by the designer. The Hierarchy Extractor identifies all instances of the subcircuit in the main netlist. During this Subcircuit mapping, a complete list of essential and intended matched transistor pairs is created. The detection of symmetric transistors in the layout is initiated from the list obtained after Subcircuit mapping.

Table 1: Outline of the HiLSD algorithm.

```
HierarchicalLayoutSymmetryDetection
begin
    detectNetsTransistors
    clusterTransistors
    for each  $s \in L$   // s = subcircuit, L = Library
        mapSubcircuits
    end for
    detectLayoutSymmetry
end
```

Table 1 shows all the steps in the hierarchical symmetry detection algorithm. The procedures *detectNetsTransistors* extracts the netlist from the layout and *clusterTransistors* groups transistors that are physically contiguous in the layout. The routine *mapSubCircuits* inside the loop identifies all instances of the library subcircuits and maps them to the layout data-structure. This mapping process identifies all the matched transistors that are meant to be symmetric in the layout. Finally, the detection of layout symmetry and generation of constraints are accomplished in the routine *detectLayoutSymmetry*. Each of these processes is explained in detail in Sections IV and V.

IV. Netlist, Cluster and Hierarchy Extraction

A. Netlist Extraction

A transistor with a single rectangle each for its gate, source and drain terminals is henceforth called a *unit transistor*. A net is defined as an electrical connection between the terminals of transistors or external ports.

The layout representation and netlist extraction schemes are adopted from the *Magic* VLSI layout system [11]. Unit transistors are detected by an efficient search for overlaps between the poly-silicon and the diffusion layers. The netlist database stores the location, size, orientation and terminal information for each unit transistor. Once the transistors are extracted, a simple recursive algorithm detects the nets from the layout using the terminals of the transistors as the starting points.

B. Proximity Based Netlist Clustering

The netlist clustering process is especially important as it reduces the number of symmetry axes for multi-fingered transistors. In the layout, each multi-fingered transistor *M* contains multiple contiguous elements *C*, where each contiguous element consists of physically contiguous unit transistors *T*. The clustering scheme partitions the netlist based on the manner in which the transistors are laid out.

The netlist, which at the end of extraction comprised of the set of unit transistors T^S and the set of nets N^S, now consists of the same set of nets N^S and the set of multi-fingered transistors M^S defined as $\{M \mid \forall M \exists \text{ a unique } \{G_M, S_M, D_M\} \subset N^S\}$ where $\{G_M, S_M, D_M\}$ is the set of the gate, source and drain nets of the multi-fingered transistor *M*. Each multi-fingered transistor *M* is a set of physically contiguous elements C^S i.e., $C \in M$ or in other words, $M = C^S = \{C\}$. And each contiguous elements is defined as $C = \{T \mid T \in T^S,\ \forall T\ \{G_T, S_T, D_T\} = \{G_M, S_M, D_M\},\ \textit{and}\ \forall T \in C\ \textit{are physically contiguous}\}$. For the one-dimensional cross-coupled symmetric pair of Fig. 2, each multi-fingered transistor has three contiguous sets of two unit transistors each. In Fig. 3, each multi-fingered transistor has two contiguous sets of three unit transistors each.

Table 2: Algorithm for netlist partitioning.

```
clusterTransistors
begin
    for each  $N \in N^S$      // $N^S$ is the set of nets
        // $G_T, S_T, D_T$ are gate,source,drain nets of transistor T
        for each  $T \in T^S \mid N \in \{G_T, S_T, D_T\}$
            $M$ = checkCreateMFT ( $G_T, S_T, D_T$ )
            $X$ = checkCreateContiguous ( $C^S, T$ )  // $M = C^S = \{C\}$
            insertSorted ( $T, X$ )  // doubly sorted w.r.t. $x, y$ co-ordinates
        end for
    end for
end
```

The *clusterTransistors* procedure in Table 2 presents the algorithm for partitioning the netlist. Each multi-fingered transistor is stored in a hashtable with hash *key* formed by the drain, gate and source nodes. For each unit transistor T connected to a net N, a new multi-fingered transistor M is created if it does not already exist in the hashtable. This is accomplished by a call to the routine *checkCreateMFT*. The routine *checkCreateContiguous* then checks if the unit transistor T is aligned with one of the contiguous elements in M. If T is not physically contiguous with any $C \in M$, a new contiguous element is created. In either case, the routine *insertSorted* inserts T into a list of unit transistors of the corresponding contiguous element. This list of transistors in a contiguous element is doubly sorted with respect to the x and y coordinates.

C. *Hierarchy Extraction*

The designer-intended transistor-matching information is embedded in the subcircuits in the library. Identifying instances of these commonly used subcircuits in the main netlist maps the non-redundant matching information to the devices in the layout. This is accomplished by an efficient subgraph isomorphism algorithm [12] in the *mapSubcircuits* routine of Table 1.

First, both the subcircuit and the main circuit are implicitly partitioned by an iterative labeling algorithm to reduce the search space. This identifies a set of nodes in the main circuit and a single node, called a *key* node, in the subcircuit. The set of nodes in the main circuit obtained by this iterative labeling algorithm are potential start-points for checking a pattern match with the subcircuit. From each potential node in the main circuit and the key node in the subcircuit, another labeling algorithm accomplishes detection of an isomorphism with the subcircuits graph.

V. Layout Symmetry Detection

The hierarchy extraction process generates a subcircuit-based netlist. From the subcircuit-based netlist, a list of designer-intended non-redundant matched multi-fingered transistor pairs is created. The layout symmetry detection scheme identifies if each pair of these multi-fingered transistors is actually laid out symmetrically. The process also generates the corresponding constraints for the ensuing compaction step in layout automation[8].

The algorithm for layout symmetry detection is shown in Table 3. For each transistor pair intended to be matched, the *detectTopology* routine identifies the pair's layout topology by traversing through the list of contiguous elements. Based on the topology, the unit transistors are inserted into two or four sorted lists. Thus, for the common-centroid topology of Fig. 3, the six unit transistors in the top and bottom halves of the transistors M_1 and M_2 respectively are collected into a list L_L. The bottom and top halves of M_1 and M_2 are collected into another list L_R. The unit transistors in L_L and L_R are then pairwise compared in the *checkSymmetry* routine to detect the vertical axis of symmetry, s_6, and generate the corresponding constraints. For the horizontal symmetry axis s_3, the bottom halves of both M_1 and M_2 are collected into a list L_B, and the top halves are collected into a list L_T and pairwise compared. For the layout of Fig. 2, six unit transistors are inserted into each list L_L and L_R and a single axis of symmetry s_6 is detected. Prior co-ordinate based double sorting of the unit transistors in each multi-fingered transistor ensures that pairwise comparison can detect axes of symmetry.

VI. Results

A. *Symmetry Detection Experiments*

The HiLSD program was employed to detect symmetry in various analog/RF layouts and generate constraints for the layout retargeting methodology [8] illustrated in Fig. 4. Table 4 compares the symmetry detection data for HiLSD with the DLSD method presented in [10]. Various symmetry topologies were employed on the different layouts. The differential amplifier, the latched comparator and the 4:1 comparator used symmetric transistors with minimal multi-fingered structures. The voltage-controlled oscillator was laid out with extensive multi-fingered symmetric transistors. The two-stage and folded-cascode operational amplifiers utilized multi-fingered interleaved and common-centroid symmetry topologies. And the 5-bit flash analog-to-digital converter consisted of 31 instances of a latched-comparator laid out in an array of 8x4.

For each method, the number of symmetry axes detected, the number of symmetric transistor pairs, and the number of constraints due-to-symmetry are reported. The DLSD method extracted a large number of redundant symmetry axes. As it detected symmetry between every pair of unit transistors in each multi-fingered transistor, a large number of axes were observed for the two-stage operational amplifier and the voltage-controlled oscillator circuits. For the array structure of the comparator blocks in the 5-bit analog-to-digital converter, the DLSD method detected symmetry for every transistor in one comparator cell to every transistor in another comparator cell in the same row and column. These redundant constraints not only slowed down the compaction steps in layout retargeting, but also rendered the problem unsolvable in some cases.

Table 3: Algorithm for symmetry detection.

```
detectLayoutSymmetry
begin
    // ListSym = { ($M_i$ , $M_j$) | $M_i$ and $M_j$ are intended matched pair }
    for each ( $M_i$ , $M_j$ ) ∈ ListSym
        topology = detectTopology ( $M_i$ , $M_j$ )
        if ( topology == common_centroid ) then
            $L_L$ = insertToList ( $M_i$ , $M_j$ , left )
            $L_R$ = insertToList ( $M_i$ , $M_j$ , right )
            $L_B$ = insertToList ( $M_i$ , $M_j$ , bottom )
            $L_T$ = insertToList ( $M_i$ , $M_j$ , top )
            checkSymmetry ( $L_L$ , $L_R$ )
            checkSymmetry ( $L_B$ , $L_T$ )
        else if ( topology == horizontal_interleaving ) then
            $L_L$ = insertToList ( $M_i$ , $M_j$ , left )
            $L_R$ = insertToList ( $M_i$ , $M_j$ , right )
            checkSymmetry ( $L_L$ , $L_R$ )
        else if ( topology == vertical_interleaving ) then
            $L_B$ = insertToList ( $M_i$ , $M_j$ , bottom )
            $L_T$ = insertToList ( $M_i$ , $M_j$ , top )
            checkSymmetry ( $L_B$ , $L_T$ )
        else        // simple transistor layout
            checkSymmetry ( $M_i$ , $M_j$ )
        end if
    end for
end
```

We compare the scaling of the symmetry detection by the two methods with arrays of comparators. Fig. 6 shows the number of symmetric transistor pairs detected by DLSD and HiLSD as the number of comparators is scaled. The y-axis is in logarithmic scale. The graph shows that DLSD detects a huge number of redundant symmetry axes.

Table 4: Comparison between Hierarchical Symmetry Detection (HiLSD) and Direct Symmetry Detection (DLSD)

Circuits	# Multi-Fingered Transistors	# Unit Transistors	Design Rule Constraints	DLSD			HiLSD		
				Symmetry Axes	Transistor Pairs	Symmetry Constraints	Symmetry Axes	Transistor Pairs	Symmetry Constraints
Differential Amplifier	5	5	1,602	1	2	18	1	2	18
Latched Comparator	15	20	8,639	10	19	132	2	6	42
2-stage Opamp	9	48	5,902	69	262	1,578	2	12	78
Folded Cascode Opamp	14	43	8,352	29	173	1,206	6	20	168
4:1 Comparator	20	32	26,182	26	166	1,026	3	13	78
VCO	16	198	645,986	680	5,525	33,156	4	362	2,178
5-bit Flash ADC	435	590	320,937	261	6,218	4,193	12	186	1,302

B. *Automatic Analog Layout Retargeting with HiLSD*

We performed experiments on analog layout retargeting [8] to a new technology and specifications based on the hierarchical and direct symmetry detection methods. Fig. 7 shows a comparator layout in TSMC 0.25um CMOS process. This layout was retargeted under new specifications to the TSMC 0.18um CMOS technology using both DLSD and HiLSD based symmetry detection.

The symmetry constraints generated by the two methods were passed onto the resizing tool. Table 5 shows the number of symmetry axes, transistor pairs, symmetry constraints, and user runtime for the resizing schemes under the two methods. The retargeted layout obtained by using HiLSD for symmetry detection is shown in Fig. 8. The retargeted layout under this preserved all the required matching considerations, while incorporating a lesser number of symmetry constraints. The circuit performance of the latched-comparator in the two technologies achieved by these methods is reported in Table 6.

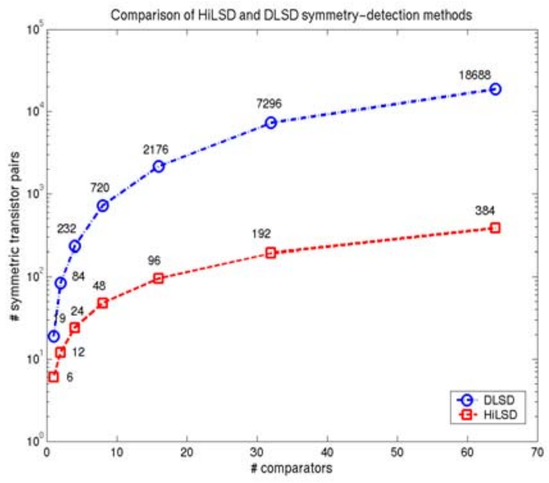


Fig. 6: Comparison of HiLSD and DLSD symmetry detection for array of comparators. X-axis represents number of comparators in the array. Y-axis denotes number of symmetric transistor pairs (Log scale).

The analog comparator section of the 5-bit flash analog-to-digital converter was constructed by placing 31 units of the latched-comparator into an 8x4 array. Each unit comparator was aligned and matched with other units in the same row and column. For any unit comparator in the section, another comparator corresponding to its preceding or following bit was positioned next to each other to minimize the mismatch. The layout of the comparator section of the ADC in TSMC 0.25um CMOS technology is shown in Fig. 9.

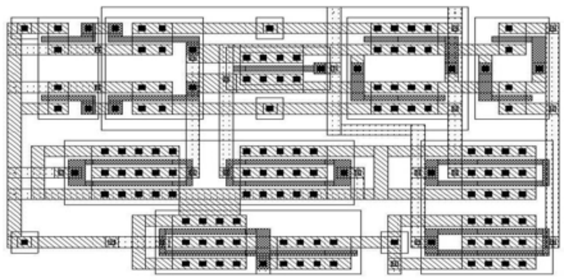


Fig. 7: Comparator Layout in TSMC 0.25um technology.

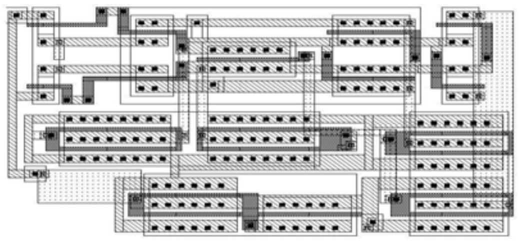


Fig. 8: Retargeted Layout of comparator in TSMC 0.18um

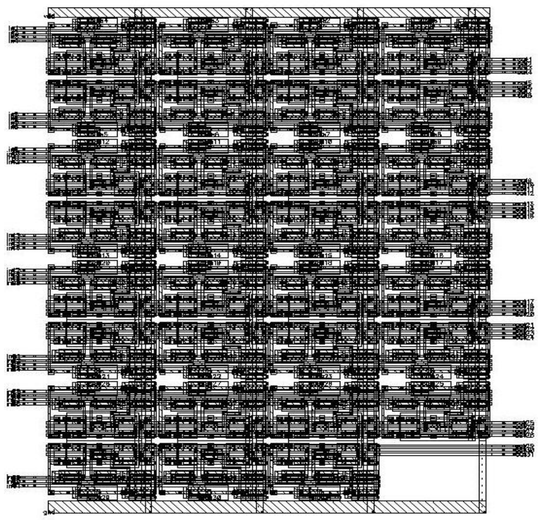


Fig. 9: Comparator block of a 5-bit flash ADC in TSMC 0.25um.

The flash analog-to-digital converter was retargeted to the TSMC 0.18um CMOS process; first with the symmetry information obtained from DLSD method and then with the HiLSD algorithm. DLSD detected 6,218 symmetric transistor pairs in the layout, while the HiLSD method identified only 186 symmetric pairs. This huge difference in detected symmetric pairs is due to the redundant symmetric pairs from the transistors on different unit comparators

Table 5: Comparison between layout retargeting with DLSD and HiLSD symmetry detection schemes.

Design	Latched Comparator		5-bit Flash Analog-to-Digital Converter	
Symmetry Detection Method	DLSD	HiLSD	DLSD	HiLSD
Multi-fingered Transistors	15	15	435	435
Unit Transistors	20	20	590	590
Design Rule Constraints	8,639	8,639	320,937	320,937
Symmetry Axes	10	2	261	12
Transistor Pairs	19	6	6,218	186
Additional Symmetry Constraints	132	42	41,943	1,302
Runtime on Solving Symmetry Constraints	0.65 s	0.26 s	2 hr 36 min	48 min
Total Runtime for Retargeting Tool	10.06 s	9.31 s	4 hr 20 min	2 hr 14 min

listed in same row or column. During resizing, these unnecessary symmetric-pairs resulted in the increase of symmetry constraints from 1,302 to 41,943, which subsequently increased the runtime of solving the symmetry constraints from 48 minutes to 156 minutes. The overall runtime escalated from about 2 hours to 4 hours. Nevertheless, both target layouts showed similar symmetries and matching. The original layout had an area of 12,780 um^2. The target layout from direct symmetry detection had an area of 7,955 um^2. And the target layout for hierarchical symmetry detection had an area of 6,984 um^2. The reduction in area is attributed to the avoidance of unwanted axes of symmetry that constrain the layout. The retargeted layout obtained through HiLSD method is shown in Fig. 10.

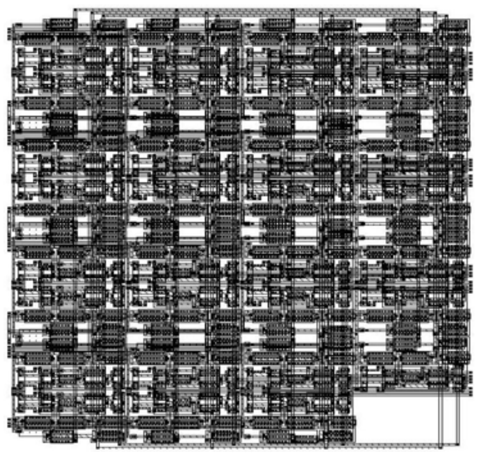


Fig. 10: Retargeted Layout of comparator block of ADC in TSMC 0.18um (HiLSD based symmetry detection).

Table 6: Comparison of a latched-comparator performance

Design Specs.	Layout in 0.25um	Retargeted Layout in 0.18um	
	-	DLSD Based	HiLSD Based
Power Supply	2.5 V	1.8 V	1.8 V
Ref. Voltage	1.28 V	1.28 V	1.28 V
Frequency	500 MHz	750 MHz	750 MHz
Resolution	20 mV	20 mV	20 mV
Area	369 um^2	225 um^2	214 um^2
Power	0.84 mW	0.45 mW	0.45 mW

VII. Conclusions

A new symmetry detection tool, Hi-LSD, based on hierarchical extraction and subcircuit specific symmetric transistor pairs is presented. The tool significantly reduces search-space and ignores all unintended symmetry axes exhibited on the layout. Employing Hi-LSD on a 5-bit flash analog-to-digital converter ignores all unintended axes of symmetry and reduces the number of symmetric pairs from 6,218 to 186. When applied with the automatic analog layout-retargeting tool, the runtime for regenerating the new ADC layout is reduced from 4 hours to 2 hours.

With the symmetry constraints described in a hierarchical circuit netlist by circuit designers, Hi-LSD also provides the first automatic tool for verifying if a layout meets all the symmetry constraints required by circuit designers.

Acknowledgements

The authors wish to thank Mr. Sankaran Aniruddhan, System-On-Chip Lab, University of Washington, for help with the design examples.

References

[1] M. J. M. Pelgrom, A. C. J. Duinmaijer and A. P. G. Welbers, “Matching properties of MOS transistors”, *IEEE J. Solid-State Circuits*, vol. 24, pp. 1433-1440, Oct. 1989.

[2] B. Razavi, *Design of Analog CMOS Integrated Circuits*, McGraw Hill, 2001.

[3] A. Hastings, *The Art of Analog Layout,* Prentice Hall, 2001.

[4] J. M. Cohn, D.J. Garrod, R. A. Rutenbar and L. R. Carley, “KOAN/ANAGRAM II: New tools for device-level analog placement and routing”, *IEEE J. Solid State Circuits*, vol. 26, pp. 330-342, Mar. 1991.

[5] K. Lampaert, G. Gielen and W. M. Sansen, “A performance-driven placement tool for analog integrated circuits”, *IEEE J. Solid State Circuits*, vol. 30, pp. 773-780, Jul. 1995.

[6] J. D. Conway and G. G. Schrooten, “An automatic layout generator for analog circuits”, *Proc. European Design Automation Conference,* pp. 513-519, Mar. 1992.

[7] R. Castro-Lopez, F. V. Fernandez, F. Medeiro and A. Rodriguez-Vazquez, “Generation of technology-independent retargetable analog blocks”, *Int. J. Analog Integrated Circuits and Signal Processing,* vol. 33, pp. 157-170, Dec. 2002.

[8] N. Jangkrajarng, S. Bhattacharya, R. Hartono, and C-J. R. Shi, “Automatic analog layout retargeting for new processes and device sizes”, *Proc. IEEE Int. Symposium Circuits and Systems*, vol. 4, pp. 704-707, May 2003.

[9] R. Okuda, T. Sato, H. Onodera and K. Tamaru, “An efficient algorithm for layout compaction problem with symmetry constraints”, *Proc. Int. Conference Computer-Aided-Design*, pp. 148-151, Nov. 1989.

[10] Y. Bourai and C. J. R. Shi, “Symmetry detection for automatic analog layout recycling”, *Proc. Asian and South Pacific Design Automation Conference,* pp. 5-8, Jan. 1999.

[11] W. S. Scott and J. K. Ousterhout, “Magic’s circuit extractor”, *Proc. IEEE/ACM Design Automation Conference*, pp. 286-292, Jun. 1985.

[12] M. Ohlrich, C. Ebeling, E. Ginting and L. Sather, “Subgemini: Identifying subcircuits using a fast subgraph isomorphism algorithm”, *Proc. IEEE/ACM Design Automation Conference,* pp. 31-37, Jun. 1993.

[13] S. L. Lin and J. Allen, “Minplex – a compactor that minimizes the bounding rectangle and individual rectangles in a layout”, *Proc. IEEE/ACM Design Automation Conference*, pp. 123-130, Jun. 1986.

[14] D. Luenberger, *Linear and Nonlinear Programming* 2nd Edition, Addison-Wesley, 1984.

Multiple Specifications Radio-Frequency Integrated Circuit Design with Automatic Template-Driven Layout Retargeting

Nuttorn Jangkrajarng, Sambuddha Bhattacharya, Roy Hartono, and C-J. Richard Shi

Department of Electrical Engineering, University of Washington
Seattle, WA 98195, USA
{njangkra,sbb,rhartono,cjshi}@ee.washington.edu

Abstract – This paper presents an automatic layout retargeting tool that generates analog and RF layouts incorporating new device sizes and geometries based on new circuit specifications. A graph-based symbolic template is automatically constructed from a practical layout such that expert designer knowledge embedded in the layout is preserved. The template can be solved for multiple layouts based on different device sizes and geometries, satisfying several different specifications. Symmetry conservation and passive device modification are also embedded in the tool. The retargeting tool is demonstrated on a voltage controlled oscillator to generate three layouts with different target goals. While manual re-design is known to take days to finish, the automatic layout retargeting tool takes a few hours to generate a reusable template and takes minutes to generate comparable layouts.

I. Introduction

The ability to integrate digital, analog and radio frequency (RF) circuits on to the same silicon chip, known as *system-on-chips* or *SOC*, has revolutionized the semiconductor industry. The portability and economy that results from integrating multiple functions on a single chip, nevertheless, is accompanied with an escalating complexity. This, together with the added pressure of aggressive design cycle, not only demands innovation in the field of computer-aided-design (CAD), but also necessitates the adoption of the design-reuse philosophy.

Continued advances in the CAD tools and the cell-based design methodology have already addressed these issues in the design of digital circuits. Unfortunately, CAD tools for analog/RF design still await major innovations. Indeed, design reuse in the analog/RF domain is often limited to only the circuit topology. Significant trade-offs between the major design goals like gain, bandwidth, stability, noise reduction, linearity and power minimization necessitate considerable amount of re-design. In addition, analog/RF circuit performance is strongly affected by layout styles and layout designers often need to use their expertise to eke out the required design specifications.

Fortunately, significant progress has been made recently in the form of optimization tools that synthesize analog/RF circuits for target specifications [1]. Automatic layout generation based on optimizations coupled with floorplanning, placement and routing of pre-designed macro-cells has also been reported in [2] and [3]. Despite their effectiveness and generality in obtaining desired circuits in various specification ranges, these layout automation schemes require extensive computation and at times fail to incorporate the expertise of the layout designer. Therefore, these methods are seldom adopted in the industry.

On the other hand, *template based* methods that require designer involvement provide a viable alternative. A high-quality template, created once, can be reused for multiple layout generation under different specifications. One such approach based on template generation with the Virtuoso Parameterized Cell tool has been proposed in [4]. Founded on the same principle of design reuse, another approach of automatically retargeting analog layouts was presented in [5]. While, the template creation in [4] requires substantial effort from the user and is very time-consuming, the method in [5] presents a scheme for automatic generation of a structural template from an existing layout. In this method, an already fine-tuned layout is used to automatically create a *symbolic structural template* incorporating the floorplan, symmetry and device/wiring alignment information. The new device sizes under changes in performance specification are imposed on the template, and the layout is realized by layout compaction with symmetry constraints [6].

In this paper, we propose, for the first time, a template based layout retargeting tool for RF integrated circuits. While being based on the same general principle as [5], this work adds substantial innovations in numerous aspects. Firstly, unlike analog circuits, RF design requires extensive handling of passive devices. Changes in specification of RF circuits require major modifications in the shapes, structures and sizes of on-chip spiral inductors, capacitors, and resistors. The RF layout retargeting method proposed in this paper handles such changes in shapes and sizes of passive devices. Secondly, RF layouts operating at gigahertz frequencies oftentimes incorporate innumerable number of vias for performance requirements. A naive template creation with hundreds of thousands of vias is extremely computationally intensive. This work provides a novel scheme for reduction of template size in the presence of such large number of vias. Thirdly, an automatic symmetry detection scheme is also employed to preserve device matchings. Finally, the automatic symbolic template created in the process can be used to generate multiple high-quality RF layouts for different design specifications.

The rest of the paper is organized as follows. Section II discusses the overall retargeting methodology and the symbolic structural template. Innovation in automatic symmetry detection, via/contact removal, and passive device retargeting are explained in Sections III, IV, and V respectively. Section VI presents the result of the retargeting tool on the voltage controlled oscillator. Section VII concludes the paper.

[*] This research has been supported in part by the U.S. Defense Advanced Research Projects Agency's NeoCAD program and in part by the National Science Foundation's ITR program.

II. Analog and RF Layout Retargeting via Structural Symbolic Template

The proposed method for automatically retargeting analog and RF layout is based on a *recycling* scheme. Fig. 1 illustrates the flow and interface of a structural template based layout retargeting tool. First, a structural symbolic template is constructed from an already fine-tuned layout. The template contains circuit topology, connectivity, design rules, placement and matching information of the layout. Next, new sets of *device sizes*, obtained through simulation, for each target specification are imposed on the structural template. Finally, target layouts for each specification are obtained by solving the enhanced templates.

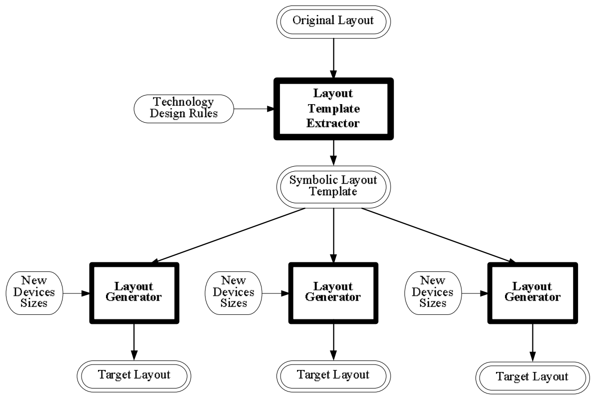


Fig. 1: Flow of the symbolic template-based layout retargeting tool.

A. Layout template Extraction

The main tasks of the layout template extractor are to identify active and passive devices, to detect device matching, and to assemble a structural symbolic template. Fig. 2 shows specific tasks of the layout extractor.

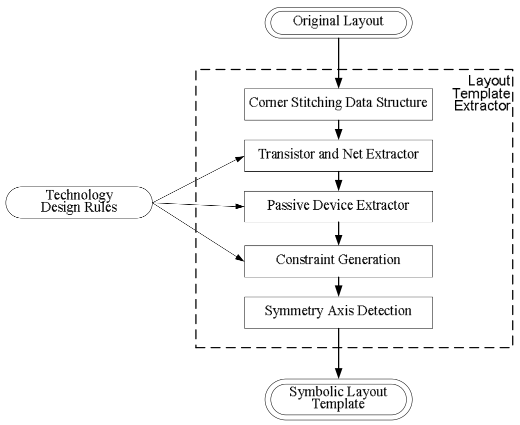


Fig. 2: Layout template extractor flow.

First, the original layout is stored in the *corner-stitching* data structure [7]. Here, each rectangle in the layout is stored explicitly as a *tile* and is linked to its neighboring tiles on lower-left and upper-right corners. Our preference for corner-stitching over other data structures, for example bins and linked-lists, is dictated by its efficiency in fast localized searches.

After the layout is stored, the MOSFET transistors are identified based on the overlap of multiple layers as defined in the technology design rules. Nets are then detected by searching for the connected neighboring tiles from all transistor terminal tiles in a depth-first-search manner. If vias or contacts are encountered, the search continues on different layers.

Specific layout patterns for passive devices are pre-defined in the technology design-rules. Passive devices, categorized into resistors, capacitors and inductors, are identified by net-traversing through the tiles and pattern-matching. Details of the passive device extraction and regeneration are discussed in section V.

As mentioned earlier, the original circuit topology, connectivity and matching have to be examined and reused in order to preserve design knowledge and integrities. Hence, the structural symbolic template has to possess an ability of maintaining the original layout's intellectual properties, an adaptability with new device sizes evaluated from new specifications, and a fast solvability. For these requirements, a *constraint graph*, which is widely used in the context of layout compaction [8], is selected. Here, each rectangular tile is represented by four independent nodes: left, right, top, and bottom tile edges. Constraints are placed between nodes in the graph to sustain layout integrity and correctness. These constraints are categorized into (1) connectivity, (2) design-rule, (3) symmetry, and (4) exact-device-size. Horizontal and vertical constraint graphs are constructed separately.

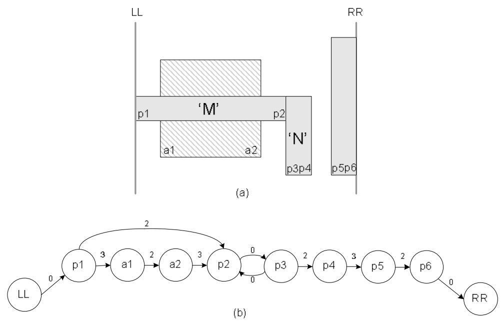


Fig. 3: Example of a constraint graph as symbolic template in horizontal direction.

Consider a sample layout of Fig. 3. The tile connectivity between *M* and *N* in the horizontal direction is retained by two constraint arcs of weight *'0'* between the edges *p2* and *p3*. The design rule constraints can be further sorted into three groups as minimum width (an arc from *p3* to *p4*), minimum spacing (an arc from *p4* to *p5*), and minimum extension (an arc from *a2* to *p2*). However, constraints due to symmetry, described in Fig. 4 such as the equal distance between *a1* to *a2* and *a3* to *a4*, cannot be directly added to the constraint graph. The handling of these constraints is explained in Section IIB.

Clearly, generating constraints from each node to every other nodes leads to significant redundancies. For example, adding a direct spacing constraint from *p2* to *p5* in Fig. 3 is redundant, because the minimum distance rule is already imposed by several arcs through *p3* and *p4*. To avoid this, a scan-line method [8] is employed.

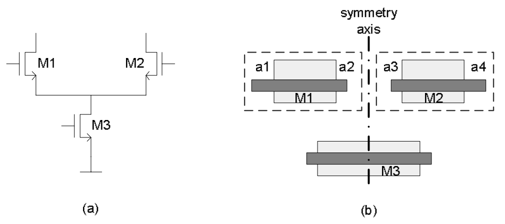


Fig. 4: An example of symmetry between transistors.

In the scan-line method, first, all rectangle edges are sorted by their abscissas. For each edge, constraint arcs are generated only from that edge to all other visible edges to its left, i.e. not blocked by other tiles. Progressing from the most-left to the most-right, the list of visible edges is updated when each scanned edge is completed. For example in Fig. 5, only arcs from *a* and *c* to *s* are included but not the one from *b* to *s*. The algorithm has a time complexity of $O(n^2)$, where *n* is the number of rectangle tiles.

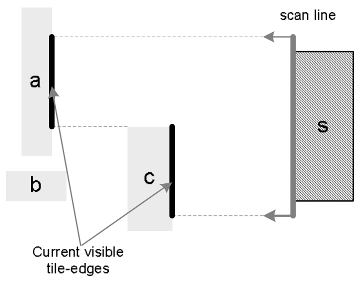


Fig. 5: Example of a scan line and its visible edges.

B. Layout Generation

After the structural symbolic template is constructed, different target layouts can be obtained by a two-step process: first, imposing new device sizes by modifying constraint arcs in the template, and second, solving the template through a combination of a linear programming and graph-based shortest-path algorithm. The detail procedures of the layout generator are shown in Fig. 6.

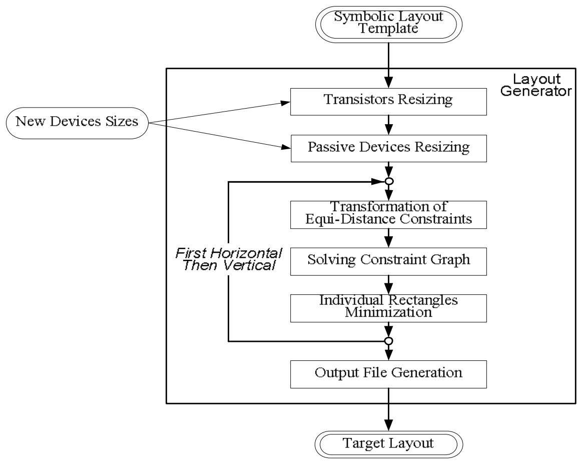


Fig. 6: Layout generator flow.

The new transistor sizes can be obtained from circuit simulations and optimizations performed manually by design engineers or automatically by circuit synthesis tools. Based on these, exact-device-size constraint arcs will be added to the graph symbolic template. For an example in Fig. 3, if the new transistor width is '*w*', an arc from *a1* to *a2* with weight '*w*' and an arc from *a2* to *a1* with weight '$(-1)\times w$' will be added to the graph. Extra care in transistor resizing is focused on the active-metal contacts. If the size of a transistor is smaller in the target layout, the drain and source area may not be able to accommodate the original number of contacts. In this case, the template has to be updated so that the number of contacts can fit within the specific area. The algorithm for this is further explained in section IV.

Retargeting of the passive devices requires modification in device geometries and structures. A special scheme is implemented and is described in section V.

The assembled constraint graph template can be solved by applying the graph-based shortest-path algorithm on both horizontal and vertical templates separately [8]. The algorithm finds the shortest distance for every rectangle edge to the left (or bottom) layout boundary. This determines legitimate tile locations of the target layout, based on the design rules and the device sizes. The shortest-path algorithm has a worst-case time complexity of $O\ (nodes \times arcs)$. However, symmetry information cannot be preserved by utilizing this algorithm only.

Prior to the graph solving, the symmetry constraints has to be converted into a form that can be imposed on the constraint graph [6]. At the beginning, a smaller *equivalent graph*, which consists of only nodes that appear in symmetry constraints, is created. The constraint arcs between nodes in the equivalent graph are present only when there is a direct path between a pair of nodes in the original graph template. Then, this equivalent graph is converted into a linear-programming equation form, in which graph-nodes are represented as variables and graph-constraint-arcs are represented as weight constrained equations between two variables. Together with the equations preserving the symmetry, the problem can be solved with integer linear programming, and the exact distance between each pair of variables can be computed. That distance is then imposed individually to the original graph template, thus the symmetrical distances are retained. For example in Fig. 4, if the distance between (*a1*,*a2*) and (*a3*,*a4*) is 'w_s', then four arcs – from *a1* to *a2* weight 'w_s', from *a2* to *a1* weight '$(-1)\times w_s$', from *a3* to *a4* weight 'w_s', and from *a4* to *a3* weight '$(-1)\times w_s$' – will be added. Afterward, the shortest-path algorithm can be carried out.

One weakness of the shortest-path algorithm is all edges are pulled toward most-left (or most-bottom), which creates excessive leftward extension in most rectangles. Therefore, an algorithm for minimizing individual rectangle areas as described in [9] has to be implemented to secure good layout.

III. Automatic Symmetry Detection

Since device symmetry in analog/RF layouts is critical to circuit performances [10], this information has to be detected and maintained while generating target layouts. On a small layout, symmetries can be manually located easily. However, on a large and complex layout, especially on one containing transistors laid out in multi-finger fashion, an automatic approach has to be implemented.

First, a circuit netlist clustering all multi-finger transistors is extracted. By mapping it to known small-sized circuit netlists with easily identified symmetries, all the multi-finger transistor symmetrical pairs are located. After that, by sorting each unit transistor in the multi-finger group based on its relative position, each unit transistor symmetrical pair is detected. This symmetry information is then used to maintain the device matching, as explained in section II.

Fig. 7 shows an example of symmetric multi-finger transistors of *M1* and *M2*. The symmetry between *M1* and *M2* can be specified straightforwardly from the schematic diagram. Once the transistor-clustered netlist is created and mapped, *M1* and *M2* in the layout are declared symmetric.

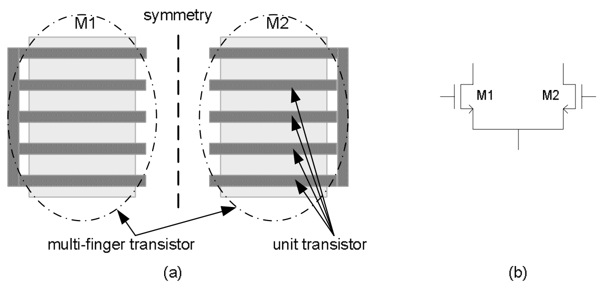


Fig. 7: Multi-finger transistor clustering for automatic symmetry detection.

IV. Via and Contact Removal

Due to high current density in analog and RF circuits, it is a common practice to layout connecting metal wires considerably wider than the design-rule minimum width. This practice also benefits the circuit performance as the parasitic effect of wire resistance and thermal noise is reduced [10].

Similarly, connections between two layers using vias or contacts also have to conduct large current. Therefore, high quality analog or RF layouts are frequently implemented with arrays of vias or contacts. Typically, one connection can contain 10's to 1000's vias or contacts.

This collection of vias or contacts exhibits two main challenges in the retargeting tool. First, it increases the size of the structural symbolic template, which results in longer template extraction time, longer layout generation time, and larger memory usage. Second, the presence of a fixed number of vias or contacts might limit the shrinkability of the tiles. Therefore, the following scheme is implemented to overcome these challenges.

During the layout representation in the corner-stitching data structure, each connected and overlapped rectangle-pairs in different layers are searched for the vias/contacts. The whole array of vias/contacts are then represented by only four rectangle tiles: top, bottom, left, and right, as shown in Fig. 8. Once the scan-line method is carried out and the structural symbolic template is created, two extra constraint arcs, *a* and *b*, are added to set the width and height of each array. Weights of those arcs are determined based on the number of vias/contacts on each connecting tiles and their related design rules. After solving the template for the target layout, vias/contacts in each connection have to be re-populated based on the available space.

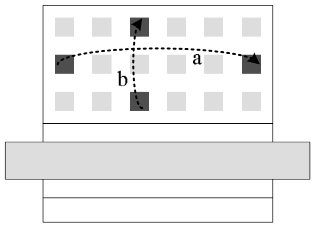


Fig. 8: Example of reduction in number of contacts. Only four contacts are kept for each rectangle pair.

The number of vias/contacts on the target layout can be adjusted based on the original rectangle sizes and aspect ratios, the original numbers of vias/contacts, or the new device sizes. Moreover, this will allow further sizing of connection metal wires based on parasitic effect or on current density.

V. Passive Device Retargeting

Along with transistors, passive devices – resistors, capacitors, and inductors – are also significant parts of analog/RF circuits. Retargeting those devices exhibits one critical issue as their geometries and rectangle structures might be modified. Merely updating rectangle widths and lengths are certainly not adequate. Here, a method of detection, representation in the template, and creation of new passive devices is presented.

Prior to the passive devices detection, the layout transistor netlist has to be extracted, in which all passive devices are embodied within the nets. Traversing through tiles in each net, a passive device is detected when the defined tiles structure is matched and the threshold geometry size is satisfied. Upon recognition, the traversed net is split, the device's geometry information is collected, and the port tiles are collected. A port is defined as a tile that connects a passive device to a corresponding net. The connection can occur on the same layer masks or on different layer masks connected through vias or contacts.

Examples of resistors and capacitors are shown in Fig. 9. On-chip resistors are generally implemented in poly-silicon layer as it exhibits high linearity and low capacitance to substrate [10]. In some technologies, n-well or active layer resistors are also used. The resistor topology supported consists of a single tile strip (Fig. 9a) or serially-connected multiple unit resistors (Fig. 9b). On-chip capacitors are commonly laid out in one of the two schemes. Firstly, a *P-P* (poly-silicon to poly-silicon) or *MIM* (metal insulator metal) capacitor (Fig. 9c) which is detected when two tiles of defined layers from two different nets overlap each other with adequate overlap area. Secondly, a MOSFET can also be used as a capacitor, referred as a MOSCAP, by shorting the source and drain terminals. Due to its structure, retargeting MOSCAPs is identical to transistors.

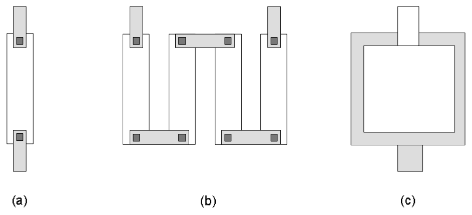


Fig. 9: Example of (a) single tile resistor. (b) multiple unit resistor. (c) MIM capacitor.

An inductor is commonly constructed with a planar spiral structure. The example of two-turns inductor is shown in Fig. 10a. This inductor structure is detected when a set of tiles of a defined inductor layer assembles at least one full turn. Limited by the corner-stitching data structure, only square shape inductors are supported.

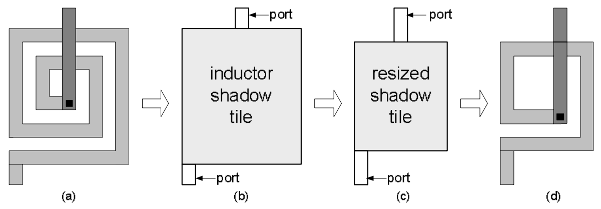


Fig. 10: Retargeting process of a planar square spiral inductor.

In the analog/RF layout, passive devices are usually isolated in space from other devices and nets to minimize the parasitic and coupling effects [10]. Therefore, to prevent an overlapping with other devices or net during the resizing process, a *shadow tile* is added on top of the passive device prior to the symbolic template generation. A shadow tile is a temporary non-physical-layer tile. It is used to allocate a dedicated area for the passive device by applying minimum spacing constraints to tiles on every layer.

Updating the passive device sizes is accomplished by adding exact-device-size constraints to the corresponding device tile variables for new sizes, and the shadow tile variables for reserving its dedicated space. In addition, more constraints are added between the shadow tile and the port tile variables in order to keep the passive device aligned and preserve the net connectivity after compaction.

Retargeting a more complex passive device structure, such as multi-turn inductor in Fig. 10a, requires a reconstruction of the tiles. First, in Fig. 10b, all device tiles, except for ports, are virtually removed from the layout and the entire device is represented by a single shadow tile in the symbolic template. During the layout generation process, the shadow tile area is adjusted according to the target device dimension, illustrated in Fig. 10c. Once the template is solved for a layout, the new passive device is independently regenerated, inserted into the allotted space, and re-connected to the corresponding ports, as shown in Fig. 10d. The new inductor is retargeted based on the geometry information, such as number of turns, outer and inner diameters, width, and spacing. These geometries can be obtained from a technology-based inductor library or through the inductor approximate expressions, for instance one proposed in [11].

VI. Experimental Result on Retargeting VCO Layouts

The automatic retargeting layout tool has been tested on RF circuits. The *CMOS complementary cross-coupled voltage controlled oscillator* was initially designed and laid out using the MIT Lincoln Laboratory's 0.18um low power FDSOI CMOS process. The VCO consists of 6 MOSFET transistors, 2 MOSCAP transistors, 2 output buffers, and 1 planar square spiral inductor. Each buffer is constructed of 3 transistors and one poly-silicon resistor. The schematic diagrams are shown in Fig. 11.

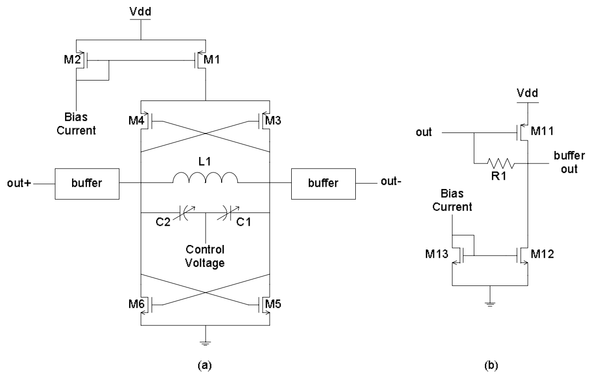


Fig. 11: Schematic diagram of (a) VCO and (b) buffer block.

The original layout is shown in Fig. 12. Each transistor is represented in multi-finger structures. The layout comprises of 202 unit transistors and 11 electrically connected nets.

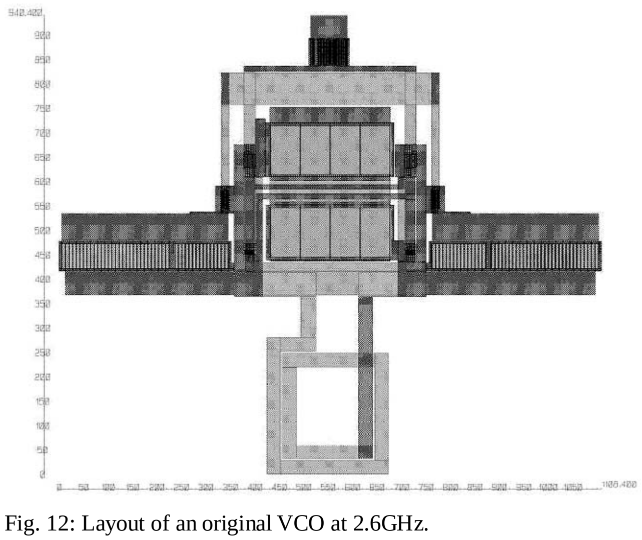


Fig. 12: Layout of an original VCO at 2.6GHz.

The tool was employed on the initial layout to generate a structural symbolic template. Originally, there were 310,673 rectangle tiles in the layout. After via/contact removal, the layout size was reduced to only 3,210 rectangle tiles. As for passive devices, two poly-silicon resistors and one planar spiral inductor were detected, and their shadow tiles were created. Since the target layouts would carry different inductor geometries, all the tiles constituting an inductor were removed. Then, the matching information between transistors was collected. The automatic symmetry detection based on netlist matching reported two symmetrical axes with 79 pairs of symmetrical unit transistors from the layout. From Fig. 11, those were obtained from *M3*:*M4*, *M5*:*M6*, *C1*:*C2*, and *M11*, *M12* and *M13* between left and right buffers.

Finishing the extractor phase, the structural symbolic template, in a graph constraint form, consisted of 11,884 nodes, 205,878 arcs and 1,272 additional arcs extracted from symmetry constraints.

The template was utilized to generate three target layouts. Different sets of device sizes were manually designed based on different specifications. Next, transistor sizes and passive device geometries were enforced onto the template. Once the retargeting process was finished, the target layouts were design-rule checked (DRC), and their netlists were extracted and simulated in SpectreRF for functionality and performance verification. Each target layout specification, along with the initial layout specification, is listed in Table 1. Target layout III, which was designed for 5.6GHz oscillating frequency, is illustrated in Fig. 13.

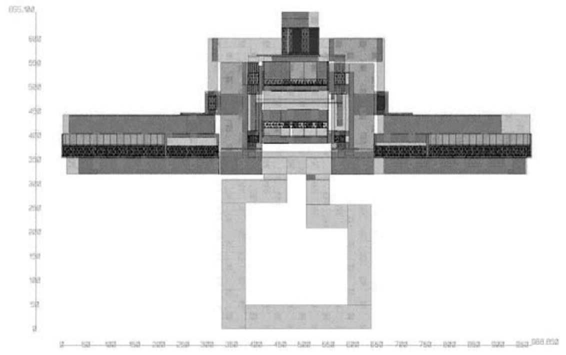


Fig. 13: Layout of a target VCO at 5.6GHz

Table 1. Specifications of original layout and target layouts

	Original Layout	Target Layout I	Target Layout II	Target Layout III
Oscillating Freq.	2.6 GHz	3.8 GHz	4.5 GHz	5.6 GHz
Phase Noise at 60kHz	-98 dBc/Hz	-96 dBc/Hz	-103 dBc/Hz	-98 dBc/Hz
Tuning Range	6 %	5 %	7 %	4 %
Output Swing	2.0 V	2.0 V	2.2 V	2.4 V
Inductor	1.0 nH	0.5 nH	0.5 nH	0.4 nH
Power	7.4 mW	9.5 mW	5.2 mW	7.0 mW
Area	1.04 mm^2	0.74 mm^2	0.71 mm^2	0.63 mm^2

The automatic retargeting program was run on a 900MHz SUN UltraSparc3 workstation. The layout template extractor phase took 1 hour 48 minutes. Once the template was created, the generation of target layout I was completed in 7.56 minutes, target layout II in 8.43 minutes, and target layout III in 9.19 minutes.

VII. Summary

In this paper, a tool capable of retargeting analog/RF layouts to different specifications is presented. A structural symbolic template is automatically generated so that the analog/RF layout integrity is preserved. New transistor sizes and passive device geometries, obtained from various specifications, can be imposed onto the template to generate several target layouts. The tool was applied successfully to produce three different operational VCO layouts within minutes after the template had been extracted.

Despite the restriction in circuit topology changes due to template extraction, the retargeting tool exhibits several key benefits. Its speed, its template reusability, and the fact that it does not require much designer involvement make this a great potential tool for many analog and RF circuit retargeting applications.

References

[1] M. D. M. Hershenson, A. Hajimiri, S. S. Mohan, S. P. Boyd, and T. H. Lee, "Design and optimization of LC oscillators", *Proc. of IEEE/ACM Int. Conference on CAD*, pp. 65-69, November 1999.

[2] M. Aktuna, R. A. Rutenbar, and L. R. Carley, "Device-level early floorplanning algorithms for RF circuits", *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 18, pp. 375-388, April 1999.

[3] K. Lampaert, G. Gielen and W. Sansen, "A performance-driven placement tool for analog integrated circuits", *IEEE Journal of Solid State Circuits*, vol. 30, pp. 773-780, July 1995.

[4] R. Castro-Lopez, F. V. Fernandez, F. Medeiro and A. Rodriguez-Vazquez, "Generation of technology-independent retargetable analog blocks", *Int. Journal of Analog IC and Signal Processing*, Kluwer Academic Publishers, vol. 33, pp. 157-170, December 2002.

[5] N. Jangkrajarng, S. Bhattacharya, R. Hartono, and C-J. R. Shi, "Automatic analog layout retargeting for new processes and device sizes", *Proc. of IEEE Int. Symposium on Circuits and Systems*, vol.4, pp. 704 -707, May 2003.

[6] R. Okuda, T. Sato, H. Onodera and K. Tamaru, "An efficient algorithm for layout compaction problem with symmetry constraints", *Proc. of IEEE/ACM Int. Conference on CAD*, pp. 148-151, November 1989.

[7] J. K. Ousterhout, "Corner stitching: a data-structuring technique for VLSI layout tools", *IEEE Trans. on CAD of Integrated Circuits and Systems*, pp. 87-100, January 1984.

[8] S. L. Lin and J. Allen, "Minplex – a compactor that minimizes the bounding rectangle and individual rectangles in a layout", *Proc. of IEEE/ACM Design Automation Conference*, pp. 123-130, June 1986.

[9] G. Lakhani and R. Varadarajan, "A wire-length minimization algorithm for circuit layout compaction", *Proc. of IEEE Int. Symposium on Circuits and Systems*, pp. 276-279, May 1987.

[10] B. Razavi, *Design of analog CMOS integrated circuits*, McGraw Hill, 2001.

[11] S. S. Mohan, M. D. M. Hershenson, S. P. Boyd, and T. H. Lee, "Simple accurate expressions for planar spiral inductances", *IEEE Journal of Solid State Circuits*, vol. 34, pp. 1419-1424, October 1999.

CrtSmile: A CAD Tool for CMOS RF Transistor Substrate Modeling Incorporating Layout Effects

Zhao Li, Ravikanth Suravarapu[*], Roy Hartono, Sambuddha Bhattacharya, Karti Mayaram[*], Richard Shi

Department of Electrical Engineering, University of Washington, Seattle, WA 98195, USA
[*]School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR 97331, USA
{lz2000, rhartono, sbb, cjshi}@ee.washington.edu [*]{suravara, karti}@ece.orst.edu

Abstract: This paper presents a new CAD tool *CrtSmile*, which automatically incorporates transistor layout effects for CMOS RF transistor modeling with an emphasis on substrate resistance extraction. The RF transistor layouts in the CIF/GDSII format are used to generate a layout dependent substrate model that can be included as a subcircuit with the BSIM3 device model. To support multi-finger RF transistor layout/bulk recognition, a pattern based layout extraction method is presented. *CrtSmile* incorporates a scalable substrate model for multi-finger transistors, which is dependent on transistor layout/bulk patterns and geometric layout information, such as the number of gate fingers, finger width, channel length, and bulk contact locations. This model is simple to extract and gives good agreement with the measured data for a 0.35μm CMOS process. A low noise amplifier design is evaluated with the new layout dependent substrate model and the proposed tool, showing the importance of CMOS RF transistor layout on substrate resistance modeling.

1. Introduction

At Giga hertz frequencies, non-quasi-static channel effects and the distributed nature of the gate and the substrate should be correctly modeled to accurately predict the MOS transistor behavior. Further, the scalability, accuracy and efficiency are three important factors in choosing the model. The influence of the substrate on MOS transistor performance has been studied in [1][5][8][10] where substrate effects have been incorporated into standard device models by attaching an external resistance network. BSIM4 [2] further applies a complex internal five-resistor substrate network to model the substrate. It has been shown that the simple one-resistor substrate network in [1] (Fig. 1) is accurate up to 10 Ghz and the substrate resistance is weakly dependent on biasing conditions [11].

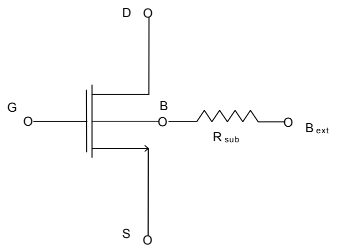


Figure 1. RF MOS transistor substrate network model of [1].

Previous work in substrate modeling is not sufficient for high frequency RF applications since transistor layout effects are not properly modeled. In general, RF transistors are realized by different kinds of multi-finger layout patterns and substrate contact locations to reduce parasitic effects and the layout areas. For example, Figs. 2(a) and 2(b) show two standard multi-finger transistors with different bulk contact patterns. In Fig. 2(a), the direction of bulk contacts is perpendicular to the finger direction. In Fig. 2(b), the direction of bulk contacts is parallel to the finger direction. Even though these two transistors have the same device size, their substrate resistances are different due to their different bulk contact patterns. The layout dependence of substrate resistance will become even severe for an interleaving multi-finger layout pattern as shown in Fig. 2(c) since the substrate is shared by two interleaving transistors. Therefore, a substrate model that does not consider multi-finger layout/bulk patterns will lead to incorrect results.

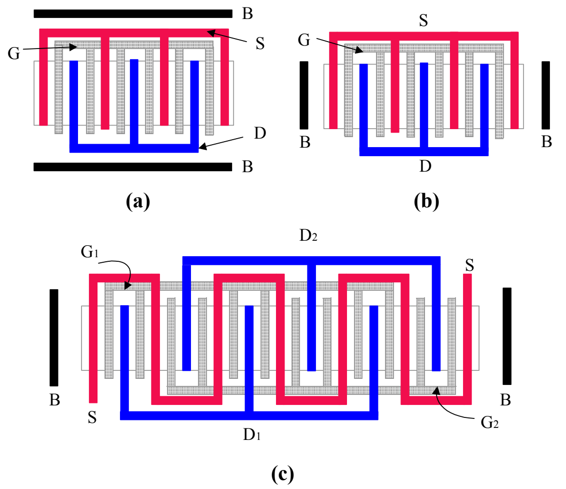


Figure 2. (a) Standard multi-finger transistor layout with perpendicular bulk contacts. (b) Standard multi-finger transistor layout with parallel bulk contacts. (c) Interleaving multi-finger transistor layout with parallel bulk contacts.

In addition to layout/bulk patterns, the substrate resistance is also dependent on geometric layout information, such as the number of fingers, finger width, channel length, bulk contact locations, etc. Recently [4][11] have proposed scalable substrate resistance models. In this paper, we present a new CAD tool *CrtSmile* (*C*MOS *R*F *T*ransistor *S*ubstrate *M*odeling *I*ncorporating *L*ayout *E*ffects), which automatically extracts layout/bulk patterns and geometric layout information from input layout files and accordingly generates layout dependent substrate models. A new scalable layout dependent substrate model based on [1] for different layout/bulk patterns and geometric layout parameters is also incorporated.

This paper is organized as follows. Section 2 introduces the system architecture of *CrtSmile*. The pattern based RF transistor layout extraction method is discussed in Section 3. Section 4 presents a scalable layout dependent substrate model. Measurement data on substrate resistance and a low noise amplifier example are shown in Section 5. Finally, Section 6 concludes this paper.

2. System architecture

The system architecture of *CrtSmile* is shown in Fig. 3. It can be used for both post-layout verification and pre-layout design.

During post-layout verification, a key task for layout extraction is to provide all the required geometric layout information as well as the layout/bulk pattern styles for the next step of substrate model generation. Therefore, a pattern based layout extraction program is developed with input layout files in the CIF/GDSII format. After the layout/bulk pattern and the geometric layout information are extracted, the layout dependent substrate model generation module is called and the final substrate resistance is calculated. For a pre-layout design, users should specify the candidate layout/bulk pattern style and all the required geometric layout parameters.

After the layout dependent substrate model generation, a behavioral substrate model is generated with geometric parameters as inputs instead of a specified substrate resistance value. The behavioral substrate model is realized with a sub-circuit in a SPICE-like simulator and attached to the BSIM3 device model for RF circuit design and optimization.

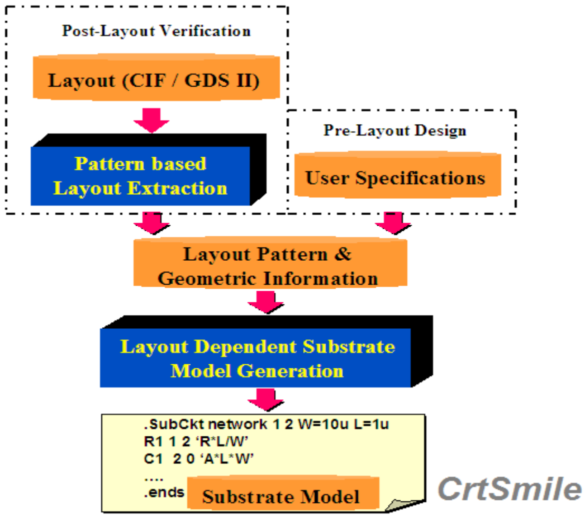

Post-Layout Verification
Layout (CIF / GDS II)
Pattern based Layout Extraction
Pre-Layout Design
User Specifications
Layout Pattern & Geometric Information
Layout Dependent Substrate Model Generation
.SubCkt network 1 2 W=10u L=1u
R1 1 2 'R*L/W'
C1 2 0 'A*L*W'
....
.ends
Substrate Model
CrtSmile

Figure 3. System architecture of *CrtSmile.*

CrtSmile is flexible for incorporating different kinds of substrate networks based on user defined substrate models. Figure 4 shows a substrate network for a four-finger transistor with parallel bulk contacts based on the BSIM4 substrate model. The substrate is a heavily doped one as in Fig. 1 of [12]. In Fig. 4, BI is the intrinsic bulk contact and the back plane is generally connected to the ground. The circled part in Fig. 4 is the BSIM4 substrate model for a single finger. Similar substrate networks can be derived based on other substrate models. In this paper, we will focus on the substrate network model due to [1].

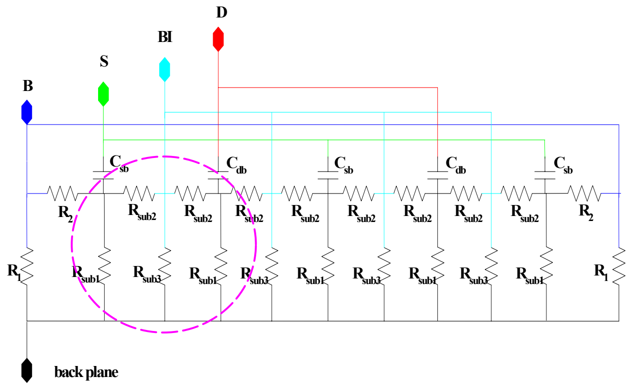


Figure 4. A substrate network based on [2] for a 4-finger transistor.

3. RF transistor layout extraction

Given a layout file in CIF or GDSII formats, the RF transistor layout extraction program is required to recognize different kinds of layout/bulk patterns as well as provide geometric layout information. For a RF transistor layout, there are a few different kinds of layout styles, such as the standard multi-finger layout as in Figs. 2(a) and 2(b), the interleaving layout as in Fig. 2(c), and the dual-gate layout as in Fig. 3 of [6]. Since the substrate resistance is also determined by bulk contact locations, there are also different kinds of bulk contact layout styles that need to be considered. Examples include parallel bulk contacts (Fig. 2(b)), perpendicular bulk contacts (Fig. 2(a)), and guard rings. A traditional layout extraction program will only recognize discrete finger transistors and cannot build the connection between the transistor active regions and bulk contact locations. Therefore, a new pattern based layout extraction program is developed to address these two problems.

The flow of the pattern based RF layout extraction program is shown in Fig. 5.

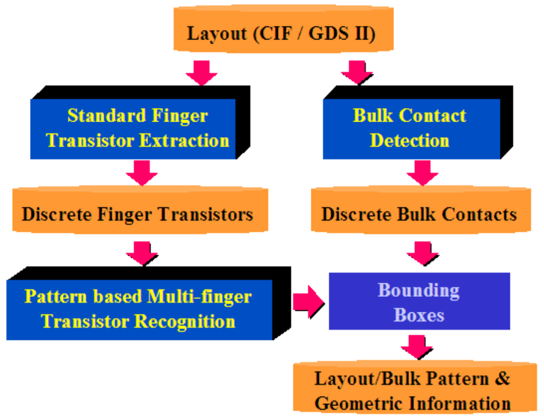

Layout (CIF / GDS II)
Standard Finger Transistor Extraction
Bulk Contact Detection
Discrete Finger Transistors
Discrete Bulk Contacts
Pattern based Multi-finger Transistor Recognition
Bounding Boxes
Layout/Bulk Pattern & Geometric Information

Figure 5. Pattern based RF layout extraction flow.

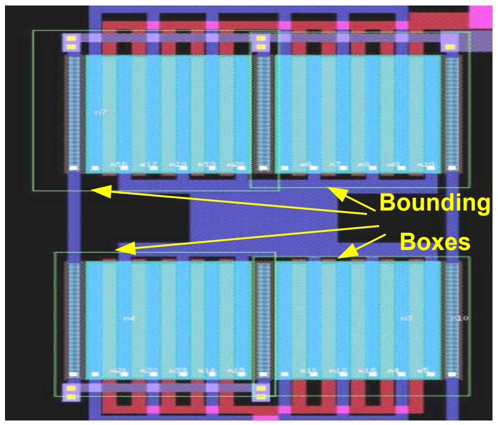


Figure 6. Four multi-finger transistors with four bounding boxes.

First, standard finger transistor extraction and bulk contact detection are performed to have discrete finger transistors and bulk contacts. After that, the pattern based multi-finger transistor recognition module is called to recognize different kinds of user defined multi-finger layout styles based on different electrical connectivity patterns. Then bounding boxes are applied to relate discrete bulk contacts to proper multi-finger transistors and recognize corresponding bulk contact layout styles. Figure 6 shows a bounding box example, in which four bounding boxes are highlighted to extract four multi-finger transistors with parallel bulk contacts. Bounding boxes can be added manually or automatically with the assumption that only bulk contacts close enough to the transistor active regions are related to the corresponding multi-finger transistors. Finally, layout/bulk pattern and geometric layout information is retrieved. The extracted circuit netlist for one of the multi-finger transistors in Fig. 6 is shown below:

```
.subckt T1 nd ng ns gnd Wf=13.00u Lf=0.80u Nf=5 Db=0.60u
m1 nd ng ns nbi NMOS w=Wf l=Lf ad='Wf*1.00u'
+as='Wf*0.50u' pd='2*(Wf+1.00u)' ps='Wf+1.00u'
m2 nd ng ns nbi NMOS w=Wf l=Lf ad='Wf*0.50u'
+as='Wf*0.50u' pd='Wf+1.00u' ps='Wf+1.00u' m='Nf-2'
m3 nd ng ns nbi NMOS w=Wf l=Lf ad='Wf*0.50u'
+as='Wf*1.00u' pd='Wf+1.00u' ps='2*(Wf+1.00u)'
xsub nbi gnd subres_para w='Wf*Nf' l=Lf n=Nf d=Db
.ends
```

where W_f is the finger width, L_f is the channel length, N_f is the number of gate fingers and D_b is the distance of bulk contacts from the transistor active region. The substrate resistance for the parallel bulk contact layout style is represented by a subcircuit (*subres_para*), which is dependent on geometric layout parameters and will be discussed in Section 4.

4. Scalable layout dependent substrate model

In this section, the substrate resistance (R_{sub}) for multi-finger transistors with parallel bulk contacts is first studied and is further extended to those with perpendicular bulk contacts.

4.1 Substrate model for parallel bulk contacts

A DC extraction of the substrate resistance is used, which is a modification of the method described in [7]. As shown in Fig. 7, p^+ bulk straps replace all gate fingers. A small voltage is applied to the appropriate bulk strap and the other bulk straps are grounded to obtain the substrate resistance of a specified finger. Meanwhile, the bulk contacts and the drain/source contacts are all grounded. The ratio of the applied voltage to the current through the bulk contacts gives the substrate resistance of that finger.

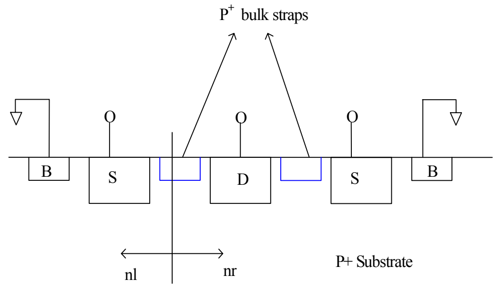


Figure 7. Simulated structure to study the dependence of R_{sub} on multiple gate fingers.

Figure 7 shows the structure that is simulated with the 2D device simulator MEDICI [9] to study the dependence of the substrate resistance on the number of gate fingers. “nl” and “nr” denote the number of gate fingers to the left and right of the gate finger of interest, respectively (i.e., nl=1, nr=2, in Fig. 7).

Figure 8 shows the finger resistance with different nl and nr for a fixed device width of 500 µm. It can be seen that the finger resistance for a fixed nl varies linearly with nr and hence in theory only two points on one straight line are needed to obtain a linear model. It should be noted that only three points on the first two straight lines (A, B, and C as shown in Fig. 8) are required to obtain the finger resistance models for nl=2 and nl=3 since the finger resistance for (nl,nr)=(2,3) is the same as that for (nl,nr)=(3,2). Noting the linear dependence for 2≤nl≤(n-1) and that the finger resistances for (nl,2)=(2,nl) and (nl,3)=(3,nl), the finger resistance for any finger with 2≤nl≤(n-1) can be written as below:

$$\frac{R_{fing}(nl,nr)-R_{fing}(2,nl)}{nr-2}=\frac{R_{fing}(3,nl)-R_{fing}(2,nl)}{3-2} \qquad (1)$$

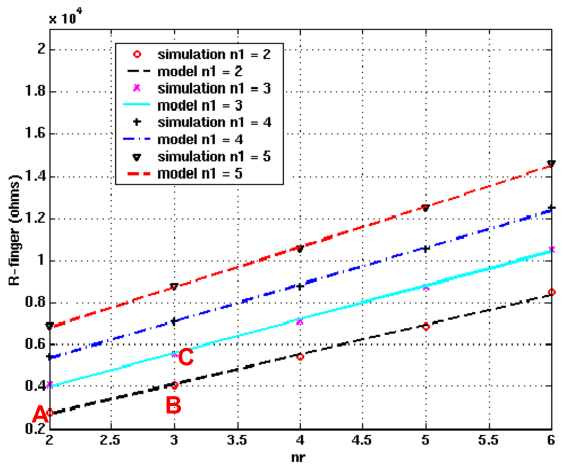


Figure 8. R_{sub} vs. nr for different nl.

Since the finger resistances for the two end fingers dominate the substrate resistance [4], they have been separately modeled using a linear model for accuracy. By obtaining the six finger resistance models for {(nl,nr)}={(1,1), (1,2), (1,3), (2,2), (2,3), (3,3)}, all the finger resistance models can be obtained from Eq. (1). The total substrate resistance R_{sub} for a fixed distance *d* from the bulk contacts to the active region is then calculated using Eq. (2) after all finger resistances have been calculated. In Eq. (2), *W* is the device width and it is seen that R_{sub} for parallel bulk contacts is inversely proportional to *W*.

$$R_{sub}(n,d)=\frac{500\mu}{W}\frac{1}{\sum_{nl=1}^{n}\frac{1}{R_{fing}(nl,nr)}} \qquad (2)$$

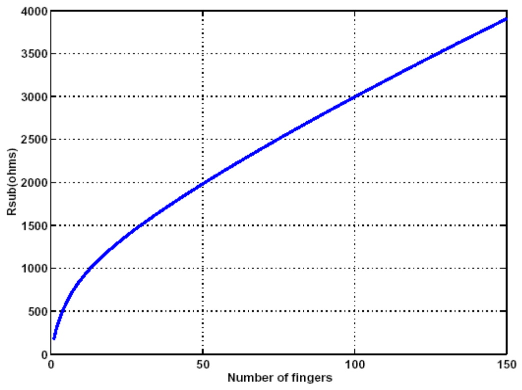


Figure 9. R_{sub} vs. number of fingers for W=500µm.

Figure 9 shows a plot of the R_{sub} variation with the number of gate fingers for a constant device width of 500 µm. It can be observed that R_{sub} increases with *n* and for large *n* has a linear dependence on *n*. This can be explained by observing that for a larger number of gate fingers, the resistance contribution of the interior fingers to the total substrate resistance is insignificant. As the device width is kept constant, the width of each finger scales inversely with the number of fingers. For large *n*, the substrate resistance that is dominated by the end fingers scales inversely with the finger width as *n* increases. Hence, an inverse dependence on the finger width is observed, that translates directly to a linear dependence on *n*.

Figure 10 shows the dependence of R_{sub} on the distance of the bulk contacts from the active region. The simulation is performed for structures with different number of fingers with fixed finger width as the distance *d* is varied from 2 µm to 15 µm. R_{sub} has a linear dependence on *d* for multi-finger transistors with parallel bulk contacts.

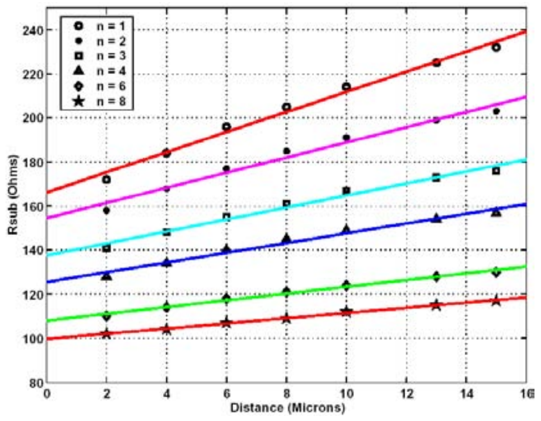


Figure 10. R_{sub} vs. *d* for different number of fingers.

A scalable model for R_{sub} as a function of *n* and *d* can be obtained by combining the previous results of this section as below:

$$R_{sub}(n,d) = \left[\frac{R_{sub}(n,d=d_2) - R_{sub}(n,d=d_1)}{d_2 - d_1}\right] * d + \left[\frac{d_2 * R_{sub}(n,d=d_1) - d_1 * R_{sub}(n,d=d_2)}{d_2 - d_1}\right] \quad (3)$$

Table I shows the values of R_{sub} obtained from device simulations for a MOS transistor with a width of 500 μm and different number of gate fingers for various channel lengths. For the typical channel lengths used in RF applications, R_{sub} has a weak dependence on the channel length. This has also been observed from measurements on a single fingered device [13].

Table I. R_{sub} vs. channel length for different number of fingers.

N / L (μm)	R_{sub} (Ohms)		
	1	4	10
0.6	165	515	950
0.7	163	508	940
0.8	160	502	933
0.9	158	495	925
1.0	155	490	915
2.0	143	465	870

Therefore, Eqs. (1), (2) and (3) describe the scalable substrate model for a multi-finger transistor with parallel bulk contacts as a function of the number of fingers (*n*), device width (*W*), and bulk contact locations (*d*). The dependence on channel length (*L*) is weak and is not included in the model.

4.2 Substrate model for perpendicular bulk contacts

So far, we have been considering the case where the bulk contacts are placed parallel to the finger direction as shown in Fig. 2(b). However, in certain applications the substrate contacts are placed perpendicular to the finger direction. The structure similar to Fig. 2(a) with two perpendicular bulk contacts is simulated using the 3D device simulator PROPHET [3] for a different number of fingers and the extracted values of the substrate resistance are shown in Table II for a finger width of 10 μm.

It can be seen from Table II that R_{sub} scales inversely with the number of fingers and the error incurred on assuming this scaling is less than 5%. Thus, when the substrate contacts are placed perpendicular to the device active area, R_{sub} is inversely proportional to the number of fingers for a given finger width.

Table II. R_{sub} vs. number of fingers for fixed finger width (10 μm).

n	R_{sub} from PROPHET (Ohms)	R_{sub} scaled from single finger value (Ohms)	% error
1	3400	3400	0
2	1805	1700	5.8
3	1210	1133	6.4
4	893	850	4.8

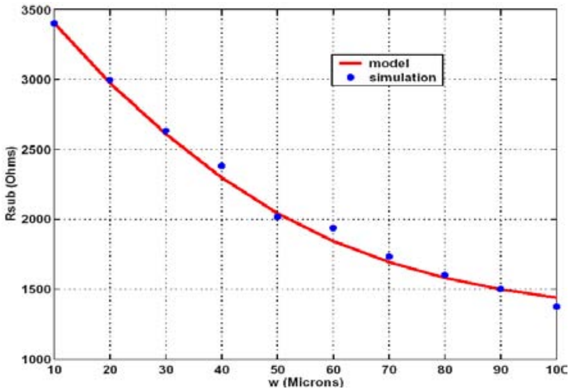


Figure 11. R_{sub} vs. finger width.

Figure 11 shows the R_{sub} variation with the finger width. It can be seen that R_{sub} decreases with an increased finger width and eventually saturates. The equation describing the dependence of R_{sub} on the finger width (W_f) and the finger number (*n*) is given by:

$$R_{sub} = \frac{1}{n} * \left(\frac{1}{\alpha * W_f^{\,3} - \beta * W_f^{\,2} + \gamma * W_f + \delta} + 1225 \right) \quad (4)$$

where the four curve fitting parameters are: α=3.655e-9, β=2.905e-8, γ=0.995e-5, and δ=3.63e-4.

Figure 12 shows a plot of R_{sub} vs. distance *d* for a single finger transistor with two perpendicular bulk contacts. It can be seen that R_{sub} has a linear dependence on *d*. Therefore, an equation similar to Eq. (3) can be derived for a scalable layout dependent substrate model for multi-finger transistors with perpendicular bulk contacts.

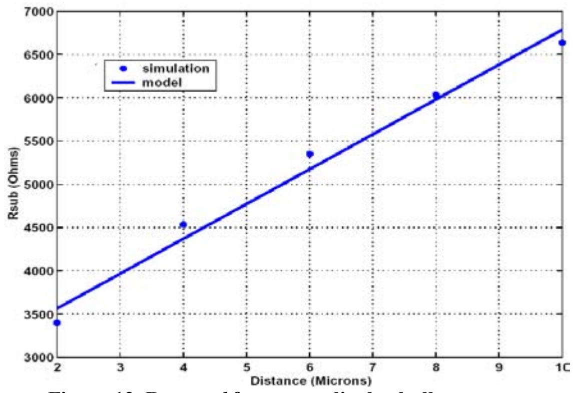


Figure 12. R_{sub} vs. *d* for perpendicular bulk contacts.

5. Experimental results

5.1 Measurement data of substrate resistance

Various test structures were fabricated in the 0.35 μm TSMC process to study the dependence of the substrate resistance on the layout of the MOS transistors. Figure 13 shows the layout of the test structures excluding the probe pads. The test structures can be divided into six different categories with each category serving a different purpose as shown in Fig. 13. Table IV describes these categories.

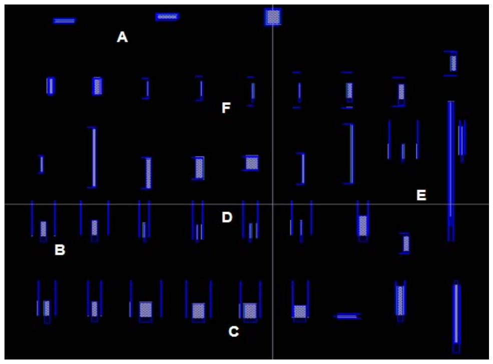


Figure 13. Test structures to study the layout dependent R_{sub}.

Table IV. Description of test structures

Label	Objective	Number of test structures
A	R_{sub} with contacts all around the device	5
B	R_{sub} dependence on distance for 4 fingers	4
C	R_{sub} dependence on distance for 10 fingers	4
D	R_{sub} dependence on distance for 1 finger	4
E	R_{sub} width dependence	7
F	R_{sub} with contacts perpendicular to the device	15

Table V shows the measured values of DC R_{sub} for a single fingered device for three different values of the finger width. The simulation results using the Substrate Coupling Analysis (SCA) tool [14] are also shown. The values of R_{sub} from SCA are in the ballpark of values obtained from measurements. It can be observed that R_{sub} scales inversely with the width of the device.

Table V. Variation of R_{sub} with finger width for a single finger device.

Width (μm)	R_{sub} (Ohms)		R_{sub} scaled from the measured value for W= 25 μm	% error
	SCA	Measurement		
25	165	202	202	0
50	90	106	101	4.7
200	23	28	25.3	9.2

Figure 14 shows a plot of the substrate resistance with the number of fingers for a device width of 200 μm with parallel bulk contacts. It can be seen that R_{sub} has a linear dependence on n for n > 2 and the linear model matches closely with the measured data. This justifies the simulated results observed in Fig. 9 of Section 4. Figure 15 shows the plot of R_{sub} with distance of the bulk contacts for devices with 1, 4 and 10 fingers, respectively. The finger width is 25 μm. Consistent with simulations in Fig. 10 of Section 4, the measured values of R_{sub} show a linear dependence on *d*.

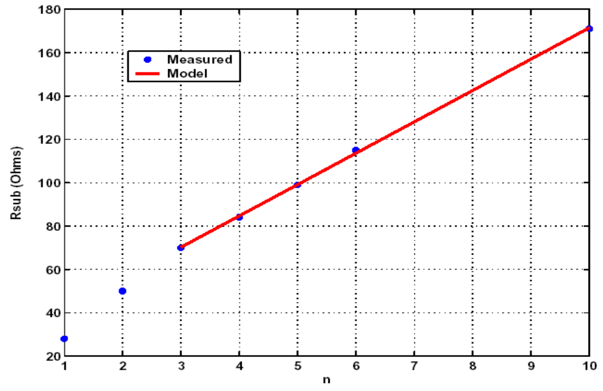


Figure 14. R_{sub} vs. number of fingers for W=200 μm.

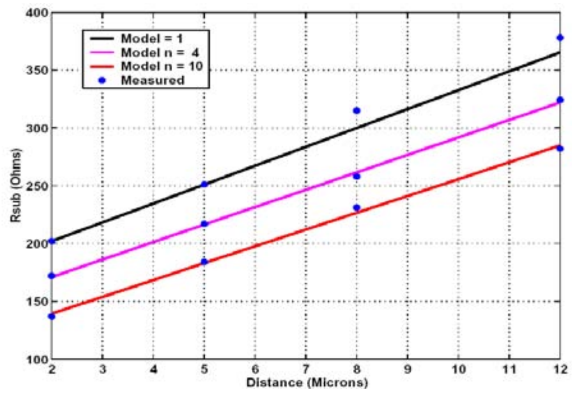


Figure 15. R_{sub} vs. *d* for a finger width of 25 μm.

5.2 Low noise amplifier

Figure 16 shows a 2.4 GHz common source low noise amplifier (LNA), in which the substrate resistances for multi-finger transistors (M1 and M2) are included to account for the silicon substrate. As discussed in Section 4, the value of the substrate resistance is dependent on the layout/bulk patterns and the geometric layout information. For this example, a standard multi-finger layout style with parallel bulk contacts is used. Both M1 and M2 are realized with 1-finger and 4-finger layout styles to show how the layout and, hence, R_{sub} affect the LNA performance. The substrate resistance model is the same as that discussed in Section 4.1. According to Fig. 9, for a transistor with a device size of 300μm, R_{sub}=267 Ω for a 1-finger layout style (R_{sub}= 500/300*160≈267 Ω), and R_{sub}=850 Ω for a 4-finger layout style (R_{sub}=500/300*510=850 Ω).

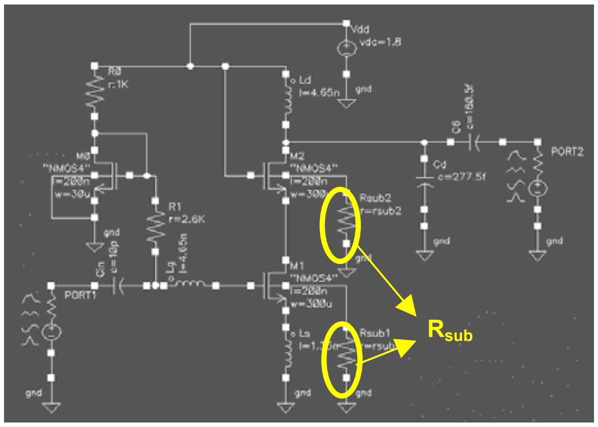


Figure 16. A low noise amplifier with substrate resistance.

Figure 17 and Figure 18 show the gain (S21) and noise figure (NF) of the LNA for both 1-finger and 4-finger layout styles. The LNA is first designed without considering the substrate resistance. The performance is then compared to an LNA in which the substrate resistance is included in the simulations. It can be seen that the substrate resistance has a significant influence on the LNA performance. Due to the substrate resistance, the gain is decreased and the noise figure is increased for these two layout styles. Since the substrate resistance is related to layout, a layout dependent substrate model is necessary for RF circuit design to properly account for the substrate effects. Although the substrate resistance is larger for a 4-finger layout style, the LNA performance degeneration is still less since parasitic capacitors (C_{db} and C_{sb}) for multi-finger transistors are decreased. On the other hand, it is obvious that the smaller the substrate resistance is the better is the agreement with a design without the substrate resistance.

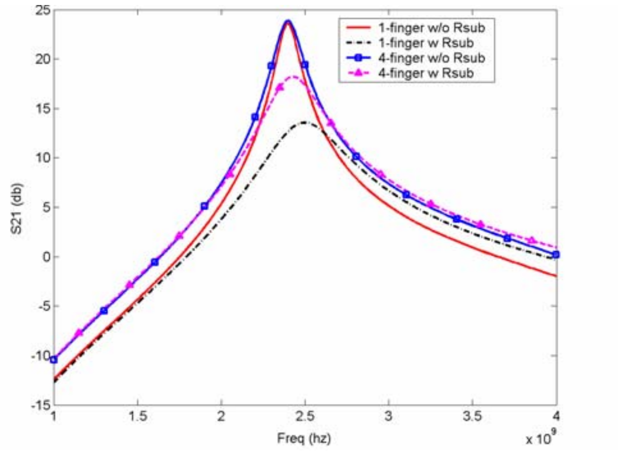


Figure 17. S21 for LNA with transistors in 1-finger and 4-finger layout styles.

For a standard multi-finger layout with parallel bulk contacts, the number of fingers has to be small to achieve a small substrate resistance as shown in Fig. 9. This will, unfortunately, introduce larger parasitic capacitance effects. Therefore, there exists a design tradeoff between the substrate resistance and parasitic capacitors. By using layout dependent substrate models, a RF circuit can be optimized for a proper layout/bulk pattern. In this manner, a good balance between the RF performance, parasitic effects, layout area, layout variation due to process variation, etc. can be achieved.

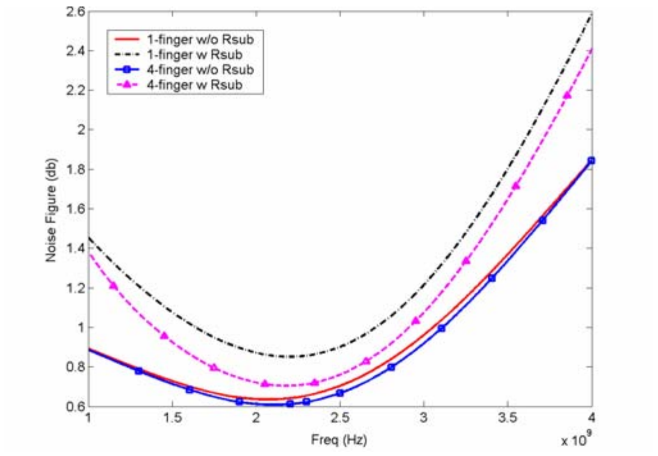


Figure 18. Noise figure for LNA with transistors in 1-finger and 4-finger layout styles.

6. Conclusions

This paper presents a layout dependent substrate model generation system *CrtSmile*. *CrtSmile* reads RF transistor layout files as the input and applies a pattern based layout extraction program to extract different kinds of layout/bulk patterns and geometric layout information. A scalable substrate model for multi-finger transistors is developed, which is dependent on the RF transistor layout/bulk patterns and geometric layout information, such as the number of fingers, finger width, channel length, and bulk contact locations. This model gives good agreement with the measured data for a 0.35μm CMOS process. A low noise amplifier example is further studied with the new layout dependent substrate model and shows the importance of CMOS RF transistor layout on substrate resistance modeling.

7. Acknowledgements

This work was supported by the Semiconductor Research Corporation under contract 2000-NJ-827.

8. References

[1] S. F. Tin, A. A. Osman, K. Mayaram, and C. Hu, “A simple subcircuit extension of the BSIM3v3 model for CMOS RF design,” *IEEE Journal of Solid-State Circuits*, vol. 35, no. 4, pp. 612-624, April 2000.

[2] http://www-device.eecs.berkeley.edu/~bsim3/bsim4.html

[3] http://www-tcad.stanford.edu/~prophet/

[4] Y. Cheng and M. Matloubian, “Parameter extraction of accurate and scalable substrate resistance components in RF MOSFETs,” *IEEE Electron Device Letters*, vol. 23, no. 4, pp. 221-223, April 2002.

[5] C. Enz and Y. Cheng, “MOS transistor modeling for RF IC design,” *IEEE Journal of Solid-State Circuits*, vol. 35, no. 2, pp. 186-201, Feb. 2000.

[6] R. Fujimoto, K. Kojima, and S. Otaka, “A 7-Ghz 1.8-db NF CMOS low-noise amplifier,” *IEEE Journal of Solid-State Circuits*, vol. 37, no. 7, pp. 852-856, July 2002.

[7] T. E. Kolding, “Test structure for universal estimation of MOSFET substrate effects at gigahertz frequencies,” *Proceedings of ICMTS*, pp. 106-111, March 2000.

[8] W. Liu, R. Gharpurey, M. C. Chang, U. Erdogan, R. Aggarwal, and J. P. Mattia, “R.F. MOSFET modeling accounting for distributed substrate and channel resistance with emphasis on the BSIM3v3 SPICE model,” *Dig. Tech. Papers IEDM-97*, pp. 309-312, Dec. 1997.

[9] MEDICI, Version 2000.2.0, Avanti! Corporation, 2000.

[10] J. J. Ou, X. Jin, I. Ma, C. Hu, and P. Gray, “CMOS RF modeling for Ghz communication IC’s,” *VLSI Technology Symp.*, pp. 94-95, June 1998.

[11] R. Suravarapu, K. Mayaram, and C.-J. Richard Shi, “A layout dependent and bias independent scalable substrate model for CMOS RF transistors,” *IEEE Radio and Wireless Conference*, pp. 217-220, Aug. 2002.

[12] A. Samavedam, A. Sadate, K. Mayaram, and T. S. Fiez, “A scalable substrate noise coupling model for design of mixed-signal IC’s,” *IEEE Journal of Solid-State Circuits*, vol. 35, no. 6, pp. 895-904, June 2000.

[13] L. F. Tiemeijer and D. B. M. Klaassen, “Geometry scaling of the substrate loss of RF MOSFETs,” *ESSDERC 1998*, pp. 480-483.

[14] Affirma Substrate Coupling Analysis Version 4.4.5, Cadence Design Systems, Inc., December 1999.

Correct-by-Construction Layout-Centric Retargeting of Large Analog Designs[*]

Sambuddha Bhattacharya, Nuttorn Jangkrajarng, Roy Hartono and C-J. Richard Shi
Department of Electrical Engineering, University of Washington
Seattle, WA, 98195-2500
{sbb,njangkra,rhartono,cjshi}@ee.washington.edu

ABSTRACT

Aggressive design cycles in the semiconductor industry demand a design-reuse principle for analog circuits. The strong impact of layout intricacies on analog circuit performance necessitates design reuse with special focus on layout aspects. This paper presents a computer-aided design tool and the methodology for a layout-centric reuse of large analog intellectual-property blocks. From an existing layout representation, an analog circuit is retargeted to different processes and performances; the corresponding correct-by-construction layouts are generated automatically and have performances comparable to manually crafted layouts. The tool and the methodology are validated on large analog intellectual-property blocks. While manual re-design and re-layout is known to take weeks to months, our reuse tool-suite achieves comparable performance in hours.

Categories and Subject Descriptors

B.7.2 [**Integrated Circuits**]: Design Aids– *layout*, J.6 [**Computer Applications**] Computer-Aided Engineering – *computer-aided design*.

General Terms: Algorithms, Performance, Design.

Keywords: Analog Integrated Circuit Design, Analog Layout Automation, Layout Symmetry, Analog Synthesis and Optimization.

1. INTRODUCTION

The integration of digital and analog circuits on *system-on-chips* has revolutionized the semiconductor industry and, yet, has given rise to an escalating complexity. Furthermore, driven by the need for superior performance and lower power consumption, the semiconductor industry continues to innovate technologies towards shrinking *transistor feature sizes*. These, together with the added pressure of aggressive design cycles, necessitate the adoption of the design reuse philosophy.

Continued advances in the CAD tools and the cell-based design methodology have largely enabled reuse of digital designs. In analog design, significant trade-offs between the major design goals like gain, bandwidth, stability, noise, linearity and power minimization demand considerable amount of time and effort of the designer.

[*] This research has been supported in part by the U.S. DARPA's NeoCAD program and in part by the NSF's ITR program.

Fortunately, significant progress has been made recently in the form of optimization tools [1][2] that automatically synthesize analog circuits meeting desired performance specifications. However, the performance of analog designs is not just affected by the device sizes and biasing, but also by the layout styles and intricacies.

Process and temperature gradients introduce mismatches in transistors that are designed to behave identically [3]. Such mismatches drastically affect analog circuit performance leading to DC offsets, finite even-order distortion and lower common-mode rejection [4]. Symmetric layout of matched transistors alleviates the effect of mismatch in analog circuits. Symmetry along with floorplanning, placement and parasitics are of immense importance in analog layout due to their strong impact on design performance. Often, layout designers use their years of accumulated expertise to "squeeze-in" the desired analog circuit performance by careful manual crafting of layouts.

These complexities pose a huge challenge to the automation of analog layouts[4][5]. Over the years, macro-cell based automated placement and routing methodologies have been proposed for analog circuits [6][7]. Unfortunately, these schemes, despite their generality, fail to incorporate the expertise of the layout designer and are seldom accepted in the industry.

Clearly, a layout automation technique that reuses the designer's knowledge embedded in existing layouts promises to be a viable alternative, especially for retargeting layouts to different specifications and technologies. Recently, an analog layout retargeting methodology is proposed in [8] where an already fine-tuned layout is used to create a symbolic structural template incorporating floorplan, symmetry and device/wiring alignment information. This structural template is then used to automatically generate a new layout.

Unfortunately, the above scheme has several shortcomings. Firstly, the layout symmetry for matched transistors is manually imposed on the template and can get increasingly prohibitive as the circuit size increases. Secondly, the methodology does not provide a smooth integration between the circuit and layout design steps. More importantly, it does not have the ability to handle layouts more complex than operational amplifiers.

In this paper, we present a layout-retargeting tool capable of automating layouts of large analog intellectual-property (IP) blocks. We introduce several techniques in the IPRAIL (Intellectual Property Reuse-based Analog IC Layout) framework [8] that allow it to handle layouts of large analog circuits. Firstly, large analog circuits not only require symmetric layouts for matched transistors, but also for entire subcircuits that need to be identical to each other. Subcircuits may be split into halves and laid apart in one or two-dimensionally symmetric ways so as to ensure similar effects of process and temperature gradients on all subcircuits that are identical by design. IPRAIL addresses these issues with a novel multi-level templating scheme that reduces the template size. Secondly, large analog IP blocks almost always contain on-chip resistors and capacitors. Such passive devices are very sensitive and need to be laid out carefully to minimize the parasitic and coupling effects. Furthermore, passive devices identical by design are also laid out symmetrically. Layout retargeting has to be performed while maintaining such restrictions. Thirdly, user intervention at any step in the template creation or refinement restricts the usability of retargeting tools to smaller layouts. The tool presented in this paper achieves complete automation of the retargeting flow.

Given the strong dependence of circuit performance on layout, we present a scheme for design reuse with IPRAIL at its core. IPRAIL is combined with a commercial circuit optimization engine to formulate a viable correct-by-construction layout-centric design reuse methodology for analog circuits.

This paper is organized as follows. Section 2 discusses the layout-centric design reuse methodology. Section 3 describes the structure and flow of the IPRAIL tool-suite. Sections 4, 5, and 6 present the techniques for template size reduction. Section 7 discusses passive device retargeting. Section 8 presents the experimental results. Section 9 concludes the paper.

2. LAYOUT-CENTRIC DESIGN REUSE

The layout-centric design reuse methodology proposed in this paper is illustrated in Fig. 1. An existing high quality manually crafted analog layout is used as the starting point in the redesign of the circuit that targets a different specification and/or a different technology. This layout is read into IPRAIL that is at the core of this methodology.

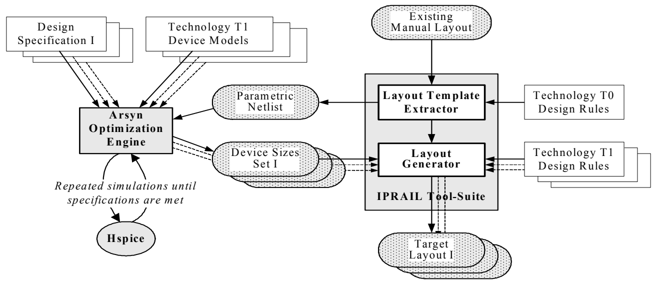

Design Specification I
Technology T1 Device Models
Existing Manual Layout
Parametric Netlist
Arsyn Optimization Engine
Layout Template Extractor
Technology T0 Design Rules
Device Sizes Set I
Layout Generator
Technology T1 Design Rules
Repeated simulations until specifications are met
IPRAIL Tool-Suite
Hspice
Target Layout I

Fig. 1: Reuse based design methodology for analog circuits.

The IPRAIL tool-suite comprises of a *layout template extractor* and a *layout generator*. The Layout template extractor creates a resizable symbolic template and extracts a parametric netlist. The parametric netlist, target design specifications and technology specific simulation models are fed into *Arsyn* [2], a commercially available simulation based circuit optimizer. Arsyn automatically synthesizes the analog circuit meeting the required design specifications. The device sizes generated by Arsyn are fed to the layout generator. The layout generator imposes the device sizes on the symbolic template and automatically constructs a new layout according to the target specifications and/or technology.

As the template extractor detects matched devices in the layout, this can be passed directly as constraints to Arsyn through the parametric netlist. The template also retains the mappings of the devices in the parametric netlist with the corresponding layout abstractions. This facilitates automated imposition of the device sizes obtained from Arsyn on the template prior to layout generation. Thus, the entire flow of data between the two tools requires minimal intervention from the designer.

As the intricacies in the existing design layout is automatically retained in the symbolic template, the layouts generated for different target specifications are automatically customized to the specific needs of the circuit under consideration. As shown in Fig. 1, multiple designs corresponding to different target specifications and technologies can be synthesized and laid out automatically. This reuse of the circuit and layout topologies by the automation tools significantly reduces the design cycle while involving very minimal designer intervention.

3. IPRAIL TOOL-SUITE

3.1 Symbolic Template Extractor

The template extractor creates the symbolic structural template based on the input layout, from which the layout topology, connectivity, and matching are acquired. These intellectual properties are maintained in the template and, along with the new devices sizes obtained from the optimizer, are later utilized as the basis for generating the retargeted layout. The preservation of the layout properties is achieved using sets of various constraint equations that arise from the physical form of the input layout.

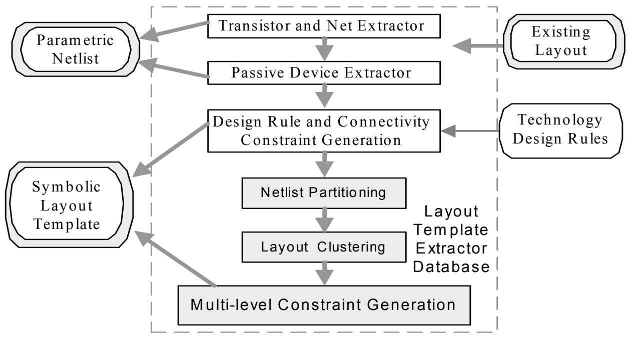

Parametric Netlist
Transistor and Net Extractor
Existing Layout
Passive Device Extractor
Design Rule and Connectivity Constraint Generation
Technology Design Rules
Symbolic Layout Template
Netlist Partitioning
Layout Clustering
Layout Template Extractor Database
Multi-level Constraint Generation

Fig. 2: Layout Symbolic Template Extractor flow.

The detailed tasks of the template extractor are depicted in Fig. 2. First, transistors, nets, and passive devices are extracted from the layout, and compiled to generate a parametric netlist for Arsyn. Next, the scan line [9] method is employed to establish constraints for the symbolic template based on connectivity, topology, and design rules.

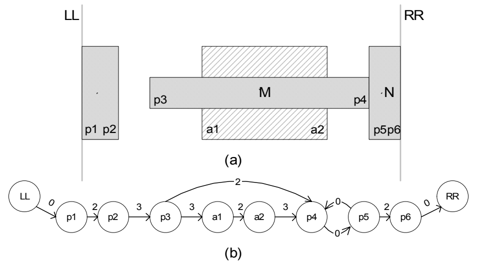


Fig. 3: A horizontal constraint graph as a symbolic template.

To enhance the computational speed of the ensuing layout generation process, the constraints equations, wherever possible, are converted into a constraint graph [9]. Each rectangle in the layout is transformed into four independent nodes, representing its left, right, top, and bottom edges. Constraints are placed between nodes in the graph to sustain layout integrity and correctness. Different constraint categories are connectivity, design rule, exact device size, and symmetry. Horizontal and vertical constraint graphs are constructed independently.

Consider the simple layout of Fig. 3, the connectivity between rectangles M and N in the horizontal direction is retained by two constraint arcs of weight '*0*' between edges *p4* and *p5*. The design rule constraint is further decomposed into three types: minimum width – an arc from *p1* to *p2*, minimum spacing – an arc from *p2* to *p3*, and minimum extension – an arc from *a2* to *p4*.

Matching between a pair of transistors is established by laying out the transistors symmetrically. Two transistor layouts are deemed symmetric if they are geometric mirror images of each other. As illustrated in simplified example of Fig. 4, this implies equi-sized channel, drain and source regions, identical orientation and close proximity of the two transistors. Identical channel, drain and source regions are implicitly enforced by the device sizes obtained from Arsyn. Mirroring and location are enforced by the following equations.

$$(e_{bottom} - f_{bottom}) = 0 \quad (1)$$

$$(s_0 - g_{right}) - (h_{left} - s_0) = 0 \quad (2)$$

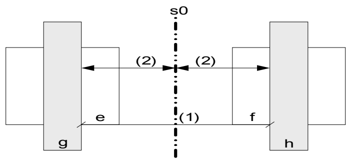


Fig. 4: Simplified layout of two transistors symmetric about axis 's_0'.

3.2 Layout Generator

The algorithm for generation of a new layout from the template is based on symbolic compaction [10]. This is accomplished in two steps: first, the new device sizes are imposed on the template. Then this updated template is solved by a combination of linear programming (LP) [11] and graph based longest-path algorithm [12]. The detailed steps for layout generation are shown in Fig. 5.

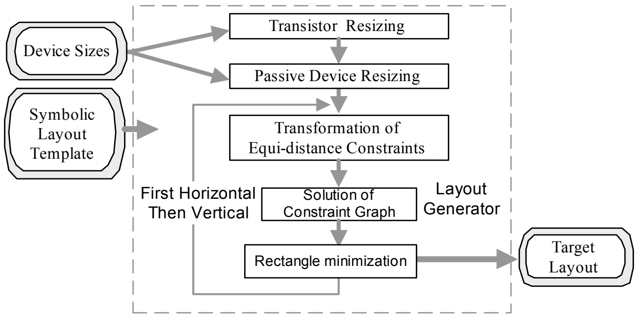


Fig. 5: The layout generator flow.

The exact device sizes obtained from Arsyn are imposed on the template by an additional pair of constraint arcs with equal and opposite weight are added in opposite directions. The template mentioned so far consists of a constraint graph and three variable equi-distance constraints of Eq. (2) imposed due to layout symmetry. In order to solve this modified compaction problem in its graph form (due to superior speed compared to LP), the symmetry constraints need to be first transformed before they can be imposed on the constraint graph. This necessitates multiple longest-path based transformations of the constraint graph into a smaller graph and subsequent LP runs [10]. Finally, the equi-distance constraints of Eq. (2) are transformed to the form

$$(s_0 - g_{right}) = (h_{left} - s_0) = b \quad (3)$$

where *b* is a constant obtained from LP. This is then imposed on the constraint graph and the entire problem is solved with the longest-path algorithm first in the horizontal direction and then in the vertical direction. Finally, as the longest-path algorithm results in some unwanted extension of rectangles, the rectangle minimization algorithm [13] is applied to obtain the final target layout. The graph transformation process for symmetry constraints is computationally intensive.

Therefore, reduction of such constraints is a one of the motivations of our approach.

4. LAYOUT-NETLIST MAPPING FLOW

The primary challenge in retargeting large analog circuits lies in creating a template consisting of minimum number of constraints. In addition to matched transistors, large analog circuits contain entire sub-blocks that are identical by design and demand special attention during layout. Consider a 2-bit comparator circuit that is composed of 4 unit comparators. Gradients in the process parameters across the entire layout introduce differences between the unit comparators that result in non-linearities. This is alleviated by laying out the unit comparators in a common-centroid fashion as illustrated in Fig. 6, where the unit comparator denoted by *A* is split across the ends into two parts A_1 and A_2.

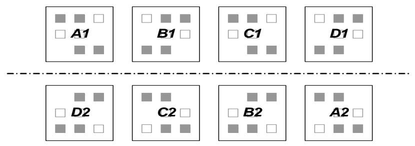


Fig. 6: Four analog subcircuits in two-dimensional symmetric layout.

In these cases, the layout comprises of several identical sub-blocks that are flipped or translated with respect to each other. A naive direct constraint generation for detecting all symmetric or translated devices and nets [14] leads to a tremendous increase in template size for large circuits. This paper addresses this difficulty in constraint generation by extensive partitioning of and mapping between the netlist and layout abstractions.

The layout-netlist mapping flow is shown in Fig. 7. The netlist extracted from the layout consists of *unit transistors*, i.e., transistors with only one rectangle each for its gate, drain and source nodes. These unit transistors are clustered into groups and a compact netlist is obtained by *proximity-based netlist clustering*.

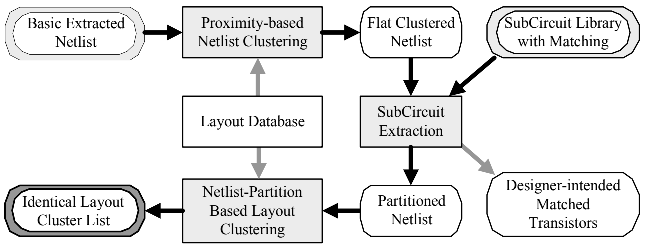

Basic Extracted Netlist
Proximity-based Netlist Clustering
Flat Clustered Netlist
SubCircuit Library with Matching
Layout Database
SubCircuit Extraction
Designer-intended Matched Transistors
Identical Layout Cluster List
Netlist-Partition Based Layout Clustering
Partitioned Netlist

Fig. 7: Multi-level Layout-Netlist mapping flow.

Often, analog design environments consist of a library of commonly used subcircuit topologies, such as differential pairs, current mirrors, comparators etc., that are extensively used in large designs. *Subcircuit extraction* identifies all instances of the subcircuits from the library in the clustered netlist.

The subcircuit mapping step creates a fully partitioned netlist. Based on this partitioned netlist, the corresponding clusters of rectangles in the layout are identified, thus resulting in a partitioned layout. Furthermore, since the library subcircuits contain information about designer-intended matched transistor pairs, subcircuit mapping also produces a full list of essential matched transistor pairs in the netlist [15].

The netlist and layout partitioning process also establishes mapping at different levels between the layout and the netlist. Actual constraint generation is then triggered from the list of matched transistors and the lists of layout clusters. For large analog circuits, such mapping and partitioning is essential to reduce the size of template to manageable levels.

The steps in the multi-level layout-netlist mapping flow are elaborated in Section 5. The constraint generation process is described in Section 6.

5. NETLIST AND LAYOUT PARTITIONING

5.1 Proximity Based Netlist Clustering

In the layout, each multi-fingered transistor M contains multiple contiguous elements C, where each contiguous element consists of physically contiguous unit transistors T. Fig. 8 shows two multi-fingered transistors in a *common-centroid* layout. Here, each multi-fingered transistor has two contiguous sets of three unit transistors each. The clustering scheme partitions the netlist based on the manner in which the transistors are laid out [15].

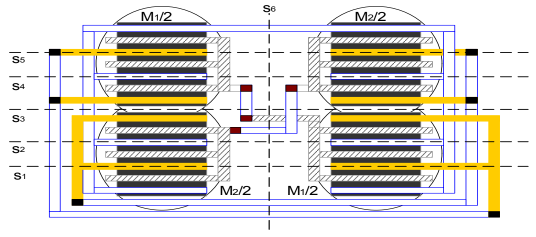


Fig. 8: A common-centroid layout of a symmetric transistor pair. Rectangles with crossed pattern represent the polysilicon layer.

The netlist, which at the end of extraction, comprised of the set of unit transistors T^S and the set of nets N^S, now consists of the same set of nets N^S and the set of multi-fingered transistors M^S defined as $\{M \mid \forall M, \exists$ a unique $\{G_M,S_M,D_M\} \subset N^S\}$ where $\{G_M,S_M,D_M\}$ is the set of the gate, source and drain nets of the multi-fingered transistor M. Each multi-fingered transistor M is a set of physically contiguous elements C^S i.e., $C \in M$. And each contiguous element is defined as $C = \{T \mid T \in T^S, \forall T\ \{G_T,S_T,D_T\} = \{G_M,S_M,D_M\}$, and $\forall T \in C$ are physically contiguous$\}$.

5.2 Subcircuit Extraction

A subgraph isomorphism algorithm [16] is adopted for subcircuit extraction from the clustered netlist. The extraction results in the netlist partitioning essential for layout clustering and also generates a list of designer-intended matched transistors [15].

First, an iterative labeling algorithm is used to partition both the subcircuit and the main circuit. This identifies a set of nodes in the main circuit and a single node, called a *key* node, in the subcircuit. The set of nodes in the main circuit obtained by this iterative labeling algorithm are potential start-points for checking a pattern match with the subcircuit. From each potential node in the main circuit and the key node in the subcircuit, another labeling algorithm detects isomorphism with the subcircuit graph.

5.3 Netlist-Partition Based Layout Clustering

After the subcircuit extraction creates several netlist partitions each corresponding to a subcircuit instance, regions in the layout belonging to the same netlist partition are clustered together. The algorithm for layout clustering is shown in Table 1.

The algorithm starts from a seed device, D_s, which belongs to a netlist partition N_s. First, a layout cluster, L_c, is created containing only D_s. All devices that are proximal to D_s in the layout are collected in a queue Q_s. This is accomplished by a scan line [9] based procedure *ProximalDevices*. From any device, four scan lines look for other devices in close proximity to its left, right, top and bottom edges. If a device D_p in Q_s, is in the netlist partition N_s, then D_p is added to the same layout cluster L_c that D_s belongs to. If D_p does not belong to N_s, the algorithm recursively calls itself to start a new layout cluster. The algorithm terminates when all devices in the layout have been grouped into layout clusters. In practice, a multi-fingered transistor may be composed of two or more contiguous elements that are laid out far apart to account for different process gradients. Thus, a layout cluster may not comprise all contiguous elements of a multi-fingered transistor.

Table 1: Layout clustering algorithm.

```
CreateLayoutClusters ($D_s$)
begin
  if ($D_s$ already assigned to a cluster)
    return
  endif
  $L_c$ = CreateCluster ($D_s$)
  $Q_s$ = ProximalDevices ($D_s$)        // Scan-line based routine
  foreach $D_p$ in $Q_s$
    If ($D_p$.partition == $D_s$.partition)
      AddCluster ($L_c$ , $D_p$)         // Add $D_p$ to $L_c$
    else
      CreateLayoutClusters ($D_p$)       // Recursively call with seed $D_p$
    endif
  end for
end
```

5.4 Detection of Identical Layout Clusters

The previous steps mark each device (or contiguous element of multi-fingered transistor) in the layout with its netlist level cluster, subcircuits and layout-cluster information. The layout clusters A and B are identical only when their devices are one-to-one matched in terms of device sizes and device locations. Consider two layout clusters, A and B, located on the same abscissa. First, all rectangles in each cluster are collected in heaps H_A and H_B. The rectangles are then sorted in an increasing order with respect to the coordinates of the leftmost corner. The two heaps are pair-wise compared [14]. If they are identical, A and B represent *translate-matched* clusters. In case they are not, another heap H_{B2} is created for cluster B where the rectangles are sorted in a decreasing order of their rightmost corner. If the two heaps H_A and H_{B2} are found to be identical upon pair-wise comparison, then they represent *flip-matched* clusters.

6 MULTI-LEVEL CONSTRAINT GENERATION

6.1 Intra-Cluster Symmetry Constraints

From the partitioned netlist generated through the subcircuit extraction process, a list of designer-intended matched devices is obtained [15]. The layout rectangles of such matched multi-fingered transistors within a layout-cluster are collected in sorted lists. For the common-centroid topology in Fig. 8, the six unit transistors on left are collected into a list L_L, and right into a list L_R. The unit transistors in L_L and L_R are then pair-wise compared to detect the vertical axis of symmetry, s_6, and generate the corresponding constraints. For the horizontal symmetry axis s_3, the bottom halves of both M_1 and M_2 are collected in a list L_B, and the top halves are collected in a list L_T and pair-wise compared.

6.2 Inter-Cluster Constraints

Identical and symmetric clusters are laid out either mirror-wise flipped or translated with respect to each other horizontally or vertically. Fig. 9 shows two translated and flipped identical clusters. To ensure matching at the cluster-level, in addition to the intra-cluster constraints, a minimal number of constraints are imposed between the two clusters. First, two devices that represent the same subcircuit elements on different layout-clusters are chosen as the reference devices of the respective clusters. By imposing constraints between two such reference devices, inter-cluster translation or flipped matching is ensured.

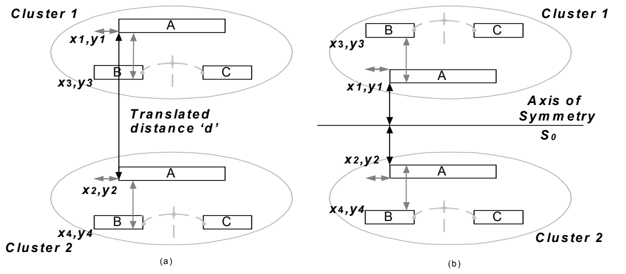


Fig. 9: Inter-cluster matching. (a) Translated matched clusters. (b) Flipped matched clusters. Dotted lines show intra-cluster symmetry.

For the translated clusters in Fig. 9a, the following two equations between two cluster's reference devices *1A* and *2A* are necessary for cluster matching.

$$y_1 - y_2 \geq d \quad (5)$$
$$x_1 = x_2 \quad (6)$$

where *d* is the distance between two reference devices and is determined by coupling or design rule constraints. In addition, an equation relating each symmetric group inside each cluster with the reference device of the cluster is included to ensure cluster matching. The following equations relate devices *1B* and *2B* with the reference devices *1A* and *2A*.

$$x_1 - x_3 = x_2 - x_4 \quad (7)$$
$$y_1 - y_3 = y_2 - y_4 \quad (8)$$

Similarly, for the flipped clusters in Fig. 9b, Eq. (5) is replaced with the following equation

$$y_1 - s_0 = s_0 - y_2 \quad (9)$$

where s_0 is the symmetry axis between two devices. The other constraints are similar to the translated case.

Secured by both intra-cluster and inter-cluster symmetries, if a layout consists of *c* clusters with *n* devices in each cluster, the number of symmetry axis is reduced from worst-case of *(nc)(nc-1)/2* to *(nc-1)*. And if each cluster consists of *g* groups of intra-cluster symmetric devices, where *n > g*, the total number of symmetry constraints will reduce from worst-case of *(nc)(nc-1)/2* to $n^{1/2}c(c-1)$, or by a factor of $n^{3/2}/2$, when $c >> 1$.

7. PASSIVE DEVICE RETARGETING

Layouts of large analog circuits consist of multiple on-chip passive devices that are usually isolated in space from other devices and nets to minimize the parasitic and coupling effects [4]. Passive devices like resistors and capacitors are automatically detected by a traversal through the nets in the layout when the calculated value of the passive exceeds a threshold. Connectivity between a passive device and the nets at its ends are maintained on *port* rectangles. To prevent overlaps or close proximity to other devices upon retargeting, a *shadow rectangle* is placed on top of the passive devices prior to the symbolic template generation, as shown in shaded area in Fig. 10. A shadow rectangle is a temporary non physical layer rectangle that is used to allocate a dedicated area for the passive device, by applying spacing constraints to rectangles on every layer. The constraints come from one of the following: coupling constraints, specialized design rules for passives as seen in certain technologies, or designer's input. Such constraints have the form:

$$X_{shadow} - x_{other} \geq d \quad (10)$$

Constraints are added between the shadow rectangle, the ports and the device to maintain connectivity upon retargeting.

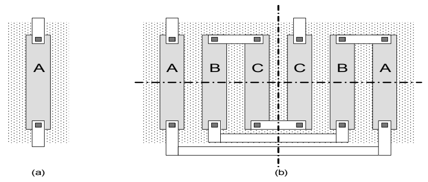


Fig. 10: (a) A unit resistor (b) One-dimensional interdigitated symmetric layout of resistors. In both cases, the shaded background represents the *shadow rectangle* used in retargeting.

Frequently, multiple passive devices are designed to be identical to each other. Fig. 10b shows three resistors laid out in an interdigitated fashion with one-dimensional common-centroid symmetry [5]. This layout ensures matched resistance between resistors *A*, *B*, and *C*, regardless of process gradients. The template extractor automatically detects such symmetry in passive devices and imposes appropriate symmetry constraints in a manner similar to intra-cluster symmetry for transistors. Prior to the layout generation step, new passive device sizes obtained from Arsyn are imposed as fixed width constraints. The shadow rectangle is appropriately extended and the entire set of constraints passed on to the layout generation engine.

8. RESULTS

We applied IPRAIL and our layout-centric design methodology to the design and layout of a *5-bit flash analog-to-digital converter* (ADC). The ADC comprised of a resistor chain, an analog section, and a digital block. The analog section consisted of 32 latched comparators. Each comparator, whose schematic is shown in Fig. 11, compares the input signal with different reference voltage levels obtained from resistor chain acting as a voltage divider. Based on the input voltage, the analog section produces a *thermometer code*. This is then converted to 5-bit binary output by the 32-to-5 decoder.

The analog section of the ADC was initially designed and laid out manually in 0.25um TSMC technology process, while the digital section was designed in a standard-cell based ASIC flow. To minimize mismatch between comparators due to process gradients, they are laid out in a one-dimensional common-centroid fashion. The resistor chain is constructed in the polysilicon layer and laid out in a fashion similar to Fig. 10b.

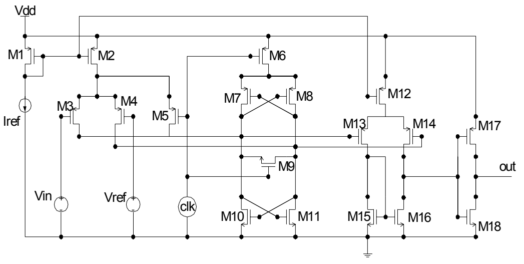


Fig. 11: Comparator schematic. M_3 and M_4 form the input diff-pair.

This ADC design in TMSC 0.25um technology process is then retargeted to TSMC 0.18um technology process with different specifications. The structural symbolic template for the layout was constructed by incorporating the scan line method and inter/intra cluster symmetry. The template, in a graph constraint form, consisted of 29,918 nodes, 1,113,599 arcs, and 2,574 symmetry related arcs. The symmetry related arcs were extracted from 192 intra-cluster symmetric transistors, 31 inter-cluster symmetries, and 63 resistor symmetries. In comparison, a direct extraction [14] results in 43,935 symmetry related constraint arcs.

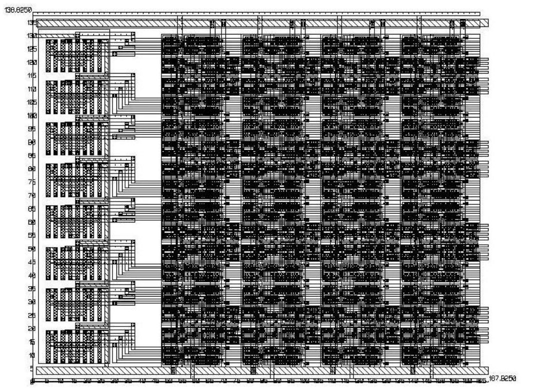


Fig. 12: Analog section of a 5-bit ADC in TSMC 0.25um technology.

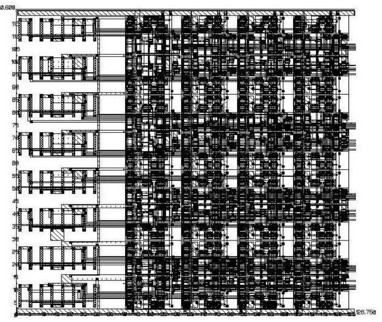


Fig. 13: Analog section of a retargeted ADC in TSMC 0.18um.

Table 2: ADC post layout specifications in 0.25um and 0.18um TSMC.

Performance Parameters	Original (0.25um)	Target (0.18um)
Supply Voltage	*2.50 V*	*1.80 V*
Reference Voltage	*1.28 V*	*1.28 V*
½ LSB Resolution	*20 mV*	*20 mV*
Sampling Rate	*500 MHz*	*750 MHz*
Diff. Nonlinearity (max)	*0.12 LSB*	*0.11 LSB*
Int. Nonlinearity (max)	*0.66 LSB*	*0.39 LSB*
Power Consumption	*42 mW*	*18 mW*
Total Area	*79,800 um^2*	*36,650 um^2*

The extracted parametric netlist was passed to Arsyn and the circuit synthesized in TSMC 0.18um to achieve the desired specifications. The new device sizes obtained from Arsyn were enforced on the template. The target layout was obtained by solving the template by a combination of LP and graph based longest-path algorithm. The analog section of the original layout is shown in Fig. 12 and the target layout is in Fig. 13.

The specifications achieved in the post layout simulation for both designs (0.18um and 0.25um) are listed in Table 2. IPRAIL was run on a 900MHz SUN UltraSparc3 workstation. The template extraction phase took 8 hours 52 minutes, and the generation of the target layout was completed in 1 hour 51 minutes.

9. CONCLUSIONS

A symbolic structural template based layout-retargeting tool capable of handling the complexities associated with large analog designs is presented. The fully automated IPRAIL tool combines several key techniques like netlist partitioning, layout clustering, automatic symmetry detection, and passive device matching in a novel way as a part of the template extraction. Multiple high quality layouts can be automatically generated from the symbolic template. As the target layouts are generated from a high quality manual layout, they are correct by construction. More importantly, the target layouts retain all complex layout techniques employed to alleviate the process gradients and other factors that affect circuit performance. This layout-retargeting tool is successfully combined with a commercial circuit synthesis tool in a layout-centric design reuse methodology. The methodology is shown to be effective in retargeting existing large analog designs to multiple specifications across different technologies. Large analog circuits that are known to take several weeks to months to synthesize and lay out are automatically created within hours with comparable performance.

REFERENCES

[1] M. J. Krasnicki, R. Phelps, J. R. Hellums, M. McClung, R. A. Rutenbar and L. R. Carley, "ASF: A practical simulation-based methodology for the synthesis of custom analog circuits", *Proc. Int. Conference Computer-Aided-Design*, Nov. 2001, pp. 350-357.

[2] Arsyn User's Manual, Orora Design Technologies Inc., 2003.

[3] M. J. M. Pelgrom, A. C. J. Duinmaijer and A. P. G. Welbers, "Matching properties of MOS transistors", *IEEE J. Solid State Circuits*, vol. 24, pp. 1433-1440, Oct. 1989.

[4] B. Razavi, *Design of Analog CMOS Integrated Circuits*, McGraw Hill, 2001.

[5] A. Hastings, *The Art of Analog Layout,* Prentice Hall, 2001.

[6] J. M. Cohn, D.J. Garrod, R. A. Rutenbar and L. R. Carley, "KOAN/ANAGRAM II: New tools for device-level analog placement and routing", *IEEE J. Solid State Circuits*, vol. 26, pp. 330-342, Mar. 1991.

[7] K. Lampaert, G. Gielen and W. M. Sansen, "A performance-driven placement tool for analog integrated circuits", *IEEE J. Solid State Circuits*, vol. 30, pp. 773-780, Jul. 1995.

[8] N. Jangkrajarng, S. Bhattacharya, R. Hartono, and R. Shi, "IPRAIL – Intellectual property reuse-based analog IC layout automation", *Integration – The VLSI Journal*, vol. 36, pp. 237-262, Nov. 2003.

[9] S. L. Lin and J. Allen, "Minplex – a compactor that minimizes the bounding rectangle and individual rectangles in a layout", *Proc. IEEE/ACM Design Automation Conference*, Jun. 1986, pp. 123-130.

[10] R. Okuda, T. Sato, H. Onodera and K. Tamaru, "An efficient algorithm for layout compaction problem with symmetry constraints", *Proc. Int. Conference Computer-Aided-Design*, Nov. 1989, pp. 148-151.

[11] D. Luenberger, *Linear and Nonlinear Programming* 2nd Edition, Addison-Wesley, 1984.

[12] T. H. Cormen, C. E., Leiserson and R. L. Rivest, *Introduction to Algorithms*, MIT Press, 2001.

[13] G. Lakhani and R. Varadarajan, "A wire-length minimization algorithm for circuit layout compaction", *Proc. Int. Symposium on Circuits and Systems*, May 1987, pp. 276-279.

[14] Y. Bourai and C. J. R. Shi, "Symmetry detection for automatic analog layout recycling", *Proc. Asia and South Pacific Design Automation Conference,* Jan. 1999, pp. 5-8.

[15] S. Bhattacharya, N. Jangkrajarng, R. Hartono, and C-J. R. Shi, "Hierarchical extraction and verification of symmetry constraints for analog layout automation", *Proc. Asia and South-Pacific Design Automation Conference*, Jan 2004, pp. 400-405.

[16] M. Ohlrich, C. Ebeling, E. Ginting and L. Sather, "Subgemini: Identifying subcircuits using a fast subgraph isomorphism algorithm", *Proc. IEEE/ACM Design Automation Conference,* Jun. 1993, pp. 31-37.

Challenges and Techniques for Layout Automation of Radio-Frequency Integrated Circuits

Sambuddha Bhattacharya, Nuttorn Jangkrajarng, Roy Hartono and C-J. Richard Shi
Mixed-Signal CAD Research Laboratory,
Department of Electrical Engineering, University of Washington
Seattle, WA 98195, USA

ABSTRACT

Radio-frequency (RF) integrated circuits are notoriously difficult to lay out because of significant couplings between layout geometries and strong impact of substrate parasitics. Achieving the desired electrical performance of RF integrated circuits necessitates intricate floorplanning of active and passive devices, careful control of the relative placement of devices and wires, and placement of bulk contacts to alleviate the lossy behavior of substrates. This paper discusses the various challenges and different approaches for layout automation of RF integrated circuits.

1. INTRODUCTION

As wireless standards evolve to higher frequencies and market expands rapidly, the radio-frequency (RF) circuit design is heading towards full on-chip integration. With increasing demand for portability and affordability, all of digital, analog and RF circuits are converging to single chip solutions. Unlike digital circuits, the analog and RF circuit design and layout requires multiple iterations due to the inherent complexity and precision requirements. While computer-aided design (CAD) tools for the digital domain have long attained maturity, substantial innovations are necessary for industrial acceptability of such tools for analog and RF circuits. In this context, automation of RF layout is particularly difficult due to the strong impact of layout geometry on circuit performance.

Traditionally, CAD tools for RF layout generation comprised interactive engines for aiding manual design iterations [1][2]. In the past decade, several attempts have been made to address RF layout automation. These can be classified broadly into two areas: Stochastic optimization based place and route systems, and template-based layout automation tools. This paper reviews these schemes and points out their advantages and shortcomings.

The paper is organized as follows. Section 2 discusses the challenges in layout automation for RF circuits. Section 3 outlines the stochastic optimization based place and route schemes. Section 4 describes template-based layout automation. Section 5 compares the two techniques and Section 6 concludes the paper.

2. CHALLENGES IN RF LAYOUT AUTOMATION

Device Matching and Symmetry: Transistors designed to behave identically may exhibit finite mismatch due to asymmetry in their layout structures or locations [3]. Transistor mismatch can drastically affect circuit performance leading to DC offsets, finite even-order distortion and lower common-mode rejection. For large or stacked transistors commonly seen in RF circuits, layouts drawn by maintaining simple geometric mirroring (symmetry) may not establish acceptable matching due to spatial variations in process parameters like oxide thickness, mobility etc. In such cases, *common-centroid* configurations are often employed to cancel out the mismatches introduced due to process gradients. Layout automation engines need to ensure layout symmetry between matched transistors.

Device Floorplanning: Device Floorplan strongly affect the parasitic capacitances of MOS transistors. RF circuits commonly employ large transistors that need to be laid out in a multi-fingered fashion to minimize drain and source capacitances [4]. In addition, careful layout is necessary to reduce the parasitic gate resistance and capacitance [5].

Passive Device Integration: Optimal layout of on-chip spiral inductors requires significant time and effort due to multiple iterations of layout modification and extensive 3-D electromagnetic computations. The parasitic resistance and capacitance associated with on-chip spirals, the electric coupling between spirals and the substrate and the magnetically induced eddy currents flowing into the substrate greatly complicate various on-chip inductor performance specifications like quality factor (Q), self-resonant frequency and inductance value. All these effects greatly complicate the automatic generation of optimal on-chip spiral inductor layout geometry and topology.

Relative Placement: Pronounced electromagnetic interference may result due to relative position of different devices. These effects are difficult to model in simple abstractions necessary for layout automation engines.

Routing: Wire length, bends and proximity to other signals are of immense importance in ensuring desired RF integrated circuit performance. Traditional approaches for ensuring desired performance involve quasi-static or full-wave electromagnetic simulations of layout patterns and subsequent modifications by experienced layout designers. Distributed parasitic models need to be evaluated in the core of layout automation engines for RF integrated circuits.

Substrate Parasitics: Digital circuitry in RF systems-on-chips can inject noise into the substrate that can couple with sensitive RF circuits thereby creating havoc with their functionality. These sensitive cells are isolated from the rest of the design by inserting guard rings [6]. Furthermore, substrate parasitics at the back-gate of transistors depend on the layout floorplan and topology [7]. All these aspects need to be addressed by layout automation engines.

3. OPTIMIZATION-BASED MACRO-CELL PLACEMENT AND ROUTING

These methods extend the macro-cell based placement and routing approaches proposed for analog integrated circuits [8][9]. Figure 1 presents the flow diagram for these methods for RF integrated circuits.

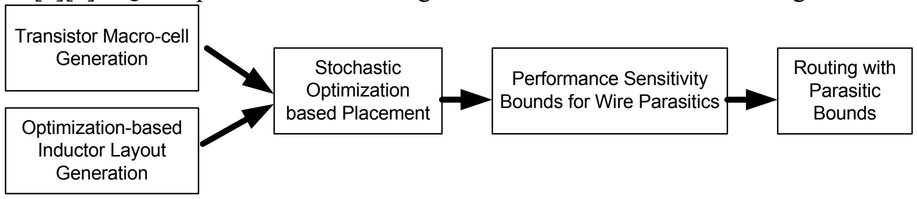

Transistor Macro-cell Generation
Optimization-based Inductor Layout Generation
Stochastic Optimization based Placement
Performance Sensitivity Bounds for Wire Parasitics
Routing with Parasitic Bounds

Figure 1: Methodology for optimization-based macro-cell placement and routing for RF circuits.

The layouts of individual transistors that comprise the macro-cells are either drawn manually by expert layout designers or are obtained by optimization under device parasitic constraints [10]. Several approaches have been proposed recently for automatic optimal on-chip inductor layout generation. The method in [11] uses an enumeration scheme where the geometric parameters of the inductor are first discretized, each combination of the resulting parameter values is then simulated with a modified nodal equation solver, and finally, the parameters that result in the best performance (e.g., maximum Q) are used to generate the layout. The limitation of this approach is that the complexity of such enumeration is inherently exponential with respect to the number of modifiable layout parameters. Another approach [12] uses geometric programming to generate optimal inductor layouts. For this, extensive pre-simulations are required on different inductor layout topologies and parameters to generate a curve-fitting based posynomial objective and constraint functions. Generating accurate closed-form posynomial expressions for parasitics at gigahertz frequencies is extremely difficult. Yet another approach [13] uses the discretization and simulation methods as in [11] but formulates the layout generation problem as a sequential quadratic optimization problem for superior running time. Unfortunately, the problem formulation is not necessarily convex and therefore the optimal solution is not guaranteed to be global.

As RF circuit performance depends strongly on relative placement of devices and wiring structure, [14] proposes an early floorplanning at the device-level based on a genetic algorithm. A slicing tree structure is assumed for the floorplan comprising of both active and passive devices. The method assumes that signal degradation specific constraints on wires are available and are mapped as length constraints. Inside the genetic optimization routine, individual floorplans are evaluated by a maze router that simultaneously attempts to reduce net crossings, maximize planarity and dynamically sizes channel dimensions.

Detailed routing entails mapping of high frequency performance specifications onto a set of bounds for distributed parasitics. The parasitics are calculated based on the models derived from 3-D electromagnetic field solvers. The scheme in [15] espouses this approach and derives sensitivity-based weights that are used to guide the routing solution. Another approach in [16] introduces a new routing algorithm that focuses on preserving planarity.

4. TEMPLATE-BASED LAYOUT AUTOMATION

Template based methods rely on using designer's knowledge in generating high quality layouts that meet the desired performance specifications. In these schemes, the relative position of devices is determined by experienced layout designers. Detailed sizing of devices and wires commences with the pre-determined relative positions as the starting point [17].

Recently, considerable progress has been made in template-based layout automation for analog circuits. The IPRAIL tool presented in [18] retargets existing high quality analog layouts to new process technologies and different circuit performance specifications. The underlying assumption is that the existing layout is extensively simulated after extraction of distributed parasitics and determined to be of high quality.

The details of the methodology and IPRAIL internals are shown in Figure 2. The IPRAIL tool consists of two main engines: the layout template extractor and the layout generator. The layout template extractor automatically extracts the designer's expertise embedded in intricate layouts as a set of linear constraints, collectively called a "symbolic template". The new device sizes pertaining to the new circuit specifications are obtained from circuit simulation and imposed on the symbolic template. The constraints are further modified according to the design and electrical rules for the new process technology. The generation of the new layout from this symbolic template then reduces to a modified compaction problem with symmetry constraints [19].

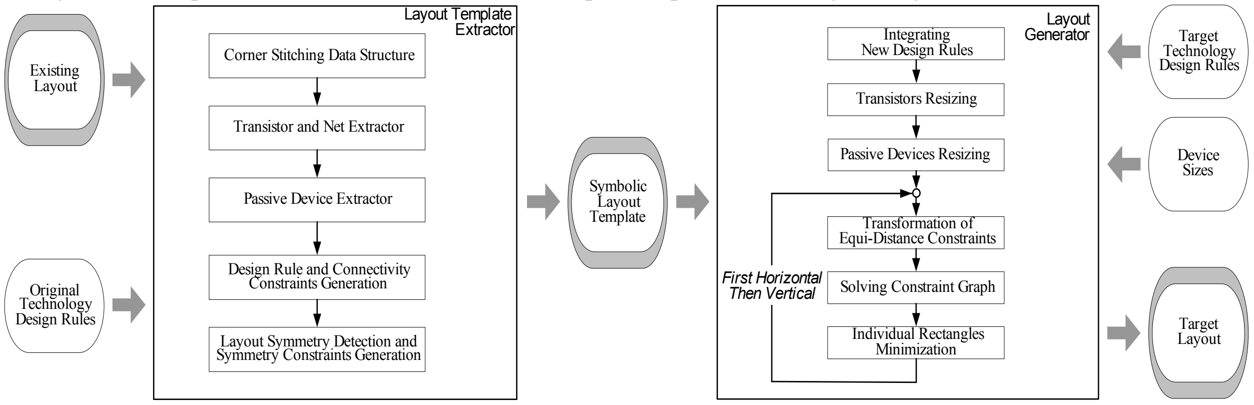

Existing Layout
Original Technology Design Rules
Layout Template Extractor
Corner Stitching Data Structure
Transistor and Net Extractor
Passive Device Extractor
Design Rule and Connectivity Constraints Generation
Layout Symmetry Detection and Symmetry Constraints Generation
Symbolic Layout Template
Layout Generator
Integrating New Design Rules
Transistors Resizing
Passive Devices Resizing
Transformation of Equi-Distance Constraints
Solving Constraint Graph
Individual Rectangles Minimization
First Horizontal Then Vertical
Target Technology Design Rules
Device Sizes
Target Layout

Figure 2: Symbolic template based layout retargeting methodology in IPRAIL

The IPRAIL framework has been extended in [20] to handle layout intricacies specific to RF circuits, especially retargeting of passive devices including on-chip spiral inductors. Here, the inductors in the target design are obtained from available inductor libraries that are characterized based on extensive electromagnetic simulations. In the template extraction step, the spiral inductors are automatically identified in the original RF layout. The mask rectangles connecting the spirals to the rest of the layout are called *ports*. Passive devices are usually isolated in space from other devices and nets to minimize the parasitic and coupling effects. In order to maintain this upon retargeting, [20] introduces the concept of *shadow rectangles*. This is illustrated in Figure 3. A shadow rectangle is a temporary non-physical mask layer whose objective is to reserve space. The original inductor is deleted, constraints added on the shadow rectangle for retargeting, and the new inductor is automatically generated in place of the shadow rectangle according to the library specifications.

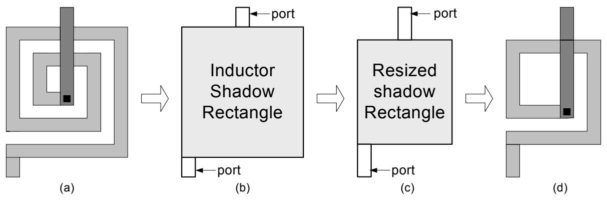


Figure 3: Inductor retargeting with shadow rectangles.

Substrate network models at the back-gate of transistors depend on the layout structures and bulk contact patterns around transistors [7]. These models are useful in automatically generating the bulk contacts around transistors in the target layout to minimize the substrate effect. The research in [20] also provides schemes for efficient retargeting for specific layout intricacies, e.g., RF layouts with millions of vias.

5. DISCUSSION

While the recent progress in RF layout automation is encouraging, both the classes of methods suffer from many shortcomings and substantial research is necessary before either of these methods or their combination can be viable for industrial application. The place and route schemes essentially suffer from two counts. Firstly, the models for evaluation of a solution quality are rather simplistic and therefore unrealistic for RF layouts. Thus, a seemingly "optimal" solution may fail to meet the desired circuit performance specifications in post-layout simulations and silicon measurements. Secondly, the method completely bypasses the layout-designer and therefore fails to leverage the accumulated expertise and intricate layout techniques.

The template-based methods reuse existing "high quality" layouts and therefore can implicitly leverages the intricate layout styles used by expert layout designers. Furthermore, since target layouts are generated from good existing layout (albeit for another technology or specification set), considerable amount of parasitic effects are implicitly controlled. However, actual qualification of the "goodness" of the existing layout greatly depends on the maturity of fully-coupled electromagnetic and circuit simulation techniques [21]. Furthermore, new algorithms need to be developed for explicit sizing and control of the parasitic elements for aggressive circuit specifications.

6. CONCLUSIONS

RF layout automation is very challenging as circuit performance is extremely sensitive to the intricacies of layout geometry. Template-based and stochastic optimization based macro-cell place and route methods have been proposed for RF layout automation. The place and route methods exhibit good flexibility but suffer from lack of meaningful performance-based models for evaluation of solutions and therefore cannot guarantee desired circuit performance. The template-based methods suffer from limited flexibility but can be effective for design retargeting. Both methods require significant innovations before they can be successfully applied to industrial RF circuits with aggressive performance specifications.

REFERENCES

[1] R. H. Jansen, "LINMIC: A CAD package for the layout-oriented design of single and multi-layer MIC's/MMIC's up to mm-wave frequencies", *Microwave Jour.*, pp. 151-161, Feb. 1986.

[2] R. H. Jansen, R. G. Arnold, I. G. Eddison, "A comprehensive CAD approach to design of MMIC's up to MM-wave frequencies", *IEEE Trans. Microwave Theory Tech.*, vol. 36, pp. 208-219, Feb.1988.

[3] M. J. M. Pelgrom, A. C. J. Duinmaijer and A. P. G. Welbers, "Matching properties of MOS transistors", *IEEE Jour. Solid State Circuits*, vol. 24, pp. 1433-1440, Oct. 1989.

[4] A. Hastings, *The Art of Analog Layout,* Prentice Hall, 2001.

[5] Q. Ouyang, S. J. Koester, J. O. Chu, K. L. Saenger, J. A. Ott and K. A. Jenkins, "Implications of gate design of RF performance of sub-100nm strained-Si/SiGe nMODFETs", *Proc. Int. Conf. on Simulation of Semiconductor Processes and Devices*, Sep. 2003, pp. 203-206.

[6] T. H. Lee, *The Design of CMOS Radio-Frequency Integrated Circuits*, Cambridge University Press, 1998.

[7] Z. Li, R. Suravarapu, R. Hartono, S. Bhattacharya, K. Mayaram and C-J. R. Shi, "CrtSmile: A CAD tool for CMOS RF transistor substrate modeling incorporating layout effects", *Proc. Asia and South Pacific Design Automation Conference,* Jan. 2004, pp. 163-168.

[8] J. M. Cohn, D.J. Garrod, R. A. Rutenbar and L. R. Carley, "KOAN/ANAGRAM II: New tools for device-level analog placement and routing", *IEEE Jour. Solid State Circuits*, vol. 26, pp. 330-342, Mar. 1991.

[9] K. Lampaert, G. Gielen and W. M. Sansen, "A performance-driven placement tool for analog integrated circuits", *IEEE Jour. Solid State Circuits*, vol. 30, pp. 773-780, Jul. 1995.

[10] E. Malavasi and D. Pandini, "Optimum CMOS stack generation with analog constraints", *IEEE Trans. Computer-Aided Design*, vol. 14, pp. 107-122, Jan. 1995.

[11] A. M. Niknejad and R. G. Meyer, "Analysis, design, and optimization of spiral inductors and transformers for Si. RF ICs", *IEEE Jour. Solid State Circuits*, vol. 33, pp. 1470-1481, Oct. 1998.

[12] M. D. M. Hershenson, S. S. Mohan, S. P. Boyd and T. H. Lee, "Optimization of inductor circuits via geometric programming", *Proc. IEEE/ACM Design Automation Conference*, Jun. 1999, pp. 994-998.

[13] Y. Zhan and S. S. Sapatnekar, "Optimization of integrated spiral inductors using sequential quadratic programming", *Proc. IEEE/ACM Design Automation and Test in Europe Conference*, Mar. 2004, pp. 1-6.

[14] M. Aktuna, R. A. Rutenbar and L. R. Carley, "Device-level early floorplanning algorithms for RF circuits", *IEEE Trans. Computer-Aided Design*, vol. 18, pp. 375-388, Apr. 1999.

[15] E. Charbon, G. Holmlund, B. Donecker and A. Sangiovanni-Vincentelli, "A performance-driven router for RF and microwave analog circuit design", *Proc. IEEE Custom Integrated Circuits Conference*, May 1995, pp. 383-386.

[16] A. Nagao, I. Shirakawa and T. Kambe, "A layout approach to monolithic microwave IC", *IEEE Trans. Computer-Aided Design*, vol. 17, pp. 1262-1272, Dec. 1998.

[17] J. F. Zurcher, "MICROS – A CAD/CAM program for fast realization of microstrip masks", *IEEE Trans. Microwave Theory Tech.*, vol. MTT-S, pp. 481-484, Jun. 1985.

[18] N. Jangkrajarng, S. Bhattacharya, R. Hartono, and C-J. R. Shi, "IPRAIL – Intellectual property reuse-based analog IC layout automation", *Integration – The VLSI Journal*, vol. 36, pp. 237-262, Nov. 2003.

[19] R. Okuda, T. Sato, H. Onodera and K. Tamaru, "An efficient algorithm for layout compaction problem with symmetry constraints", *Proc. Int. Conf. on Computer-Aided-Design*, Nov. 1989, pp. 148-151.

[20] N. Jangkrajarng, S. Bhattacharya, R. Hartono, and C-J. R. Shi, "Multiple specifications radio-frequency integrated circuit design with automatic template driven layout retargeting", *Proc. Asia and South Pacific Design Automation Conference,* Jan. 2004, pp. 394-399.

[21] Y. Wang, D. Gope, V. Jandhyala and C-J. R. Shi, "Integral equation-based coupled electromagnetic-circuit simulation in the frequency domain", *Proc. Int. Symp. of Antennas and Propagation Society*, Jun. 2003, pp. 22-27.

Automatic Device Layout Generation for Analog Layout Retargeting*

Roy Hartono, Nuttorn Jangkrajarng, Sambuddha Bhattacharya and C.-J. Richard Shi

Department of Electrical Engineering, University of Washington, Seattle WA 98195, USA
{rhartono,njangkra,sbb,cjshi}@ee.washington.edu

Abstract

This paper presents a technique for automatic active device layout generation and insertion incorporated in a layout retargeting tool-suite for analog integrated circuits. While the use of a graph-based symbolic template in the retargeting tool maintains the overall layout topology, layout symmetries, and embedded expertise of the designers, the device generator allows further optimization of active devices in terms of device width, length, and finger variables through template modification. Combining the device layout generator with a design-space exploration engine that searches for optimal sets of design variables satisfying performance requirements, a new automatic design reuse methodology is presented. Multiple high quality analog circuits corresponding to different target specifications are synthesized in less than an hour, and their layouts with different device sizes and structures are generated in less than a minute of CPU time.

1. Introduction

The difficulty in analog layout generation lies in ensuring achievement of the desired performance specifications upon fabrication. Therefore, analog layouts, traditionally, have been drawn manually by expert layout designers at the cost of increased design cycle time. In order to address this issue, [1,2] presents a methodology for analog layout automation by employing a layout reuse principle, that incorporates the designer's expertise embedded in existing layouts. In this method, an existing analog layout is automatically converted into a collection of constraints called a symbolic template, which is then solved by linear programming and graph-based longest path algorithms.

The desired performance of post-layout designs of analog circuits is greatly affected by device parasitics arising from multi-finger transistors [3,4], layout symmetry for matched transistors [3,5], and relative positions of devices and nets. Unfortunately, in the previous layout retargeting engines [1,2], the automatic creation of the symbolic templates limits changes in layout structures, specifically the number of transistor fingers and the passive device structures. This may constrain the layout solution quality, especially for circuits that are sensitive to parasitic capacitances or resistances. To overcome this, the integrated device layout generator, presented in this paper, helps to alter device layouts by automatically modifying the symbolic templates. Automatic passive device layout generation technique, with structure modification, in the layout-retargeting tool-suites has been addressed in [6]. This paper is targeted towards the challenges and techniques in an automatic layout generation for active devices within a layout-retargeting framework.

Unlike the stand-alone device layout generation methods presented in [7,8], this paper addresses the active device layout generation as a part of a layout retargeting engine like [1,2]. This mitigates a flexibility problem with layout retargeting since device topology can be changed while retaining the remaining properties of the existing layout, e.g., relative placement of devices and wires, wiring topology, guard rings etc. Furthermore, [7,8] attempted to optimize device folding by approximate cost models for diffusion capacitances and may result in suboptimal designs. In contrast, this work formulates the problem as a part of design performance optimization by simulation-based design space exploration [9,10]. A tighter integration of the layout automation with design synthesis ensures optimal design performance.

In this paper, a complete design reuse methodology is presented, in which the layout retargeting engine [1,2] is enhanced with the automatic active device layout generation and the commercial design space exploration engine [9]. In this methodology, a parametric netlist is obtained from an existing layout, which also includes transistor widths, lengths, and number of fingers as variables. Along with several target specifications and technology process parameters, the netlist is simulated and optimized by the design space exploration engine to find device geometry variables for each specification. Target layouts are then generated through the layout retargeting engine, where transistor layouts are reconstructed through a series of device generation processes. These include multi-finger transistor detection and removal, graph template construction, wiring restoration and device insertion, symmetry enforcement, and layout generation.

This paper is organized as follows. Section 2 discusses the layout-centric design reuse methodology that incorporates device layout restructuring. Section 3 presents the detailed steps in the active device layout restructuring as a part of layout retargeting. Section 4 presents the experimental results. And section 5 concludes the paper.

2. Design Reuse Methodology

This work incorporates the layout centric analog design reuse methodology of [1,2]. As illustrated in Fig. 1, a layout-retargeting engine is at the core of this methodology along with a simulation-based design space exploration tool. The layout retargeting tool reads in an existing high-quality

* This research has been supported in part by the U.S. Defense Advance Research Projects Agency's NeoCAD program and in part by the National Science Foundation's ITR program.

manually crafted analog layout, automatically creates a constraint-based symbolic template and solves the template to generate target layouts.

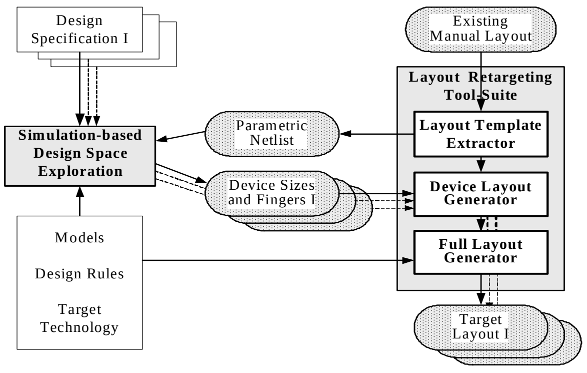


Fig. 1: A design reuse methodology flow, including layout retargeting, device layout generation and design optimizer.

While the retargeting engines in [1,2] do not change the number of fingers of the transistors, the improved layout-retargeting tool incorporates the automatic device layout generation that allows changes in the transistor layout structures. The layout template extractor creates a resizable symbolic template and generates a parametric netlist. This netlist contains transistor widths, lengths and number of fingers as design variables from which the simulation-based optimizer [9] synthesizes multiple target designs according to the design goals. The device sizes and number of fingers obtained from the synthesis tool are then used for automatic device layout generation. Finally, the layout generator incorporates the new transistor structures and generates layouts of the whole analog design according to the target specifications and/or technologies.

2.1 Symbolic Template Extraction

The symbolic template extraction essentially involves the automatic generation of connectivity, design-rule and layout symmetry constraints from the input layout. The constraints are generally linear and ensure that the generated target layouts retain the same intricacies as the input layout. For each layout, two separate sets of constraints are generated, one for horizontal and one for vertical directions.

As illustrated in Fig. 2, the template extraction involves identification of active and passive devices, generation of a parametric netlist, generation of design-rule and connectivity constraints and extraction of constraints due to device layout symmetry. The transistor, net and passive device identification uses the techniques presented in [6,11]. The generation of design-rule and connectivity constraints employs the scan-line method [12].

The symbolic layout template is constructed as a constraint graph from the linear constraint equations as shown in Fig 3. Each rectangle in the layout is transformed into four independent nodes, representing its left, right, top, and bottom edges. Connectivity and design rule constraints are placed between nodes in the graph to sustain layout integrity and correctness. The connectivity between rectangles *M* and *N* in the horizontal direction is ensured by a pair of constraint arcs of weight '*0*' between edges *p4* and *p5*. The design rule constraint is further decomposed into three types: minimum width – an arc from *p1* to *p2*, minimum spacing – an arc from *p2* to *p3*, and minimum extension – an arc from *a2* to *p4*.

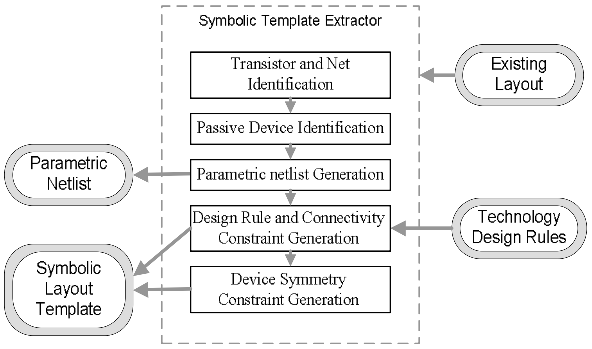

Symbolic Template Extractor
Transistor and Net Identification
Existing Layout
Passive Device Identification
Parametric Netlist
Parametric netlist Generation
Design Rule and Connectivity Constraint Generation
Technology Design Rules
Symbolic Layout Template
Device Symmetry Constraint Generation

Fig. 2: Template extractor flow.

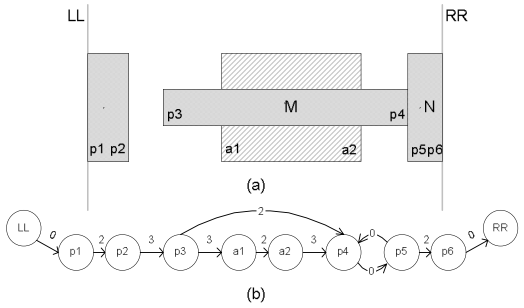


Fig 3: A layout and its horizontal graph symbolic template.

Matching between a pair of transistors is established by laying out the transistors symmetrically. Two transistor layouts are deemed symmetric if they are geometric mirror images of each other. As illustrated in a simplified example of Fig. 4, this implies equi-sized channel, drain and source regions, identical orientation and close proximity of the two transistors. Transistor layout symmetry is extracted according to the method presented in [2,13]. Constraints due to layout symmetry are imposed according to the following equations.

$$(e_{bottom} - f_{bottom}) = 0 \quad (1)$$

$$(s_0 - g_{right}) - (h_{left} - s_0) = 0 \quad (2)$$

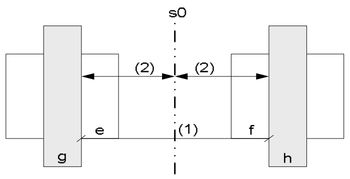


Fig. 4: A simplified layout of two transistors symmetric about axis 's_0'.

2.2 Automatic Device Layout Generation

Generating optimal target layouts requires flexibility in device reconfiguration, so that the layout parasitics can be controlled. Therefore, the device width, length, and fingers are passed as variables along with the parametric netlist to the synthesis tool to obtain the optimal solutions. These solutions are then used to generate the target layouts by means of device layout construction and subsequent full layout generation.

As part of the parametric netlist, two relevant device parasitics – diffusion-to-bulk capacitance and gate poly resistance – are approximated based on the geometry and number of fingers of each transistor and are added to the netlist in a parametric form. Fig. 5 shows a transistor layout and its extracted parasitic model, where *w* is the total width, *l* is the length, *m* is the number of fingers, and *d* is the diffusion size calculated from the contact size and spacing distance between contact and polysilicon.

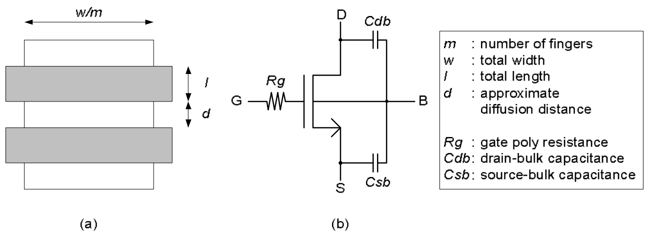

w/m
l
d
D
Cdb
Rg
G
B
S
Csb
m : number of fingers
w : total width
l : total length
d : approximate diffusion distance
Rg : gate poly resistance
Cdb: drain-bulk capacitance
Csb: source-bulk capacitance
(a)
(b)

Fig. 5: A layout of a multi-finger transistor and its transistor schematic showing R_g, C_{db} and C_{sb}

The diffusion to bulk capacitances, in terms of C_{db} and C_{sb}, are approximated in terms of areas and perimeters of the drain and source (as A_s, A_d, P_s, and P_d) and are added to each transistor model. Since the drain and source areas of the CMOS are not assigned specifically in the layout and in order to simplify the equation for any number of fingers, the approximate areas and perimeters can be calculated as:

$$A_d = A_s = wd\,(m+1)/(2m) \tag{3}$$

$$P_d = P_s = d(m+1) + w/m \tag{4}$$

The gate resistor (R_g) is added to the netlist and is calculated based on [3] as

$$R_g = (1/3m)\,(\rho w/ml) \tag{5}$$

where ρ is the resistivity of polysilicon.

Using the parametric netlist and the technology process parameters, the synthesis engine invokes multiple circuit simulation runs during the optimization process and obtains the set of design variable values that meet the specifications. Each transistor's size and number of fingers are assigned as variables and various target specifications are passed as goals for the tool.

To minimize the topology changes, each device that maintains the number of fingers is resized by simply adding fixed-width constraints to represent its new width and length. However, the devices that require changes in the number of fingers are retargeted through a process of layout geometry removal and construction, explained in detail in section 3.

2.3 Full Layout Generation from Symbolic Template

The new layout generation process minimizes the layout area subject to the constraints generated by template extractor and updated by the new device sizes or the active device layout generator. Mathematically, for the horizontal direction, the problem is defined as follows:

$$Min\ (\,x_R - x_L\,) \tag{6.1}$$

$$\text{subject to}\quad x_i - x_j \le constant \tag{6.2}$$

$$x_i - x_j = x_k - x_l \tag{6.3}$$

This is essentially a modified compaction problem [12]. As linear programming (LP) [14] is computationally expensive, this problem is solved by combining the graph-based longest path algorithm with LP. This entails the transformation of the equi-distance constraints of Equation (6.3) into a graph-imposable form of Equation (7) according to the technique presented in [15].

$$x_i - x_j = x_k - x_l = constant \tag{7}$$

After the constraint graph is updated with the derived form of equi-distance constraints, the graph-based longest path algorithms [16] is employed to generate a feasible solution for the final target layout. As the longest-path algorithm may introduce unwanted wiring parasitics, individual rectangle minimization [17] is performed before generating the target layout. Fig. 6 shows the detailed steps in layout generation.

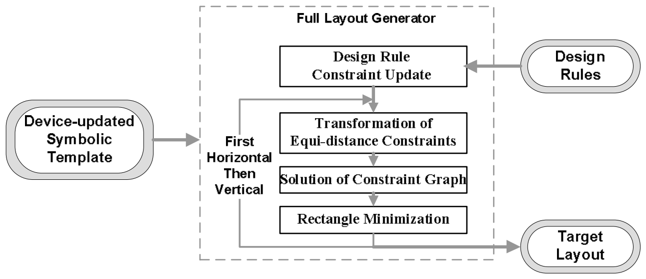

Full Layout Generator
Design Rule Constraint Update
Design Rules
Device-updated Symbolic Template
First Horizontal Then Vertical
Transformation of Equi-distance Constraints
Solution of Constraint Graph
Rectangle Minimization
Target Layout

Fig. 6: Final layout generation flow.

3. Active Device Generation

In the layout generation process, varying the number of device fingers requires a modification in the extracted symbolic template. This restructuring process, presented in detail in [18], is accomplished by the active device layout generation for successful layout retargeting.

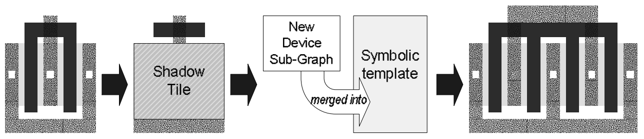


Fig. 7: Simplified process of active device layout generation

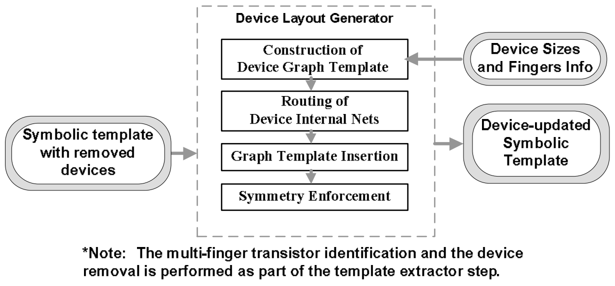

Device Layout Generator
Construction of Device Graph Template
Device Sizes and Fingers Info
Routing of Device Internal Nets
Symbolic template with removed devices
Graph Template Insertion
Device-updated Symbolic Template
Symmetry Enforcement
*Note: The multi-finger transistor identification and the device removal is performed as part of the template extractor step.

Fig. 8: Device layout generator flow.

The simplified device generation process is illustrated in Fig. 7. Fig. 8 shows various steps in the device layout generator, involving identification of multi-finger transistors, physical removal of modified devices, construction of new device graph templates, routing of device internal nets and ports, graph merging for device insertion, and symmetry enforcement.

3.1 Multi-finger Transistor Identification and Removal

During the initial layout extraction process, every discrete transistor is identified when there is an overlap between polysilicon and diffusion layers. Discrete transistors that are parallel connected and laid-out contiguously on one diffusion layer rectangle are clustered together as a contiguous element. One or more parallel connected contiguous elements are grouped together as a multi-finger transistor.

Prior to the reconstruction process, every corresponding device with modified number of fingers has to be removed from the layout and replaced by a rectangle on a temporary layer, called a shadow rectangle, to represent its existence in the layout and to provide a dedicated space for the later device insertion. In this removal phase, only the device core layout and its internal wiring are deleted from the input layout. However, the connection information at ports, rectangles that connect the device with the rest of the layout polygons, is retained. These identification and removal procedures are performed along with the device identification steps in the symbolic template extraction section.

3.2 New Device Graph Template Construction

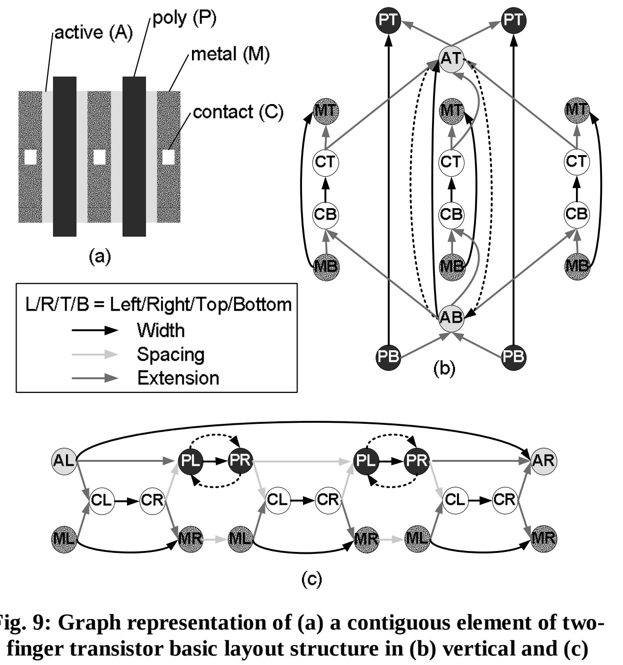


Fig. 9: Graph representation of (a) a contiguous element of two-finger transistor basic layout structure in (b) vertical and (c) horizontal directions.

In this phase, each individual device is recognized and constructed as a contiguous element. Based on the geometry variable solutions calculated by the circuit optimization tool both horizontal and vertical sub-graphs are built for each device. These sub-graphs have a similar structure as the rest of the symbolic template where a rectangle edge is represented by a node, and a constraint between edges is represented by an arc. For each sub-graph, in order to construct a device with *N* number of fingers, the device generator creates a set of nodes to represent *one* diffusion rectangle, *N* poly rectangles, and *(N+1)* source or drain terminal rectangles, each consists of a contact and a metal-one. The nodes positions are maintained by constraint arcs based on the design rule and device sizes. For the two-finger transistor of Fig. 9(a), the corresponding horizontal and vertical graphs are shown in Fig. 9(b) and (c) respectively.

3.3 Device Insertion and Wiring Restoration

To proceed with the retargeting process, all sub-graphs of the newly created devices need to be merged into the main symbolic template. To ensure the same device layout orientation, the sub-graph may be rotated prior to the insertion. The device sub-graph is inserted within the space allocated by the shadow rectangle nodes. The layout integrity is maintained by imposing the design rule constraints between the inserted device and its surrounding rectangles.

After the insertion, the device connectivity is restored using a simple rectangle-based dogleg routing technique according to the port locations obtained during the removal process. In this procedure, all rectangles required for routing are generated as graph nodes and added to the sub-graph via the connectivity arcs.

3.4 Symmetry Enforcement and Layout Generation

Symmetries between transistors are important in the analog layout and, therefore, have to be maintained between the newly generated devices during the retargeting process. Symmetry constraints are imposed between a pair of new devices only if the symmetry exists in the original layout and both new devices have identical sizes and orientation. The symmetry is maintained by imposing equi-distance and alignment constraints between the device fingers and the symmetry axis. This completes imposition of all types of constraints necessary for active device generation.

Next, the layout of the transistors along with the rest of layout comprising passive devices and wires is finalized by the layout generator, in which the graph template is solved through the graph-based longest-path algorithm and the rectangle minimization algorithm.

In the template construction and insertion process, each drain or source terminal area is initially populated by only one contact. In order to reduce the resistance due to contacts, the terminal area are populated with multiple contacts after the rectangle minimization process is completed.

4. Results

This section presents the results of applying the improved analog layout re-targeting methodology on a single-ended folded-cascode operational amplifier (opamp) to generate several layouts with different specifications. The methodology coupled a simulation-based circuit optimizer (Arsyn [9]), an automatic layout retargeting tool, and a device layout generator.

An input folded-cascode opamp was designed and laid out in TSMC 0.18µm CMOS process. It consisted of 14 multi-finger transistors. Fig. 10 shows the schematic of the opamp, and Fig. 11 shows the initial layout. The opamp specifications for the initial design are listed in the second row of Table 1 (marked as *original*).

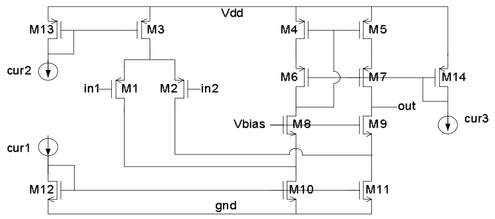


Fig. 10: Schematic of a single-output folded-cascode opamp.

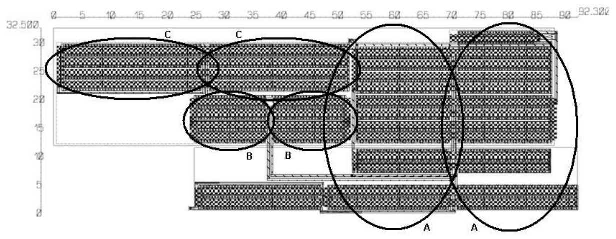


Fig. 11: Input layout of the folded cascode opamp. A, B, and C are transistor symmetry blocks.

In this paper, four different specifications were aimed. Table 1 lists out all goal specifications the optimizer targeted for, as well as the specifications achieved after layout generation. The first target was set for an overall performance. The second, third and fourth targets were set to maximize bandwidth, gain, and phase margin, respectively. The transistor geometry sizes and the number of fingers were optimized to meet each specification and are shown in Table 2.

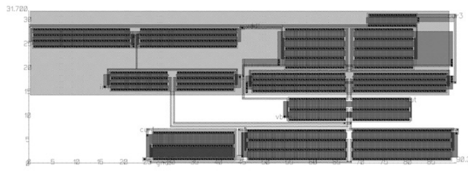


Fig. 12: A target layout of specification I in Table 1 and 2 – aiming for overall specification.

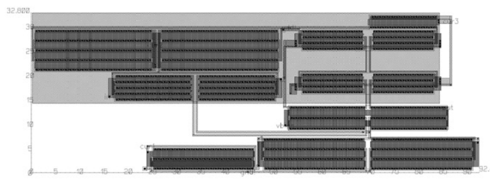


Fig. 13: A target layout of specification II in Table 1 and 2 – aiming for bandwidth maximization.

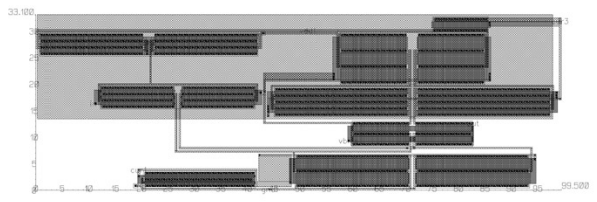


Fig. 14: A target layout of specification III in Table 1 and 2 – aiming for gain maximization.

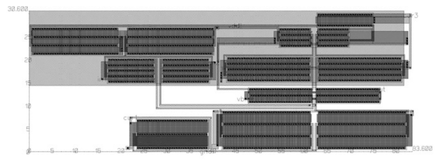


Fig. 15: A target layout of specification IV in Table 1 and 2 – aiming for phase margin maximization.

With the collaboration between the circuit optimizer, the layout retargeting tool, and the active device layout generator, the original folded cascode operational amplifier layout was successfully retargeted into four different specifications, as shown in Fig. 12 to Fig. 15. The post layout simulation, presented in Table 1, shows all target layouts successfully satisfied their specific goals. The runtimes of both the optimizer and the layout retargeting tool on a Sun Ultra10 workstation are reported in Table 1. In case of any unrealistic goals, the circuit optimizer will try to meet partial goals based on assigned cost and layouts are generated. Or each specification can be re-adjusted until all desired goals are met.

5. Conclusions

In this paper, a layout retargeting methodology combined with a design space exploration engine and active device generation is presented. The automatic device layout generation allows the retargeting tool to vary the device sizes and fingers. This provides superior control over the layout parasitic effects that helps achieve more demanding design specifications. The use of design space exploration engine coupled with layout optimization helps avoid costly design iterations and greatly reduces the manual efforts and solution time. Our methodology and tools have successfully generated various highly optimized analog layouts targeted at different specifications.

REFERENCES

[1] N. Jangkrajarng, S. Bhattacharya, R. Hartono, and R. Shi, "IPRAIL – Intellectual property reuse-based analog IC layout automation", *Integration – The VLSI Journal*, vol. 36, pp. 237-262, Nov. 2003.

[2] S. Bhattacharya, N. Jangkrajarng, R. Hartono and C-J. R. Shi, "Correct-by-construction layout-centric retargeting of large analog designs", *Proc. IEEE/ACM Design Automation Conf.,* Jun. 2004, pp. 139-144.

[3] B. Razavi, *Design of Analog CMOS Integrated Circuits*, McGraw Hill, 2001.

[4] A. Hastings, *The Art of Analog Layout*, Prentice Hall, 2001.

[5] M. J. M. Pelgrom, A. C. J. Duinmaijer and A. P. G. Welbers, “Matching properties of MOS transistors”, *IEEE J. Solid State Circuits*, vol. 24, pp. 1433-1440, Oct. 1989.
[6] N. Jangkrajarng, S. Bhattacharya, R. Hartono, and C-J. R. Shi, “Multiple specifications radio-frequency integrated circuit design with automatic template-driven layout retargeting”, *Proc. Asia and South-Pacific Design Automation Conf.*, Jan 2004, pp. 394-399.
[7] E. Malavasi, D. Pandini and V. Liberali, “Optimum stacked layout for analog CMOS ICs”, *Proc. IEEE Custom Integrated Circuits Conf.*, Mar. 1993, pp. 17.1.1-17.1.4.
[8] B. Basaran and R. A. Rutenbar, “An O(n) algorithm for transistor stacking with performance constraints”, *Proc. IEEE/ACM Design Automation Conf.*, Jun. 1996, pp. 221-226.
[9] Arsyn User’s Manual, Orora Design Technologies Inc., 2003.
[10] M. J. Krasnicki, R. Phelps, J. R. Hellums, M. McClung, R. A. Rutenbar and L. R. Carley, “ASF: A practical simulation-based methodology for the synthesis of custom analog circuits”, *Proc. Int. Conf. Computer-Aided-Design*, Nov. 2001, pp. 350-357.
[11] W. S. Scott and J. K. Ousterhout, “Magic’s circuit extractor”, *Proc. IEEE/ACM Design Automation Conf.*, Jun. 1985, pp. 286-292.
[12] S. L. Lin and J. Allen, “Minplex – a compactor that minimizes the bounding rectangle and individual rectangles in a layout”, *Proc. IEEE/ACM Design Automation Conf.*, Jun. 1986, pp. 123-130.
[13] S. Bhattacharya, N. Jangkrajarng, R. Hartono, and C-J. R. Shi, “Hierarchical extraction and verification of symmetry constraints for analog layout automation”, *Proc. Asia and South-Pacific Design Automation Conf.*, Jan 2004, pp. 400-405.
[14] D. Luenberger, *Linear and Nonlinear Programming* 2nd Edition, Addison-Wesley, 1984.
[15] R. Okuda, T. Sato, H. Onodera and K. Tamaru, “An efficient algorithm for layout compaction problem with symmetry constraints”, *Proc. Int. Conf. Computer-Aided-Design*, Nov. 1989, pp. 148-151.
[16] T. H. Cormen, C. E., Leiserson and R. L. Rivest, *Introduction to Algorithms*, MIT Press, 2001.
[17] G. Lakhani and R. Varadarajan, “A wire-length minimization algorithm for circuit layout compaction”, *Proc. Int. Symposium on Circuits and Systems*, May 1987, pp. 276-279.
[18] R. Hartono, N. Jangkrajarng, S. Bhattacharya, and C-J. R. Shi, “Active device generation for automatic analog layout retargeting tool”, *University of Washington Electrical Engineering Technical Report UWEETR-2004-0015*, 2004.

Table 1: An existing specification and a performance comparison - between goals, specifications optimized by the synthesis tool, and specifications from the retargeted layouts - of a folded-cascode operational amplifier.

Layout		**Bandwidth (MHz)**	**Gain (dB)**	**Phase Margin (deg)**	**Gain Margin (dB)**	**Power (mW)**	**Area (μm^2)**	**Runtime**
Original		*71.7*	*61.9*	*42.0*	*12.4*	*1.072*	*3,000.0*	-
Target I (overall)	**Goal**	*90.0*	*60.0*	*60.0*	*12.0*	-	-	-
	Synthesis	*96.4*	*60.5*	*60.3*	*25.5*	*1.406*	-	*4 m 12 s*
	Layout	*95.5*	*58.2*	*61.1*	*26.7*	*1.469*	*2,862.5*	*11.9 s*
Target II (max. BW)	**Goal**	***120.0***	*50.0*	*50.0*	*12.0*	-	-	-
	Synthesis	***127.2***	*63.4*	*62.9*	*12.1*	*1.059*	-	*37 m 20 s*
	Layout	***125.3***	*63.4*	*62.6*	*12.0*	*1.059*	*3,030.7*	*13.3 s*
Target III (max. Gain)	**Goal**	*60.0*	***80.0***	*50.0*	*12.0*	-	-	-
	Synthesis	*64.6*	***80.3***	*49.8*	*28.9*	*0.797*	-	*55 m 05 s*
	Layout	*63.1*	***80.2***	*50.2*	*29.0*	*0.797*	*3,293.4*	*12.8 s*
Target IV (max. PM)	**Goal**	*60.0*	*50.0*	***80.0***	*12.0*	-	-	-
	Synthesis	*86.4*	*56.0*	***80.2***	*21.0*	*1.503*	-	*52 m 55 s*
	Layout	*84.3*	*56.0*	***80.0***	*21.5*	*1.503*	*2,558.2*	*12.7 s*

Table 2: Multi-finger transistor sizes and currents obtained from the synthesis tool for each target specifications.

Size	**Original**	**Target I**	**Target II**	**Target III**	**Target IV**
multi-finger transistors in total width (μm) / total length (μm) / # of fingers					
M1 & M2	*48.0 / 1.2 / 4*	*36.0 / 0.4 / 3*	*62.4 / 0.4 / 4*	*41.4 / 0.4 / 3*	*40.0 / 0.4 / 4*
M3 & M13	*96.0 / 1.2 / 4*	*61.2 / 0.4 / 3*	*96.0 / 1.2 / 4*	*58.2 / 0.4 / 3*	*56.4 / 1.0 / 3*
M4 & M5	*63.6 / 1.2 / 4*	*49.6 / 1.2 / 4*	*37.2 / 0.4 / 3*	*37.2 / 2.2 / 3*	*20.4 / 0.4 / 3*
M6 & M7	*63.6 / 1.2 / 4*	*58.8 / 0.4 / 3*	*37.2 / 0.4 / 3*	*100.0 / 0.4 / 4*	*72.0 / 0.4 / 4*
M8 & M9	*31.2 / 1.2 / 2*	*24.0 / 1.2 / 2*	*31.2 / 1.2 / 2*	*21.2 / 1.0 / 2*	*27.6 / 0.4 / 2*
M10 & M11	*41.4 / 1.2 / 2*	*84.0 / 1.2 / 3*	*62.4 / 1.2 / 3*	*42.4 / 2.0 / 2*	*57.6 / 1.8 / 3*
M12	*41.4 / 1.2 / 2*	*34.4 / 1.2 / 2*	*41.6 / 1.2 / 2*	*41.6 / 0.4 / 2*	*30.8 / 2.2 / 2*
M14	*13.8 / 1.2 / 1*	*10.2 / 1.2 / 1*	*13.8 / 1.2 / 1*	*10.2 / 1.2 / 1*	*12.0 / 1.4 / 1*
current (μA)					
I1	*140*	*100*	*100*	*100*	*120*
I2	*80*	*70*	*70*	*70*	*50*
I3	*100*	*130*	*120*	*120*	*140*

IPRAIL MANUAL

Sambuddha Bhattacharya, Roy Hartono, Nuttorn Jangkrajarng and C-J. Richard Shi
MSCAD Research Group, University of Washington, USA

1. Introduction

Analog circuit performance is strongly dependent on the layout intricacies. Several layout features like symmetry and matching, relative placement of devices, alleviation of parasitic effects are incorporated into analog layouts by expert layout designers. Complete automation of analog layout generation similar to digital design methodology is rather difficult.

Our analog layout automation methodology is based on the concept of design re-use. IPRAIL re-uses existing analog layouts crafted by layout designers and retargets them to different technologies and specifications.

2. Methodology

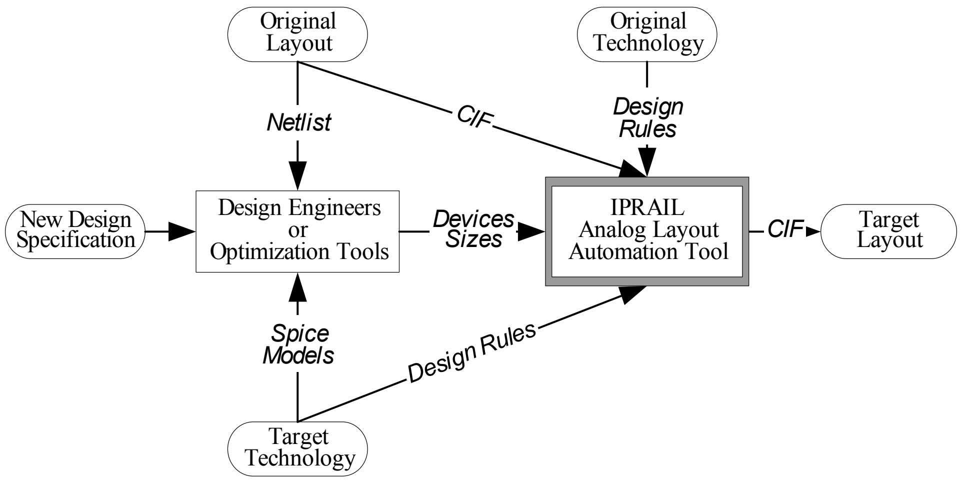


Figure 1: IPRAIL Layout Automation Framework

As illustrated in Figure 1, the *original layout* and its technology information are first fed into IPRAIL. The device sizes under the new specifications are obtained either by manual simulations or from an analog circuit synthesis tool. First, IPRAIL converts the original layout into a *resizable symbolic template*. It then generates the new layout, henceforth called *target layout*, by imposing the target process design rules and new device sizes as constraints on the symbolic template. The entire process of automatic creation of symbolic template and generation of target layout takes a few minutes of CPU time.

3. Installation

The platform required for *IPRAIL* is SunOS 5.8 (Solaris 8). *IPRAIL* also requires the X11 graphics package normally available in the Solaris environment. Please follow the instructions below to setup the environment for *IPRAIL*.

1. Copy the *iprail_demo.tar.gz* file into your home directory and deflate it.
    a. *gunzip iprail_demo.tar.gz*
    b. *tar -xvf iprail_demo.tar*
2. Check the directory structure created in your home.
    a. *~yourhome/demo*
    b. *demo* has the following subdirectories: *bin, env, icn, lib, sim* and *cadence*.
3. Execute the script *install.csh* from the *demo* directory. It includes some Cadence path setup in the directory *~yourhome/demo/cadence*.
4. *source .iprail* in *~yourhome/demo*
5. Prior to invoking IPRAIL, please exit *netscape* and other programs that use Graphics. Otherwise some technology layers may remain transparent in IPRAIL. That may be confusing for an inexperienced user.

This completes the environment setup for *IPRAIL*.

4. Tutorial

We will work with two examples of analog layouts.

a. Cascode Operational Amplifier
b. Two-stage Operational Amplifier

The schematics of the two examples are shown in the following figures.

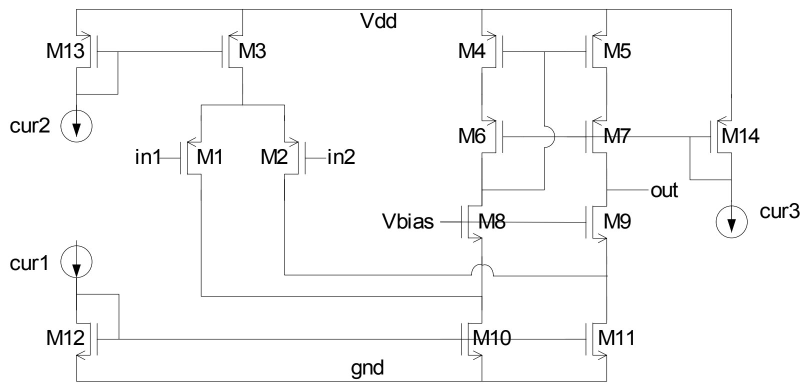


Figure 2. Schematic of a Folded-cascode Single Ended Operational Amplifier

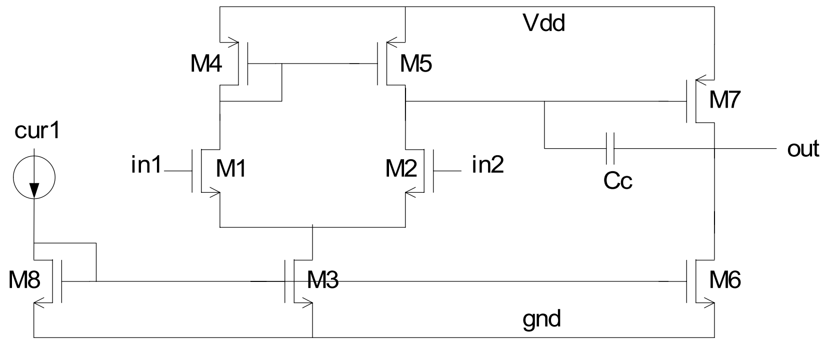


Figure 3. Schematic of a Two-Stage Operational Amplifier with Compensation

The handcrafted layouts for the two examples in TSMC 0.25um technology are available as CIF files. Our goal is to automatically generate corresponding layouts in the TSMC 0.18um technology. It is assumed that the design and simulation for the specifications in the new technology (0.18um) is already complete and available.

The layout (CIF), device sizes and symmetry information for the two examples are available in the directories *demo/bin/Cascode* and *demo/bin/TwoStage* . The technology and design rule files are available in the directories *demo/bin/Technology* and *demo/bin/DesignRule* respectively. The *demo/bin/BridgeFile* directory contains a *bridgefile* which relates between the 0.25um and 0.18um technologies. All these files need to be loaded into IPRAIL at some stage. At a later stage, some of these files may be made transparent to the user.

The complete tutorial is divided into five subsections. The first three subsections step through the entire automatic layout generation flow. The last two subsections step through the Design Rule Checking (DRC) and Post-Layout Simulation for verifying the correctness of the automatically generated layout. It is assumed that the Cadence environment is available to the user.

4.1 Loading the Original (0.25um) Layout into IPRAIL

This sub-section describes the process for invoking IPRAIL and loading the original hand-crafted layouts. Once the layout is loaded, the circuit structure is automatically extracted from the layout and stored in the database.

1. Go to *~yourhome/demo/bin* directory
2. Invoke IPRAIL by typing *iprail* in the Unix command line. Maximize IPRAIL GUI by clicking the button on top-right corner.
3. Load the IPRAIL's technology file. (*.tch)
    a. Click on 'Load' in the menu bar
    b. Click on 'Technology'
    c. Select the Technology file *Technology/MCNC.tch*
    d. Click 'OK'

4. Load the bridge file for the loaded technology
    a. Click on ‘Load’ in the menu bar
    b. Click on ‘BridgeFile’
    c. Select the Bridge file *BridgeFile/bridgeFileTSMC*
    d. Click ‘OK’
5. Load the initial design rule
    a. Click on ‘Load’ in the menu bar
    b. Click on ‘InitialDesignRule’
    c. Select the design rule file *DesignRule/UWtsmc25.tech*
    d. Click ‘OK’
    e. Clicking the correct technology file is very important for successful operation of IPRAIL. Make sure you’ve chosen the right technology file.
6. Load the target design rule
    a. Click on ‘Load’ in the menu bar
    b. Click on ‘TargetDesignRule’
    c. Select the target design rule file *DesignRule/UWtsmc18_updated.tech*
    d. Click ‘OK’
    e. Clicking the correct technology file is very important for successful operation of IPRAIL. Make sure you’ve chosen the right technology file.
7. Load the layout CIF file
    a. Click on ‘Load’ in the menu bar
    b. Click on ‘Layout(CIF)’
    c. Select the layout file *Cascode/*.cif* or *TwoStage/*.cif*
    d. Click ‘OK’

With this, you should be able to view the layout in 0.25um in the IPRAIL GUI.

4.2 Resizing the Layout

This sub-section steps through the most important functions in IPRAIL. The symmetry information in the layout is extracted in the symmetry detection phase. A symbolic template is created internally from which the new layout will be generated. The new device sizes obtained from simulation in 0.18um technology are loaded into IPRAIL. The resizing process generates the new layout from the symbolic template and the new device sizes.

1. Load the symmetry information
    a. Click on ‘Symmetry’ on the menu bar
    b. Click on Textual, for textual file input
    c. Select the symmetry file *Cascode/*.sym* or *TwoStage/*.sym*
    d. Click OK
2. Load the device size information
    a. Click on ‘Resize’ on the menu bar
    b. Click on ‘Trans_Size_File’

    c. Select the device size file *Cascode/*.size* or *TwoStage/*.size*
    d. Click OK
3. In order to resize the layout according the new sizes defined in the file
    a. Click on 'Resize' on the menu bar
    b. Click on 'Resize'
4. Exit IPRAIL by clicking the EXIT menu.

At the end of this process, IPRAIL generates the new layout in TSMC 0.18um technology. The layout is exported as a CIF file *sq_out.cif*.

4.3 Checking the Retargeted Layout in IPRAIL

The new layout can be loaded into IPRAIL for viewing.

1. Invoke IPRAIL by typing *iprail* in the Unix command line.
2. Load the IPRAIL's technology file. (*.tch)
    a. Click on 'Load' in the menu bar
    b. Click on 'Technology'
    c. Select the technology file *Technology/MCNC.tch*
    d. Click 'OK'
3. Load the bridge file for the loaded technology
    a. Click on 'Load' in the menu bar
    b. Click on 'BridgeFile'
    c. Select the Bridge File *BridgeFile/bridgeFileTSMC*
    d. Click 'OK'
4. Load the initial design rule
    a. Click on 'Load' in the menu bar
    b. Click on 'InitialDesignRule'
    c. Select the design rule file *DesignRule/UWtsmc18_updated.tech*
    d. Click 'OK'
5. Load the layout CIF file
    a. Click on 'Load' in the menu bar
    b. Click on 'Layout(CIF)'
    c. Select the layout file *sq_out.cif*
    d. Click 'OK'

With this, you should be able to view the generated layout in IPRAIL. The layout is then passed through Design Rule Checking (DRC) in the Cadence Environment.

4.4 Design Rule Checking in Cadence

The CIF file is imported into the Cadence environment and checked for DRC by the following steps. If the cadence paths are different from the ones in *install.csh*, you will need to update the *install.csh*, *cds.lib* and *display.drf*.

1. Go to the directory *demo/cadence* and copy the *sq_out.cif* from *demo/bin*.
2. Execute Cadence layout editor
    a. Type **icfb** on the command line
3. Create a new library if it has not been created.
    b. On the Library Manager menu bar, click on 'File'
    c. Select 'New'
    d. Select 'Library…' and a new window will pop up
    e. Enter the new library name (e.g. *mylibrary*)
    f. Select 'Attach to existing tech library' radio button, and menu box will appear
    g. Select the appropriate technology library (e.g. *Uwtsmc18*)
    h. Hit 'OK'
4. To import the CIF file
    i. On the icfb main window, click 'File'
    j. Select 'import'
    k. Choose 'CIF…' and a "CIF In" window will appear
    l. Click on the 'User – Defined Data' button.
    m. Set the correct Layer Map Table (e.g. */usr/nikola/groups/vlsi/pkgs/cadence/.current/cds/local.18/pipo/cifInLayermap*)
    n. Click 'OK'
    o. Fill in the input file with the complete path of the output cif file (i.e. ~/demo/bin/*sq_out.cif*)
    p. Fill in the Library Name field with the library name that you created
    q. Set the Scale UU/DBU to 0.0100000 micron
    r. Hit 'OK'
5. To open the imported layout
    s. On the Library Manager window, look at the *library* column and double click on the corresponding library name (*e.g. mylibrary*)
    t. In the "Cell" column, single click on '__out__'. This the default name of the resized layout
    u. In the "View" column, double click on *layout*, and the layout will be shown in a new window
6. To run Design Rule Check (DRC)
    v. On the Virtuoso window's menu bar, click 'Verify'
    w. Select 'DRC…'
    x. Make sure the Rules file is correctly setup (e.g. *divaDRC.rul*)
    y. Hit 'OK'
    z. Go to the icfb main window to check the DRC report and errors

After the DRC checking is complete, you should see the following message on the icfb main window

```
********* Summary of rule violation for cell "__out__ layout" *********
  Total errors found: 0
```

4.5 Post-Extraction Simulation

The simulation setup is stored inside the directories *demo/sim/cascode_opamp* and *demo/sim/2stage_opamp*. Each has two subdirectories inside *original_25* and *target_18*. These contain the extracted netlists of the original design (0.25um) and the target generated layout (0.18um). We will run HSPICE simulations to verify that the specifications have been met for either design in both 0.25um and 0.18um technologies. We shall specify the steps for the Cascode opamp. The TwoStage opamp can be simulated similarly.

1. Go to the directory *demo/sim/cascode_opamp/original_25*
2. From the command line execute *hspice cascode_25.sp*
3. Open the GUI by executing *awaves* from the command line.
4. Use the anchor and other measure options to verify the gain, bandwidth, phase_margin and gain_margin of the design.

This verifies the original Cascode layout in TSMC 0.25um technology. Now we shall simulate the layout generated by IPRAIL for TSMC 0.18um technology.

1. Go to the directory *demo/sim/cascode_opamp/target_18*
2. From the command line execute *hspice cascode_18.sp*
3. Open the GUI by executing *awaves* from the command line.
4. Use the anchor and other measure options to verify the gain, bandwidth, phase_margin and gain_margin of the design.

Here is the comparison of the specifications for the Cascode Opamp.

Table 1. Post-Layout Simulation Results for Cascode Opamp

	0.25um TSMC	0.18um TSMC
Gain	60.9 dB	60.6 dB
Bandwidth	51.7 MHz	63.4 MHz
Gain Margin	12 dB	10.5 dB
Phase Margin	63 deg	61 deg

The specifications for the two stage opamp can be verified similarly.

Table 2. Post-Layout Simulation Results for 2Stage Opamp

	0.25um TSMC	0.18um TSMC
Gain	57.6 dB	64.3 dB
Bandwidth	135 MHz	103 MHz
Gain Margin	9.5 dB	9.1dB
Phase Margin	50 deg	56 deg

This concludes the IPRAIL tutorial!

FROSTY: A Fast Hierarchy Extractor for Industrial CMOS Circuits*

Lei Yang and C.-J. Richard Shi
Department of Electrical Engineering, University of Washington
Seattle, WA 98195
{yanglei, cjshi}@ee.washington.edu

Abstract: This paper presents FROSTY, a computer program for automatically extracting the hierarchy of a large-scale digital CMOS circuit from its transistor-level netlist description and a library of subcircuits. To handle the complexity of industrial circuits, FROSTY combines traditional structural recognition and pattern matching methods into a two-step extraction process. First, gate structures based on channel-connected-components are recognized from a circuit netlist and library subcircuits. Then annotated graphs representing the connectivity and properties of gate structures are constructed. Comparing to transistor-level netlists, these graphs are much smaller in size, more distinguishable in structure, and are thus more suitable for labeling based pattern matching. An efficient pattern matching algorithm is applied to extract the circuit hierarchy from these condensed circuit graphs. FROSTY has been demonstrated to be orders of magnitude faster than the best known extraction program SubGemini, capable of extracting the entire hierarchy of industrial designs with several hundred thousand transistors in a few minutes on a Sun workstation. Further FROSTY is scale with the size of a circuit.

1. INTRODUCTION

With the rapid development of IC industry, continuously increasing CMOS circuit complexity poses a great challenge to CAD tools, and makes hierarchical expression of circuits very important. There are several levels of abstractions to represent circuits. *Transistor level* describes circuits through a number of transistors and their interconnections. *Gate level* represents logic gates as building blocks to describe circuits. In digital CMOS designs, there is another higher level of circuit, which includes functional blocks consisting of a number of gates, for example: latch, flip-flop, adder, etc. This block level provides a behavioral description of digital integrated circuits.

Automatic recognition of a high level structure from the transistor level netlist of a circuit design is important for many tasks in VLSI design. The early automatic extractors have been developed mostly for functional verification of a circuit layout with respect to its netlist [1][2]. Later, researchers have also shown how to extract higher level structures to speed up the simulation [3]. If the circuit is described at the transistor level, the simulation time is long compared to a behavior block level simulation. This is extremely useful for post-layout simulation before the tapeout. Hierarchy extraction has also been used in formal verification, as well as circuit diagnosis and test generation [4].

Existing extraction algorithms appeared in literature can be classified to two categories: structural recognition and pattern matching. Structural recognition uses rule-based techniques to identify logic gates from sets of channel-connected MOS transistors [5][6]. This category of algorithms is fast but it can only recognize structures with generic rules, for example, static CMOS gates with complementary structures between p-part and n-part. It cannot handle well irregular-structured blocks, for example, DFF, latches, or high-level blocks with structures that are hard to pre-defined as rules.

Pattern matching based extraction algorithms map a flat circuit to a graph, in which transistors are nodes and interconnection wires are edges. Then a subgraph-isomorphism technique is applied to find a one-to-one correspondence between nodes and edges within the two graphs [7][8][9]. However, finding subcircuits in a transistor level ***object*** circuit is a NP-complete problem and is much slower compared to structural recognition. The complexity of pattern matching is determined by two factors [10]. The first factor is how to construct a discriminative graph labeling algorithm. If the model graph vertices carry unique labels that correspond to the labels of the vertex images in an object graph, then subcircuit recognition is a relatively easy task (*more distinguishable in structure*). Unfortunately, the graphs representing directly the transistor-level netlist are hard to be distinguishable, since both the connectivity and the types of transistors a node connected to can be in-distinguishable for most circuit nodes in digital CMOS circuits. As a result, the construction of a discriminative labeling algorithm is a difficult task. The second factor is how to efficiently find subcircuits in the object circuit. The labeling procedure and the recognition strategy are related and both affect the performance of the subcircuit extraction program.

Some efforts have been dedicated to develop good pattern matching algorithms. SubGemini [9] is one of them. It labels part of the nodes with the node's information as well as its neighbors' information and then performs breadth-first-search in the object graph. SubGemini has been demonstrated to be faster than the previous pattern matching algorithms.

In this paper, we propose to combine structural recognition and pattern matching into a two-step extraction process. In the first step, a structural recognition algorithm is applied to transistor level circuits to extract gate level structures. The second step entails generating a directed graph based on the gate level strcutures. Every node in this graph corresponds to one gate, every edge represents one interconnection wire, and the edge direction stands for the signal flow in a circuit. Then the pattern matching process can be applied to recognize the user-defined behavior blocks.

The proposed two-step process has been implemented into a computer program called FROSTY. It is very fast, due to the following reasons. First, compared with the transistor level pattern matching algorithms, the gate level pattern matching algorithm

* This research was supported by DARPA NeoCAD Program under Grant No. N66001-01-8920 and NSF CAREER Award under Grant No. 9985507.

can significantly reduce the size of the graph because every graph node is a gate instead of a transistor. Secondly, pattern matching of directed graphs (gate level) is faster than undirected graphs (transistor level). Finally, every node in the graph can be labeled according to its gate property, including gate type, the gate logic function, fanout number of gate, number of inputs, etc; this can guarantee most of the nodes in the graph have discriminative labels.

Given a transistor level circuit and a used-defined library file, FROSTY recognizes all CMOS gates and user-defined blocks in the library file and outputs a block level netlist. The design of FROSTY is driven by the observation that for industrial CMOS designs, every design company introduces a considerable amount of its own custom behavioral blocks. If those blocks can be extracted from the transistor level circuit, it will be very helpful for design verification, test generation, and fast simulation. In this paper, we use a set of post layout designs from Boeing's digital CMOS ASIC divisions.

The remainder of this paper is organized as follows. An overview of FROSTY is given in Section 2. Section 3 presents the detailed FROSTY algorithm and its time complexity analysis. Section 4 describes performance results on benchmarks from the Boeing Company. Concluding remarks are made in Section 5.

2. PROGRAM OVERVIEW

FROSTY reads in a transistor level digital CMOS netlist (*object circuit*), and a library file in the SPICE format, as shown in Figure 1. The library file contains user specified subcircuit blocks that are to be recognized from the object circuit. After extraction, FROSTY outputs the object circuit description in terms of standard CMOS logic gates and user-defined blocks in the library using VHDL or Verilog formats. FROSTY also produces a header file that contains the functional definitions of all used standard CMOS gates. Together with VHDL or Verilog model descriptions of the library blocks, the extracted block level netlist and the header file can be used for the post-layout simulation of a transistor-level netlist at higher levels.

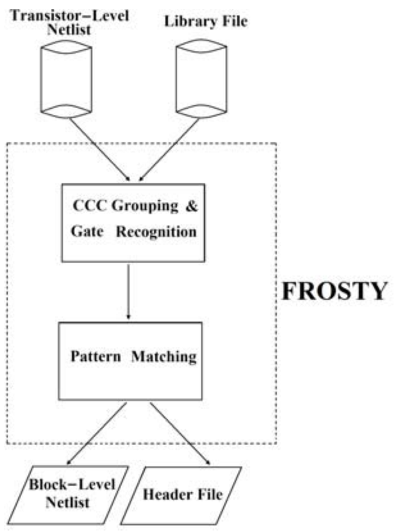


Figure 1. FROSTY flow and architecture.

3. THE TWO-STEP FROSTY ALGORITHM

FROSTY consists of two major steps. The first step is from the transistor level netlist to the gate level, and the second step is from the gate level to the user-defined behavior block level. They are described in 3.1 and 3.2, respectively.

3.1 GATE RECOGNITION

3.1.1 CCC GROUPING

After a netlist is read, the structural recognition algorithm is used for CMOS gate recognition. First, the circuit is divided into Channel-Connected-Components (CCC), which are clusters of transistors connected at the sources and drains. The recognition process starts from every VDD or GND connected transistor, and ends at the connection node between the p-tree and n-tree. All the channel-connected transistors on this path will be grouped together as a p-tree or n-tree. A shared connection node between a p-tree and an n-tree are considered as one CCC. Figure 2 shows the grouping process.

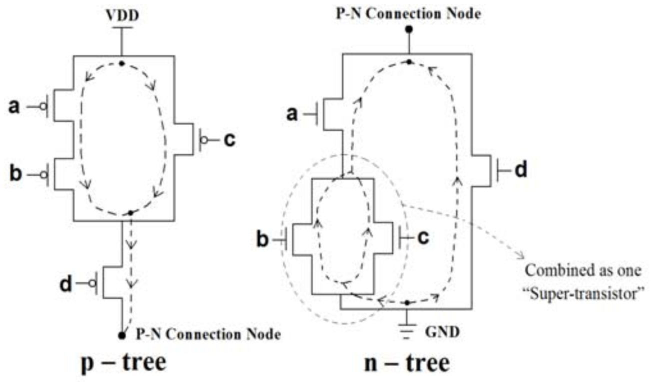


Figure 2. Group the channel-connected blocks.

3.1.2 LOGIC FUNCTION RECOGNITION

For each p-tree and n-tree in the circuit, FROSTY performs the following steps to recognize its logic function. First, FROSTY performs a parallel search inside the p-tree or n-tree. After finding all the transistors that are in parallel, FROSTY replaces them with a "super-transistor", as shown in Figure 2. Then FROSTY performs a serial search, finding all transistors connected in series and replaces them with a "super-transistor". The parallel and series search continues until only one "super-transistor" is left in the tree, at which time the logic function of the tree can be recognized. For example, in Figure 2 the logic function (seen at the P-N connection Node) of this p-tree and n-tree are:

$$f_{p-tree} = (\overline{a} \cdot \overline{b} + \overline{c}) \cdot \overline{d} \qquad f_{n-tree} = a \cdot (b + c) + d$$

Every CCC contains one p-tree and one n-tree. According to the logic function relationships of the p-tree and the n-tree, the CCC can be divided into two types of gates:

- ***Standard Gate***

If a CCC's p-tree and n-tree logic functions have a complementary relationship, the CCC is a standard CMOS gate.

- ***Pseudo Gate***

A CCC whose p-tree and n-tree logic functions are not complementary is called a pseudo gate. Figure 3 is an example of a pseudo gate, one tri-state inverter in a latch. Because the logic functions of the tri-state inverter's p-tree and n-tree do not form a complementary relationship, it is considered to be a pseudo gate.

In static digital CMOS circuits, pseudo gates always exist as part of behavior models (flip-flop, latch…). When all the behavior models have been extracted, there should be no pseudo gates left in the circuit.

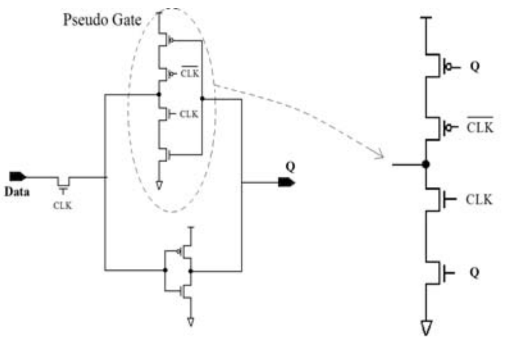


Figure 3. Pseudo CMOS Gate.

Even after structural grouping and logical recognition of the gates, there may be some transistors that cannot be grouped into any CCC. Examples are transistors from pass transistor logic, as shown in Figure 4, which are also recognized by FROSTY.

- ***Pass Transistor Logic***

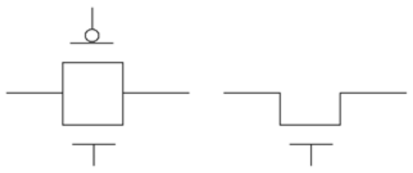


Figure 4. Pass Transistor Logic.

Currently, FROSTY can recognize static digital circuits. After gate recognition is finished, the circuit can be classified into three categories: gates, pseudo gates and pass transistor logic. For dynamic logic circuits, more categories need to be created.

3.2 USER-DEFINED-BLOCK RECOGNITION

In Step 1, the gate level structures are generated for the object circuit and all the blocks in the library. In Step 2, those structures are converted to directed graphs, then a pattern matching algorithm is applied to recognize all the behavior blocks from the object circuit.

3.2.1 DIRECTED GRAPH GENERATION

After Step 1, the circuit has been transformed into a gate-interconnected structure. With each such gate represented by a node, the circuit is then characterized as a graph with both directed edges and un-directed edges.

- ***Directed edge and undirected edge***

A directed edge represents a wire from the output of one gate to the input of another gate, which describes the signal flow in the circuit. For pass transistor logics, it is hard to detect the signal flow. So the wires connected to pass transistor logics can be considered as undirected edges.

- ***Node property***

The characteristics of the gates, such as gate type, logic function of the gate, number of inputs of the gate, gate fanout number, fanout gates properties, etc. are expressed as node properties in the graph. For example, the node that represents the tri-state inverter in Figure 3 has the node property in Table 1.

Table 1. Node property of the pseudo gate in Figure 3.

Gate type		Pseudo gate
Logical function	p-tree	$f = \overline{a} \cdot \overline{b}$
	n-tree	$f = a \cdot b$
Number of inputs		2
Fanout number		2, transmission gate + inverter

Here, let us use the D-flip-flop shown in Figure 5 and transform it from a circuit to a graph. Using the partition and gate recognition algorithm in Step 1, the circuit can be divided into 10 gates. Notice that gate 3 is a pseudo gate made up of two tri-state inverters controlled by the clock signal. Because the two tri-state inverters have the same p-n connection node (the two gates outputs are connected together), the program considers them as one CCC. Gate 10 is a transmission gate, so the edges connected to gate 10 (gate 5 - gate 10, gate 7 - gate 10, gate 8 - gate 10) are undirected edges. Other gates are standard gates.

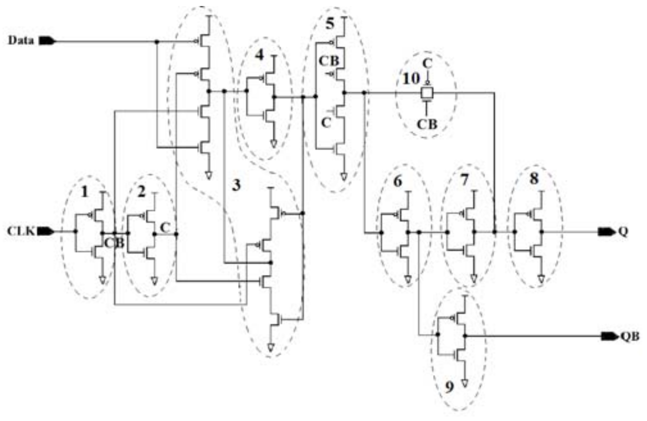
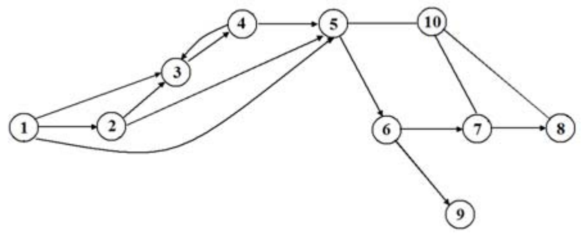


Figure 5. D flip-flop (DFF) circuit after gate recognition.

According to the connection relationships among the gates, a directed graph for this D-flip-flop can be constructed as shown in Figure 6.



$$\begin{tikzcd}
 & & & 4 \arrow[r] \arrow[dl, bend left] & 5 \arrow[rr, dash] \arrow[dr] & & 10 \arrow[d, dash] \arrow[dr, dash] & \\
 & & 3 \arrow[ur] \arrow[urr] & & & 6 \arrow[r] \arrow[dr] & 7 \arrow[r] & 8 \\
1 \arrow[r] \arrow[urr] \arrow[uurrrr, bend right] & 2 \arrow[ur] \arrow[uurrr] & & & & & 9 &
\end{tikzcd}$$

Figure 6. Graph representation of the DFF.

3.2.2 PATTERN MATCHING

After the equivalent graphs are constructed for the object circuit and the library subcircuits, a pattern matching algorithm is employed to locate all of the defined subcircuits in the object circuit.

The basic pattern matching process is illustrated with the following example. Consider the DFF in Figure 5 as a subcircuit block defined in the library file. The object circuit, shown in Figure 7, contains the DFF. The corresponding graph of this circuit is shown in Figure 8. The final pattern matching result is shown in Figure 9.

In order to find the block DFF in the object circuit, we should apply pattern matching to the subgraph (shown in Figure 6) in object graph (shown in Figure 8). This means that for every node in the subgraph, we should find its corresponding node in the object graph. In FROSTY, two node-stacks are set up to hold all the matched nodes, shown in Table 2.

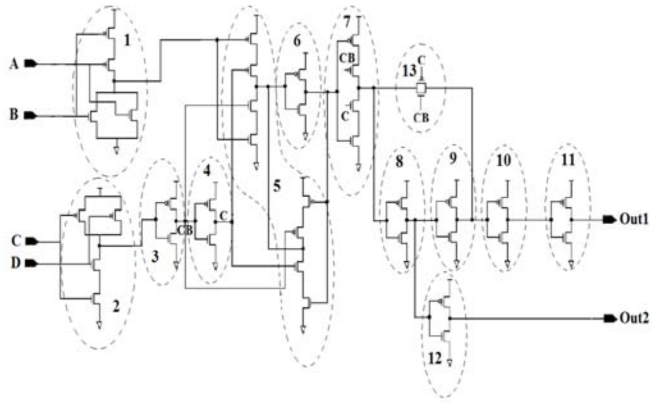


Figure 7. The object circuit after the gate recognition.

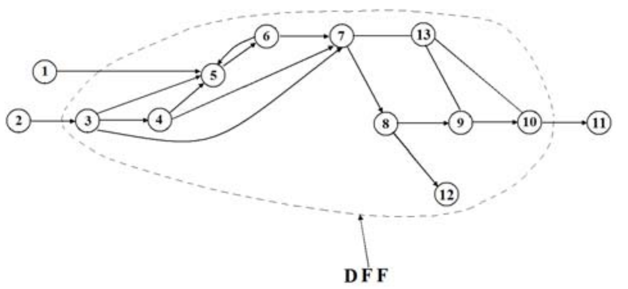


Figure 8. Graph representation of the object circuit.

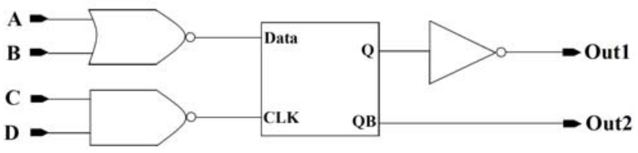


Figure 9. The extracted block-level structure.

The pattern matching process employed in FROSTY is called gradual matching [7]. It begins with a starting node in a block graph, any object graph node with the same "Node Property" as the starting node is identified as a possible location of the subcircuit. Then FROSTY verifies whether there is an actual subcircuit at each possible location.

The first step of the gradual matching process is to locate the starting node in a block graph. From this starting node, all other nodes can be reached through directed or undirected edges. This node is also called "*source node*". In order to locate this source node in the block graph, we pick up a random node first, and then backtrack to its parent nodes. This backtracking is done recursively until a node that has no parent nodes is reached. This node is a "source node". In the DFF block graph in Figure 6, the source node is node 1. However, sometimes we cannot find the source node because the graph may be a ring, as shown in Figure 10. In this case, we can pick any node to be the source node.

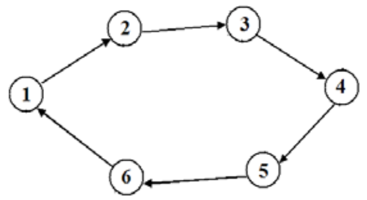


Figure 10. Ring structure of a graph.

After the source node in the block graph is found, all the nodes in the object graph will be searched to locate nodes with similar "Node Properties" as the source node. Any one of these nodes is a possible location of the subcircuit. For every such node (for example, node 3 in Figure 8), the source node and this possible node will be pushed into the block graph node-stack and object graph node-stack, respectively, to begin the gradual matching process; this is Loop 1 in Table 2.

Table 2. Pattern matching process for the example.

Matching process	Block Graph Node-Stack	Object Graph Node-Stack
Loop 1 (source node)	1	3
Loop 2	2	4
	3	5
	5	7
Loop 3	4	6
	6	8
	10	13
Loop 4	9	12
	7	9
	8	10

Then in Loop 2, the matching process starts from this pair of matched nodes in the stacks (node1$_{\text{block graph}}$-node3$_{\text{object graph}}$). In the block graph, node1 connects with nodes 2, 3, and 5, while in object graph node3 connects with nodes 4, 5, and 7. After comparing the "Node Properties" of the two series of nodes, we find the following node pairs, node2$_{\text{block graph}}$-node4$_{\text{object graph}}$, node3$_{\text{block graph}}$-node5$_{\text{object graph}}$, node5$_{\text{block graph}}$-node7$_{\text{object graph}}$, have the same "Node Property", respectively. Those pairs are identified to be matched node pairs and pushed into the stacks.

In Loop 3, the matching process starts from all of the newly matched node pairs in the previous loop. For example, from matched node pair node3$_{\text{block graph}}$-node5$_{\text{object graph}}$ in the stacks, we can match node4$_{\text{block graph}}$-node6$_{\text{object graph}}$; from matched node pair node5$_{\text{block graph}}$-node7$_{\text{object graph}}$, we can match node6$_{\text{block graph}}$-node8$_{\text{object graph}}$, node10$_{\text{block graph}}$-node13$_{\text{object graph}}$; from node pair node2$_{\text{block graph}}$-node4$_{\text{object graph}}$, we can match node5$_{\text{block graph}}$-node7$_{\text{object graph}}$. All these newly found node pairs are also pushed into the stacks.

The process in Loop 3 is iteratively performed until every block graph node matches its corresponding node in object graph. The whole process is shown in Table 2. If any conflict occurs during the gradual matching process, the process fails, and the node-stacks are emptied for next matching process.

In some cases, there may be more than one "source node" in the graph, as shown in Figure 11. In this example, either node 1 or node 2 can be a "source node". For this case, the program picks the starting node randomly. Suppose that node 1 is chosen here, the searching process will be 1-3-5-6-7-8. Since node 2 and 4 cannot be searched, backtracking will be applied. After checking the stack, unmatched node 4 is connected with matched node 6. So backtracking from node 6-4-2 is performed until all the nodes are matched.

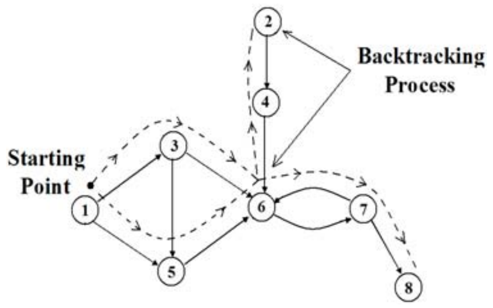


Figure 11. An illustration of the backtracking process.

3.3 OVERALL ALGORITHM AND COMPLEXITY

The entire algorithm of FROSTY is shown in Table 3. In FROSTY, hash tables are used wherever possible due to its linear search time. The complexity of the program is $O(k*n + g1*g2)$, here $k*n$ represents the complexity of Step 1, n is the number of transistors in the circuit, k is an integer number, one can see that the CPU time for Step 1 is linear to n; $g1*g2$ represents the complexity of Step 2, where $g1$ and $g2$ are the number of gates in library file and number of gates in the object circuit after Step 1.

Table 3. Algorithm FROSTY.

```
PREPROCESS FOR LIBRARY:
LOOP: for i = Block 1 : Block n (in library file) {
        Divide the Block_i into channel-connected-components (CCC)
        Recognize pass transistor logic
        Recognize the logic function of every CCC in the Block_i
      }
INITIALIZATION:
Construct hash tables to store transistors and nodes of the circuit
Divide the circuit into channel-connected-components(CCC)
Recognize transmission gates in the left transistors
Recognize the logic function of every CCC
LOOP: for i = Block 1 : Block k (in library file)
{
OuterLoop:
   Find "source node" in the Block_i and push it into node-stack
   InnerLoop:
   for j = Node_1 : Node n (in object graph) {
      if (Node Property (Node i ) = Node Property (Source Node)) {
         Push Source Node and Node i into the node-stack.
         do{ Searching from matched nodes in node-stack to find new matched nodes, and push them into node-stack. }
         while {conflict happens or all nodes in Block_i has been matched}
      }
   }
}
Output the blocks and gates to a Verilog or VHDL block-level netlist
```

4. EXPERIMENTAL RESULTS

FROSTY was written in C++ and executed on SUN Fire V480 server with 900MHz UltraSparc-III processors and 16GB RAM. Results from applying FROSTY to several industrial circuits from Boeing are presented in this section.

Table 4 shows the statistics of a set of test circuits and the FROSTY CPU time for recognizing all the gates and blocks. Test circuits PSM, PSM-7, PSM-17, PSM-43 are a set of digital CMOS designs in Boeing's "Power Supply Monitor ASIC" on F22 airplanes. Test circuits CEGRP, CEGRP-3, CEGRP-5, CEGRP-7, DFGRP, DFGRP-2, DFGRP-4, DFGRP-6 are a set of digital designs in Being's "Pressure Belt Chip". This chip is used in Boeing's airplanes to determine the structural load on aircraft wings by measuring the pressure distributed on the top and bottom of the wing. These test circuits contain a lot of Boeing defined behavior blocks, such as DFFs, latches, MUXs, adders, etc. Table 5 shows in detail how many blocks are contained in the circuits. For example, in PSM, there are 3 different structures of DFFs and the total number of DFFs is 122. Using the library file provided by Boeing, FROSTY extracts all of the blocks in the library file and outputs a behavioral Verilog/VHDL netlist containing the recognized blocks and the standard CMOS gates that do not belong to any block.

Table 5. Blocks types and numbers in test circuits.

		PSM	CEGRP	DFGRP
DFF	Types	3	10	5
	Number	122	1304	1436
Latch	Types	1	-	-
	Number	15	-	-
Adder	Types	-	3	5
	Number	-	118	646
MUX	Types	1	3	4
	Number	27	1049	508
XNOR	Types	-	2	2
	Number	-	290	74

In Table 4, we compare our results with SubGemini [9]. For each test circuit in Table 4, we try to use SubGemini to extract all of the behavior blocks in Boeing's library. However, SubGemini fails to recognize some blocks. The recognized block numbers and CPU time of SubGemini are listed in columns 8 and 9 in Table 4. The CPU time comparison between FROSTY and SubGemini is shown in Figure 12.

Table 4 also shows the speed of FROSTY. For test circuit CEGRP-7 (729652 transistors), recognizing 74998 gates and 19327 behavior blocks only takes 305.32 seconds. FROSTY is faster than SubGemini for larger circuits and libraries because it performs pattern matching at the gate level. For example, it is 20 times faster than SubGemini for the CEGRP-7 circuit.

The CPU time of FROSTY depends on two factors: 1) the size of a circuit and 2) the number of behavior blocks in the library file. To observe the relationship between circuit sizes and CPU times, we use a set of PSM circuits and perform extraction with the same library file. In Figure 13 the relationship between CPU time and circuit sizes is displayed. We can see that FROSTY CPU time is linear to the size of a circuit with the same library file.

Table 4. Results of FROSTY and SubGemini.

Circuits	#Transistors	# CMOS gates	# Behavior blocks	FROSTY CPU Time (s)			SubGemini Results	
				Setup and Step-1	Step 2	Total	# Extracted Behavior blocks	CPU Time (s)
PSM	4520	1516	164	1.17	0.58	1.75	156	1.4
PSM - 7	31640	10612	1148	7.83	3.97	11.8	1091	13.6
PSM - 17	76840	25772	2788	18.98	9.88	28.86	2651	48.4
PSM - 43	194360	65188	7052	49.9	25.2	75.1	6714	277.1
CEGRP	104236	10714	2761	26.45	15.03	41.48	1419	94.1
CEGRP - 3	312708	32142	8283	81.57	46.83	128.4	4568	1113.5
CEGRP - 5	521180	53570	13805	135.0	78.31	213.35	7093	3178.1
CEGRP - 7	729652	74998	19327	191.63	113.67	305.32	9926	5856.2
DFGRP	119257	10048	2664	30.1	45.8	75.9	2001	108.0
DFGRP - 2	238514	20096	5328	60.1	91.5	151.6	3074	1113.9
DFGRP - 4	477028	40192	10656	121.2	169.96	219.2	5652	2456.9
DFGRP - 6	715542	60288	15984	182.45	233.7	416.2	8802	4317.2

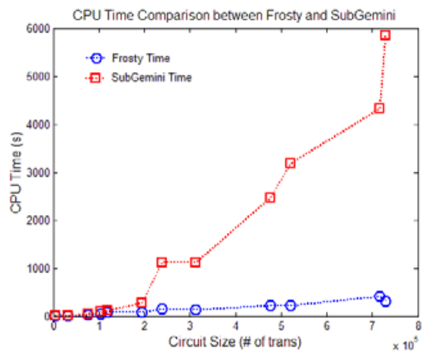


Figure 12. FROSTY and SubGemini CPU time comparison.

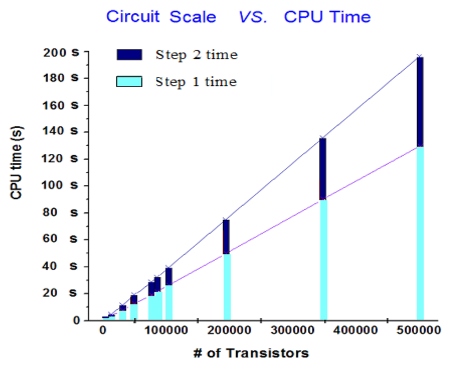


Figure 13. CPU time vs. circuit scale.

5. CONCLUSIONS

This paper presented FROSTY, a computer program for the automatic extraction of circuit hierarchy targeted for the post-layout simulation and verification of library-based large-scale CMOS circuit design. By condensing both the object circuit and library circuits into graphs of blocks and then applying the pattern matching algorithm at the gate level, FROSTY has demonstrated that it can extract an industrial design with seven hundred thousand transistors in less than five minutes on a modern Sun workstation. By representing the extracted hierarchy using high-level descriptions such as VHDL and Verilog, the output netlist can be simulated by any high-level behavioral simulator.

6. REFERENCES

[1] T. Watanabe, M. Endo, and N. Miyahara, "A new automatic logic interconnection verification system for VLSI design", *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems*, vol. CAD-2, no. 2, pp. 70-76, 1982.

[2] M. S. Abadir and J. Ferguson, "An improved layout verification algorithm (LAVA)", *Proc. European Design Automation Conference*, pp. 391-395, 1990.

[3] T. J. Thatcher and R. A. Saleh, "Automatic partitioning and dynamic mixed-mode simulation", *Proc. IEEE Custom Integrated Circuits Conference*, pp. 12.7.1-12.7.4, 1992.

[4] S. Kundu, "GateMaker: A transistor to gate level model extractor for simulation, automatic test pattern generation and verification", *Proc. of International Test Conference*, pp. 372-381, 1998.

[5] M. Boehner, "LOGEX – an automatic logic extractor from transistor to gate level for CMOS technology", *Proc. IEEE/ACM Design Automation Conference*, pp. 517-522, 1988.

[6] A. Lester, P. Bazargan-Sabet and A. Greiner, "YAGLE, a second generation functional abstractor for CMOS VLSI circuits", *Proc. of the Tenth International Conference on Microelectronics,* pp. 265-268, 1998.

[7] F. Luellau, T. Hoepken and E. Barke, "A technology independent block extraction algorithm", *Proc. IEEE/ACM Design Automation Conference*, pp. 610-615, 1984.

[8] G. Pelz and U. Roettcher, "Pattern matching and refinement hybrid approach to circuit comparison", *IEEE Transactions on Computer-Aided Design*, pp. 264-275, vol. 13, no. 2, Feb. 1994.

[9] M. Ohlrich, C. Ebeling and E. Ginting, "SubGemini: Identifying subcircuits using a fast subgraph isomorphism algorithm", *Proc. IEEE/ACM Design Automation Conference*, pp. 31-37, 1993.

[10] N. Rubanov, "SubIslands: The probabilistic match assignment algorithm for subcircuit recognition", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 22, pp. 26-38, Jan. 2003.

A Program for Fast Extraction of High-level Structural Representation from Circuit Description for Industrial CMOS Circuits*

Lei Yang and C.-J. Richard Shi

Mixed-Signal CAD Research Laboratory,
Department of Electrical Engineering,
University of Washington, Seattle, WA 98195, USA
{yanglei, cjshi}@ee.washington.edu

Abstract: This paper presents FROSTY, a computer program for automatically extracting the hierarchy of a large-scale digital CMOS circuit from its transistor-level netlist and a library of subcircuits. To handle the complexity and diversity of industrial circuits, FROSTY combines traditional structural recognition and pattern matching methods into a two-step extraction process. First, gate structures based on channel-connected-components are recognized from a circuit netlist and from all library subcircuits. Then annotated graphs representing the connectivity and properties of gate structures are constructed. Comparing to transistor-level netlists, these graphs are much smaller in size, more distinguishable in structure, and are thus more suitable for labeling based pattern matching. An efficient pattern matching algorithm is applied to extract the circuit hierarchy from these condensed circuit graphs. FROSTY has been demonstrated to be orders of magnitude faster than pattern matching based extraction program SubGemini, and can extract the entire hierarchy of industrial designs with several hundreds of thousands of transistors in a few minutes on a modern Sun workstation. Furthermore FROSTY algorithm is nicely scalable with the size of a circuit.

Keywords: *transistor-level netlist, structural recognitions, pattern matching, channel-connected-components.*

1. Introduction

With the rapid development of semiconductor industry, continuously increasing circuit complexity poses a great challenge to computer-aided-design (CAD) tools, and makes

* This research was supported in part by U.S. Defense Advanced Research Project Agencies (DARPA) NeoCAD Program under Grant N66001-01-1-8920, and in part by the National Science Foundation (NSF) CAREER Award under Grant No. 9985507.

hierarchical expression of circuits very important. Generally there are several levels of abstractions to represent digital circuits. *Transistor level* describes circuits in terms of transistors and their interconnections. *Gate level* uses logic gates as building blocks to describe circuits. *Functional level* description is composed of functional blocks such as latches, flip-flops, adders, etc; each may contain a number of gates.

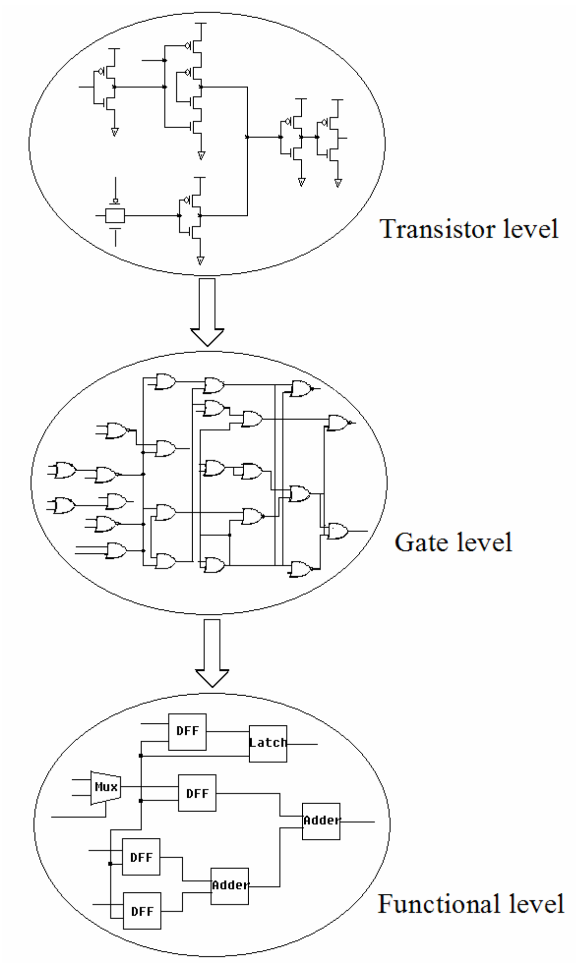


Figure 1. Hierarchical abstractions of digital circuits.

Automatic recognition of a high level structure from the transistor level netlist of a circuit is important for many tasks in very large scale integrated (VLSI) circuit design. Early automatic extractors were mostly targeted for functional verification of a circuit layout with respect to its netlist [1]. Later, researchers have shown how to extract higher level structures to speed up circuit simulation [2][3]. Other researchers[4][5] have presented the application of subcircuit recognition to high level design-for-testability (DFT) and built-in-self-test (BIST) based on the idea that registers can be selected as test resources such as test pattern generators, scan registers and test response compactors, which can reduce the complexity of making a testable design. Hierarchy extraction has also been used in formal verification, as well as floorplanning and logic resynthesis[6].

Existing subcircuit extraction algorithms appeared in literature can be classified into two categories: structural recognition and pattern matching. Due to the well defined structure of MOS circuits, efficient structural recognition algorithms can be used to identify gates in circuits [7][8]. The idea behind these algorithms is that CMOS circuits can be divided into channel-connected-components (**CCC**). Figure 2 shows an example of a static CMOS circuit partitioned into CCCs, and every CCC is a static CMOS gate. Because of the parallel and serial connection among the transistors inside a CCC, the logic function of the CCC can be easily determined. In NMOS and dynamic CMOS circuits, employing a similar idea, the basic NMOS gates and dynamic CMOS gates can also be extracted. In general, structural recognition algorithms use rule-based techniques to identify logic gates, and are fast but can only recognize structures with generic rules. Generally structural recognition algorithms cannot handle well irregular-structured blocks, for example, DFF, latches, other high-level digital blocks, or current mirrors, amplifiers, and other analog blocks with structures that are hard to be pre-defined in terms of rules.

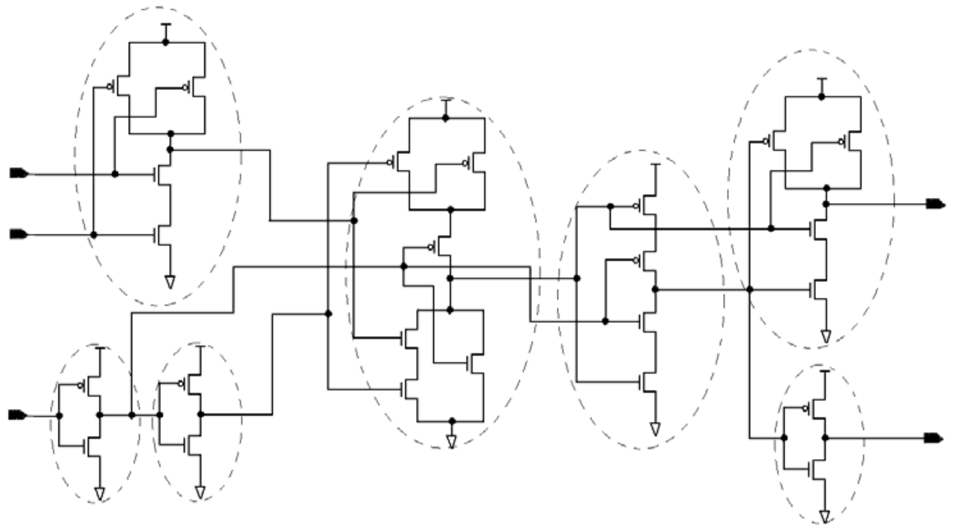


Figure 2. A static CMOS circuit composed of channel-connected-components (CCC).

Due to the limitation of structural recognition algorithms, many researchers turn to technology independent pattern matching algorithms to recognize subcircuits in a netlist [9][10][11][12]. Pattern matching of subcircuits in an object circuit is actually a subgraph isomorphism problem, in which subcircuits and the ***object*** circuit are converted to bipartite graphs. In such graphs, devices and nets are vertices and terminal connections are edges. Subgraph-isomorphism techniques are applied to find one-to-one correspondences between vertices and edges within the subgraphs and the object graph. However, the subgraph isomorphism is a NP-complete problem, which might require exponential time in most general cases. The complexity of pattern matching is determined by two factors [13]. The first factor is how to construct a discriminative graph labeling algorithm. It is obvious that if the subgraph vertices carry unique labels that correspond to the labels of the vertex images in an object graph, subcircuit recognition is a relatively easy task (*more distinguishable in structure*). Unfortunately, the graphs representing directly transistor-level netlists are hard to be distinguishable. Most subgraph isomorphism algorithms exploit the structure of local vertex connections to form labels, leading to poor discriminative capabilities because the graph can only be divided into several groups if vertex invariants such as device type and net degree are used to partition the vertices. Some other algorithms attempt to take into account the non-local vertex surrounding structures. They have better discriminative capabilities but still suffer from forming reliable labels for the vertices representing nets, especially for subcircuit external nets and their images in the object circuit.

The second factor that affects the complexity of pattern matching algorithm is how to efficiently find subcircuits in an object circuit in a rigorous mathematical way. Many algorithms have been suggested to solve this problem. They can be classified into two groups according to [13]. The first group works by searching subgraphs in the object graph [9][10][11][12]. In those algorithms, many technologies such as safe labeling and searching space reduction methods are utilized. However, these algorithms are slow for large and symmetric graphs. The second group employs optimization strategies, for example, graduated assignment [14], relaxation labeling [15], etc. This group of algorithms has reasonable searching time, but the accuracy may suffer, depending on the discriminability of the graph labeling.

Much research efforts have been dedicated to finding good pattern matching algorithms. SubGemini [12] and SubIslands [13] are two successful ones. Both of the them attempt to form as discriminative labels as possible through considering not only the local vertex characteristics, but also its surrounding neighbor. SubGemini then performs a breadth first search procedure in a reduced space to find subcircuits, while SubIslands exploits optimization strategy by constructing a probabilistic match matrix to recognize all the subcircuit images simultaneously.

In practice, it is very usefulto have a CAD tool that can extract all the gates and functional blocks in a digital CMOS design. However, since pattern matching based program such as SubGemini is technology independent, users must spend much time to select and specify all the subcircuits in a library in order to extract them. Generally, digital CMOS designs contain lots of standard gates such as inverters, NAND and NOR gates. For every single design, much time has to be spent on defining the structures of standard gates. A better choice is to define an encyclopedical library for all the designs, but CPU time is wasted in that case because programs try to extract the subcircuits that exist in the library, but may not exist in the design at all. Considering those disadvantages, structural recognition algorithms are more advantageous than pattern matching algorithms for extracting digital CMOS design. On one hand, structural recognition process is much faster. On the other hand, structural recognition programs can automatically recognize all standard gates and abstract their logic functions without having to define them in a library.

Besides standard CMOS gates, in digital CMOS designs, there also exist many functional blocks such as flip-flops, latches, adders that are composed of several CCCs. To extract those irregular-structured blocks, pattern matching is the only choice. We can certainly exploit transistor-level subgraph isomorphism to extract them. However, since we have all the CCCs partitioned and gate functions abstracted after performing structural recognition techniques, a much better approach is to do the pattern matching based on the gate level graphs instead of the transistor level.

Based on the above idea, we propose to **combine structural recognition and pattern matching into a two-step extraction process for static CMOS circuits.** In the first step, the structural recognition algorithm is applied to transistor level circuits to extract gate level structures. The second step involves generating a directed graph based on the gate level structures. Every vertex in this graph corresponds to a gate, every edge represents one interconnection wire, and the edge direction stands for the signal flow in the circuit. Then the pattern matching process can be applied to recognize the user-defined behavior blocks.

The proposed two-step process has been implemented into a computer program called FROSTY which works on static CMOS designs. FROSTY is very fast, due to the following reasons. First, compared with the transistor level pattern matching algorithms [9][10][11][12][13], the gate level pattern matching algorithm can significantly reduce the size of the graph because every graph vertex is a gate instead of a transistor or a net. Second, pattern matching of directed graphs (at the gate level) is faster than undirected graphs (at the transistor level). Finally, every vertex in the graph is labeled based on its local vertex characteristics as well as surrouding information. The local characteristics include: 1) the type of the gate; 2) the logic function of the gate; 3) the number of inputs of the gate; The surrounding information contains: 1) gate fanin number; 2) gate fanout number; 3) gate properties in the fanin stage; 4) gate properties in the fanout stage. The above vertex labeling algorithm can result in good *local discriminability* for fast pattern matching process (the concept of local discriminability is discussed later in the paper).

Given a transistor level static CMOS circuit and a user-defined library file, FROSTY recognizes all CMOS gates and user-defined blocks in the library and outputs a functional level netlist. The design of FROSTY is driven by the observation that for industrial CMOS designs, every design company introduces a considerable number of its own custom functional blocks. If those blocks can be extracted from transistor level circuits, it will be very helpful for design verification, test generation, and fast simulation. In this paper, all the test circuits are post layout designs from Boeing's digital CMOS ASIC division.

The remainder of this paper is organized as follows. An overview of FROSTY is given in Section 2. Section 3 describes the details of FROSTY algorithm and its time complexity analysis. Discussion of the algorithm is given in section 4. Section 5 describes performance results on benchmarks from Boeing. Concluding remarks are given in Section 6.

2. Program Overview

FROSTY reads in a transistor level static CMOS netlist (*object circuit*), and a library file in the SPICE format, as shown in Figure 3. The library file contains user-specified subcircuit blocks that are to be recognized from the object circuit. After extraction, FROSTY outputs the VHDL or Verilog format object circuit description in terms of standard CMOS logic gates and user-defined blocks in the library. FROSTY also produces a header file that contains the functional definitions of all used standard CMOS gates. Together with VHDL or Verilog model descriptions of the library blocks, the extracted block level netlist and the header file can be used for the high level post-layout simulation of a transistor-level netlist.

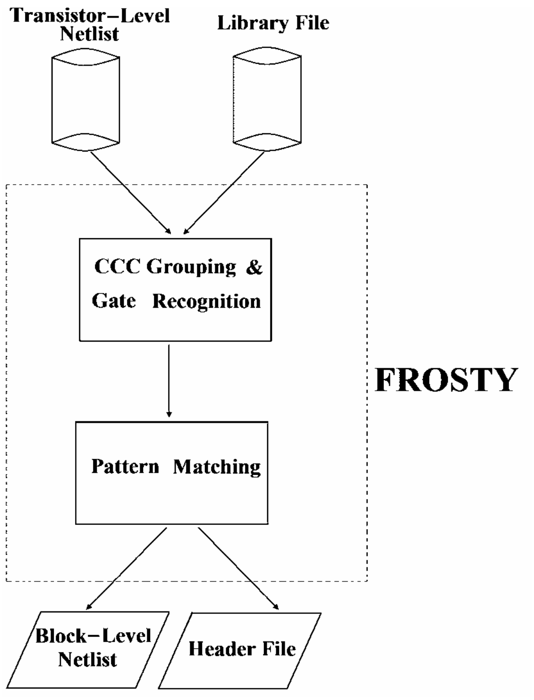


Figure 3. FROSTY flow and architecture.

3. FROSTY Algorithm

FROSTY consists of two major steps. The first step is to extract logic gates from the transistor level netlist. The second step is to perform gate-level pattern matching to recognize higher level library-defined behavior blocks. Below, both steps are described in more details.

3.1 Preliminaries and Terminology

After reading in a subcircuit and an object circuit netlist, bipartite graphs are generated for them. Here we use $G(V, E)$ to represent a graph, V is the set of vertices and E is the set of edges. In the bipartite graph, $V=A \cup B$, where A consist of vertices corresponding to transistors, B consist of vertices corresponding to nets. Each edge $e \in E$ has one endpoint in A, one endpoint in B, and A and B are disjoint sets.

Notation used in this paper is as follows. Symbol t is used to represent a transistor vertex, n is used to represent a net vertex, and e represents an edge. Vertices n and t are said to be adjacent if the set(n, t) is an edge. For a bipartite graph, the set of neighbors N_t of a transistor vertex t is the set of net vertices that are adjacent to t, and the neighbor set N_n of a net vertex n is the set of transistor vertices that are adjacent to n. Among all transistor vertices in N_n, some of them are connected with n through source or drain terminals, and they are grouped into a channel-connected-neighbor-set CN_n, while other transistor vertices connect to n through gate terminals, and they are grouped into a gate-connected-neighbor-set GN_n. Clearly, $N_n = CN_n \cup GN_n$. Generally, in the graph every transistor vertex *t* has three terminals, source, drain and gate. If they connect with vertex n_1, n_2, n_3 separately, we call n_1, n_2 channel-symmetric pair of *t* ; n_1, n_3 or n_2, n_3 are gate-channel pairs of *t*. If net vertex *n* connects to *t* through source or drain terminal, we call *t* channel-connected with *n*. If *n* connects to *t* through gate terminal, we call *t* gate-connected with *n*.

In the process of structure recognition, transistor vertices and net vertices are partitioned based on their local characteristics. Partition of transistor vertices is straightforward: a transistor vertex is either a PMOS vertex or a NMOS vertex. Net vertices can be partitioned into the following groups:

- ***Power vertices***

If a net vertex is power or ground, it is a power vertex. For example, vertex VDD and GND in Figure 4 are power vertices. In static CMOS circuits, all the transistor vertices channel-connected with VDD constitute set V_{vdd}, and all the transistor vertices channel-connected with GND constitute set V_{gnd}. FROSTY needs user to specify the VDD net and GND net in the input netlist.

- ***PN vertices***

Suppose CN_n is the channel-connected-neighbor-set of a net vertex *n*. If there exists at least one PMOS transistor vertex and one NMOS transistor vertex in CN_n, this net vertex *n* is defined as a PN vertex. For example, vertex 3 in Figure 4 is a PN vertex, normally PN vertices are the output vertices of static CMOS gates.

- ***Internal vertices***

If the gate-connected-neighbor-set GN_n of a net vertex *n* is empty (that means: $N_n = CN_n$), and all transistor vertices in CN_n are either PMOS transistors or NMOS transistors, the vertex *n* is a internal vertex. For example, vertex 1, 2 and 4 in Figure 4 are internal vertices.

- ***I/O vertices***

The input and output net vertices of a subcircuit or an object circuit are I/O vertices. For example, vertex a, b, c and d in Figure 4 are I/O vertices.

Note that a net vertex does not necessarily belong to only one of the above groups. For example, a power vertex can also be an I/O vertex.

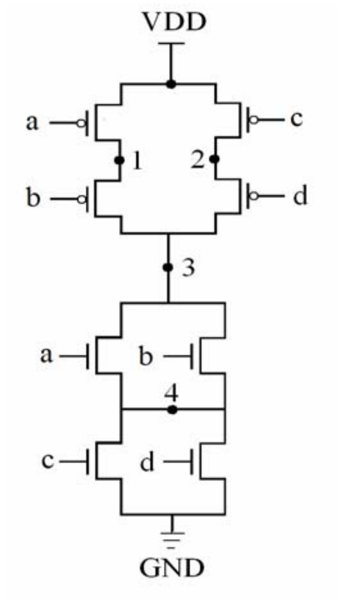


Figure 4. Different kinds of net vertices in a static CMOS gate.

3.2 Gate Recognition

3.2.1 CCC Grouping

After bipartite graphs are generated, structural recognition algorithm is used to extract logic gates from the transistor netlist. Transistor structures that are eligible for replacement by logic gates are Channel-Connected-Components (CCC), and each CCC should contain one p-tree and one n-tree in the static CMOS design. Every p-tree or n-tree has only one PN vertex, and the p-tree and n-tree that share the same PN vertex constitute a CCC. In FROSTY, a tree finding algorithm is employed to identify all n-trees and p-trees, and the algorithm is addressed in the following illustrated with an example shown in Figure 5.

To find a new n-tree, FROSTY allocates two linklists $L_{new\text{-}n\text{-}tree}$ and $L_{new\text{-}net}$. Linklist $L_{new\text{-}n\text{-}tree}$ holds all transistor vertices in this n-tree, and $L_{new\text{-}net}$ holds all newly found internal vertices in every step of the tree finding process. In the example shown in Figure 5, it takes four steps to find all transistors in the n-tree.

Step 1:

The algorithm starts from every transistor vertex *t* in the graph that is channel connected with the *GND* vertex (see algorithm line 3 in Figure 6). Here we assume this starting transistor is

m_1. So in step 1, being a member of the newly found tree, m_1 is put into $L_{new\text{-}n\text{-}tree}$, and *GND*'s channel-symmetric vertex n_1 is put into $L_{new\text{-}net}$.

Step 2:

In step 2, the searching process checks every vertex n in $L_{new\text{-}net}$ to find new transistors channel-connected with n (see line 13 in Figure 6). Since m_2 is channel-connected with n_1, m_2 is put into $L_{new\text{-}n\text{-}tree}$ as one part of the n-tree, and the newly found vertex that is put in $L_{new\text{-}net}$ is n_2, which is channel-symmetric with n_1. Meanwhile, $L_{new\text{-}net}$ is updated to delete the old vertex n_1 at step 2.

Step 3:

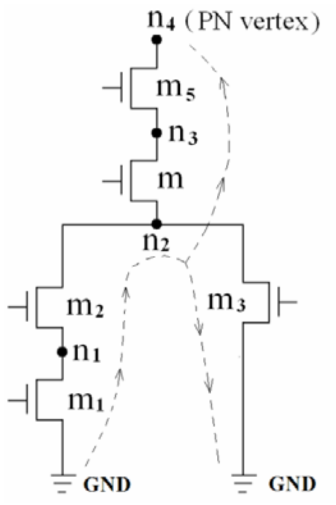
In step 3, searching from n_2, two new transistor vertices m_3 and m_4 channel-connected with n_2 are found and inserted into $L_{new\text{-}n\text{-}tree}$. Also two new net vertices n_3 and *GND* that are channel-symmetric with n_2 are found. Before putting n_3 and *GND* into $L_{new\text{-}net}$, there is a decision step (see line 16 of Figure 6) to make sure those net vertices are not PN vertices or GND vertices. So GND vertex is denied and only n_3 is put into $L_{new\text{-}net}$ and the old vertex n_2 is deleted to update $L_{new\text{-}net}$.

Step 4:

In step 4, from vertex n_3 program finds another new transistor m_5 and it is put in $L_{new\text{-}n\text{-}tree}$; Because the newly found channel-symmetric vertex n_4 is a PN vertex, it is not put into $L_{new\text{-}net}$. Now $L_{new\text{-}net}$ becomes empty after updating (delete the old vertex n_3), this finishes the searching process for one n-tree (decided by line 10 in Figure 6).

After the above four steps, a new n-tree is generated, which contains all the transistors stored in $L_{new\text{-}n\text{-}tree}$ and has one PN vertex n_4. $L_{new\text{-}n\text{-}tree}$ is inserted into the whole circuit n-tree linklist $L_{n\text{-}tree}$, and all transistor vertices contained in $L_{new\text{-}n\text{-}tree}$ are deleted from the the whole circuit transistor vertices linklist $L_{transistor}$.

The n-tree searching process starts from every GND channel-connected transistor and stops at the PN vertices and GND vertices. All the transistors belonging to this searching path constitute a n-tree. After finishing finding all the n-trees and stores them in the linklist $L_{n\text{-}tree}$, FROSTY exploits a similar algorithm to find all the p-trees in the circuit and stores them in the linklist $L_{p\text{-}tree}$. Every n-tree in $L_{n\text{-}tree}$ corresponds to a p-tree in $L_{p\text{-}tree}$, they share with the same PN vertex. This n-tree and p-tree are combined to a CCC.



	$L_{new\text{-}net}$	$L_{one\text{-}n\text{-}tree}$
1	n_1	m_1
2	n_2	m_1 m_2
3	n_3	m_1 m_2 m_3 m_4
4	empty	m_1 m_2 m_3 m_4 m_5

Figure 5. N-tree path finding example.

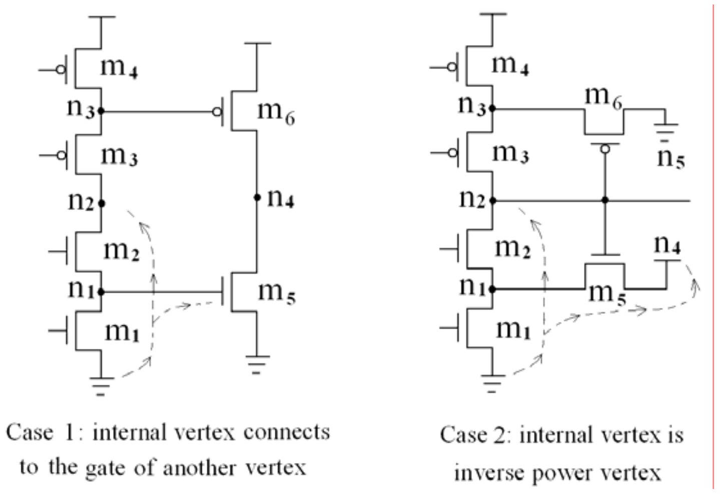
n-tree finding algorithm

```
1.  Allocate a linklist $L_{n\text{-}tree}$ to hold all n-trees
2.  Allocate a linklist $L_{new\text{-}net}$ to hold new found internal net vertices on every step.
3.  For each transistor vertex $t_o \in V_{gnd}$
4.      Allocate a linklist $L_{new\text{-}n\text{-}tree}$ to hold all transistor vertices in a n-tree
5.      Empty $L_{new\text{-}net} = NULL$
6.      Find $n_o$ (vertex $n_o$ and $GND$ are channel-symmetric pair of $t_o$)
7.      Insert $t_o$ in $L_{new\text{-}n\text{-}tree}$
8.      If (is $n_o$ not a PN vertex)
9.          Insert $n_o$ in $L_{new\text{-}net}$
10.         While($L_{new\text{-}net}$!= empty)
11.             For each $n \in L_{new\text{-}net}$
12.                 Delete n from $L_{new\text{-}net}$
13.                 For each $t \in CN_n$
14.                     If (t $\notin L_{new\text{-}n\text{-}tree}$)
15.                         Find net $n_2$, $n_2$ and $n$ are channel-symmetric of $t$
16.                         If ($n_2$ != PN vertex && $n_2$!=GND)
17.                             If ($n_2 != VDD$ && $GN_n = NULL$)
18.                                 Insert $n_2$ into $L_{new\text{-}net}$
19.                             else
20.                                 N-tree finding failure
21.                             End if
22.                         Elseif ($n_2$=GND)
23.                             Delete $t$ from $V_{gnd}$
24.                         End if
```

```
25.                         End if
26.                     End for
27.                 End for
28.             End while
29.         End if
30. End for
31. Delete all transistor vertices from transistor vertex linklist $L_{transistors}$
32. Insert $L_{new\text{-}n\text{-}tree}$ into $L_{n\text{-}tree}$
```

Figure 6. N-tree path finding algorithm.

From the above n-tree searching algorithm, we can see that newly found vertices every step in $L_{new\text{-}net}$ are either internal vertices, PN vertices or GND vertices. If this is not the case, n-tree finding process fails. For example, in the case 1 of Figure 7, the newly found vertex n_1 is not a PN vertex, nor a internal vertex (from the definition of a internal vertex, we can see the internal vertex is forbidden to gate-connect with other transistors), causing the search to fail. Structures like case 1 always appear in dynamic circuits, but seldom in static CMOS circuits. Case 2 of Figure 7 is a Schmitt trigger. During the n-tree searching process, the newly found vertex n_4 is is VDD, this also causes the search to fail. For the above two cases, no CCC is generated due to the tree finding failure.



m4
n3
m6
m3
n2
n4
m2
n1
m5
m1
Case 1: internal vertex connects to the gate of another vertex
m4
n3
m6
n5
m3
n2
n4
m2
n1
m5
m1
Case 2: internal vertex is inverse power vertex

Figure 7. Two failure cases for tree findiing algorithm

If subcircuits shown in Figure 7 exist in the library, two graphs are generated for the object circuit. The first one is a gate level graph, which is composed of partitioned CCC blocks, every CCC representing a vertex. The second graph is a transistor-level bipartite graph, which is composed of the transistors that are unable to be grouped into CCCs. FROSTY first recognizes all the subcircuits that can be partitioned into CCCs in the first graph, and then performs SubGemini algorithm to recognize those subcircuits like Figure 7 from the second graph. In static CMOS circuits, we assume there are not so many blocks like Figure 7, so the transistor-level object bipartite graph should be very small, making the SubGemini recognition process pretty fast. FROSTY algorithm is suitable for CMOS circuits that can be divided into different blocks of CCCs (recognize subcircuits in the first graph). The case when the object circuit is made up mainly of blocks that cannot be partitioned into CCCs, lies beyond the scope of our discussion.

3.2.2 Pass Transistors Identification

After all CCCs in the object graph and subgraphs have been recognized, FROSTY tries to identify pass transistors. One transistor is considered to be a pass transistor if it belongs to one of the following categories:

1) One channel terminal (source or drain) connects to the the PN vertex of a CCC, the other channel terminal connects to a net vertex whose gate-connected-neighbor-set GN_n *!= empty*.

2) One channel terminal connects to the PN vertex of a CCC, the other channel terminal connects to a output vertex.

3) One channel terminal connects to a input vertex, the other channel terminal connects to a net vertex whose gate-connected-neighbor-set GN_n *!= empty*.

The identified pass transistor logic are partitioned into 3 groups: PMOS pass transistor, NMOS pass transistor and T-gate (shown in Figure 8).

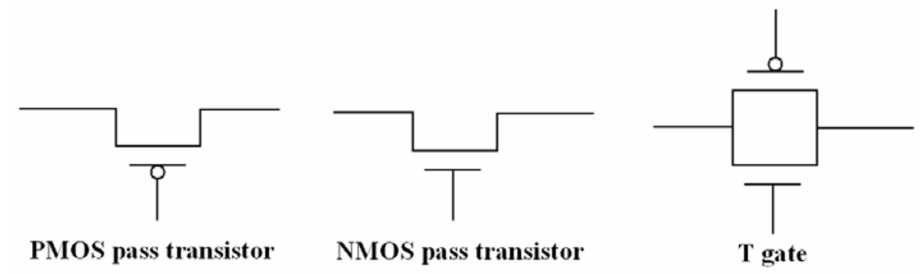


Figure 8. PMOS pass transistor, NMOS pass transistor, T gate.

3.2.3 Logic Function Recognition

For each p-tree and n-tree in the circuit, FROSTY performs a parallel-serial-searching algorithm to abstract its logic function. An example is illustrated in Figure 9 to show the parallel-serial-search process. First, FROSTY performs a parallel search inside the p-tree/n-tree. After finding the transistors that are in parallel, FROSTY replaces them with a "super-transistor". This “super-transistor” represents “OR” relationship among those gates. Then FROSTY does a serial search, finding all transistors connected in serial and replaces them with a “super-transistor”, which represents “AND” relationship among the transistors. The parallel and serial search continues until only one “super-transistor” is left in the tree, at which time the logic function of the tree is recognized. The logic function recognition process for the p-tree and n-tree in Figure 9 is shown in table 1.

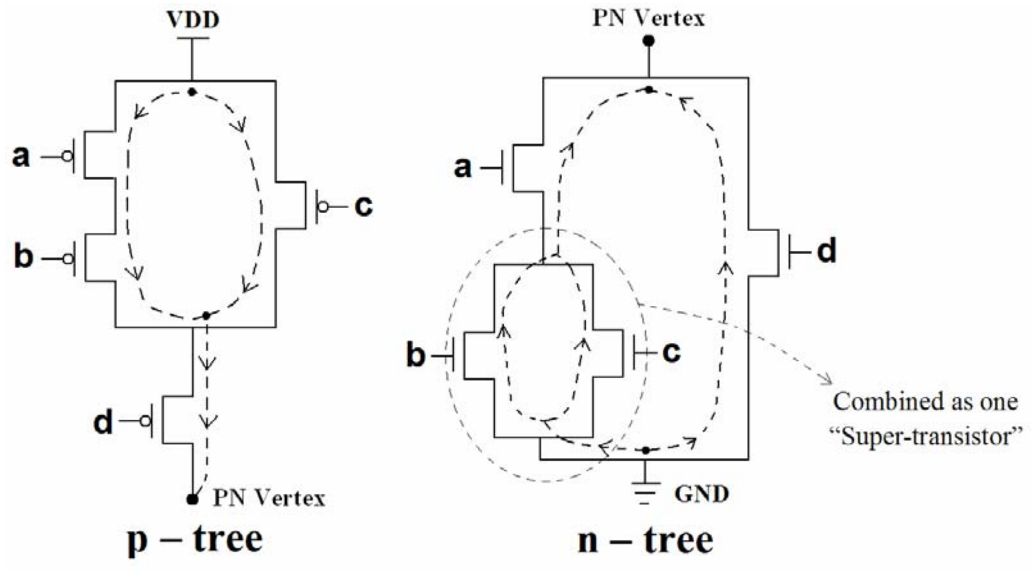


Figure 9. Group the channel-connected blocks.

Table 1. Logic function recognition process of the p-tree and n-tree in Figure 9.

Process	p-tree	n-tree
step-1 (parallel search)	_	$b+c$
step-2 (series search)	$\overline{a}\cdot\overline{b}$	$a\cdot(b+c)$
step-3 (parallel search)	$\overline{a}\cdot\overline{b}+\overline{c}$	$a\cdot(b+c)+d$
step-4 (series search)	$(\overline{a}\cdot\overline{b}+\overline{c})\cdot\overline{d}$	_

After the above parallel-serial-searching process, the logic functions of the p-tree and n-tree in Figure 9 are:

$$f_{p-tree} = (\overline{a}\cdot\overline{b}+\overline{c})\cdot\overline{d}$$

$$f_{n-tree} = a\cdot(b+c)+d$$

In static CMOS design, every CCC contains exactly one p-tree and one n-tree. According to the logic function relationships of the p-tree and the n-tree, the CCC can be divided into two types of gates:

- ***Standard Gates***

If a CCC's p-tree and n-tree logic functions have a complementary relationship, as shown below.

$$f_{p-tree} = \overline{f_{n-tree}}$$

Then the CCC is a standard CMOS gate. For example, in Figure 10, the logic function of the p-tree is $\bar{a} \cdot \bar{b} + \bar{c}$, while the n-tree is $(a+b) \cdot c$. They are complementary with each other. So it is a standard OAI12 gate.

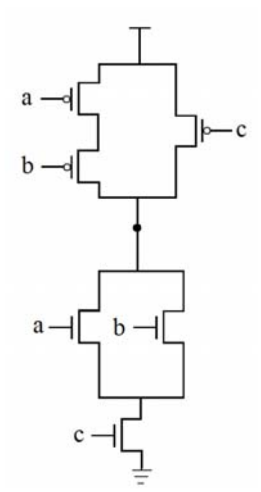


Figure 10. Standard OAI12 CMOS gate.

- ***Pseudo Gates***

A pseudo gate is defined as a CCC whose p-tree and n-tree logic functions are not complementary with each other. Figure 11 gives an example of a pseudo gate, one tri-state inverter in a latch. Because the logic functions of the tri-state inverter's p-tree and n-tree do not have a complementary relationship, it is considered to be a pseudo gate.

For some extreme case, a CCC may only contains only a p-tree/n-tree. For example, one PMOS charge transistor in SRAM circuits constitute a p-tree, without the corresponding n-tree, it is still considered a pseudo gate.

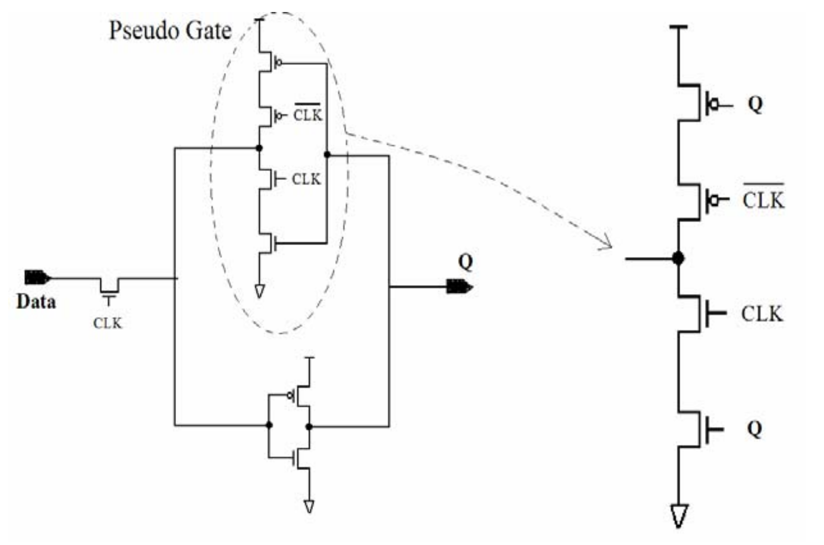


Figure 11. Pseudo CMOS gate.

Although pseudo gates do not independently exist in static CMOS circuits (it always appears as part of the functional blocks), FROSTY accepts it as the basic element to construct a gate level graph. If after extraction of all the subcircuits in the library, there are still pseudo gates left in the object circuit, that means FROSTY may fail to find some subcircuits in the object circuit and an error report will be given. For example, sometimes in the object circuit, the input/output ports of the subcircuit images connect to VDD or GND, causing recognition to fail.

After step 1 ends, all the gates and pass transistors are found. Sometimes structures (like Figure 7) that can not be partitioned into gates also exist in the object circuit. In that case, they are stored as ungrouped transistors. So, after step 1, the circuit is composed of gates, pass transistors and ungrouped transistors. Currently, FROSTY only works on static CMOS circuits. The future work may extend to dynamic circuits, which means more categories need to be created after step 1.

3.3 User Defined Block Recognition

In step 1, the gate level structures are generated for the object circuit and all the subcircuits in the library. In step 2, those structures are converted to directed graphs, then a pattern matching algorithm is applied to recognize all the behavior blocks from the graph.

3.3.1 Directed Graph Generation

After the recognition process in step 1, the object circuit is composed of gates, pass transistors and ungrouped transistors. With each gate or pass transistor represented by a vertex, the object circuit is then characterized as a graph with both directed edges and un-directed edges.

The ungrouped transistors do not appear in the construction of the gate-level graph, and those ungrouped transistors constitute another transistor-level object graph. As stated before, because in the static CMOS circuit there may only be very few blocks that are unable to be partitioned into CCCs, the transistor-level graph is very small and SubGemini algorithm is performed to recognized those blocks (this part of work is not discussed in this paper).

- ***Directed edge and undirected edge***

A directed edge is used to represent a wire from the output of one gate to the input of another gate, which describes the signal flow in the circuit. If one end of an edge connects to a pass transistor, the direction of the edge is determined by the connectionship of the other end: 1) If the other end connects to the PN vertex of a CCC or an input vertex, the signal direction is from the PN vertex or the input vertex to the pass transistor; 2) If the other end connects to the input of a CCC or an output vertex, the signal direction is from the pass transistor to the CCC or the output vertex; 3) If the other end connects to another pass transistor, it is hard to determine the signal flow, this edge is an undirected edge.

- ***Vertex labeling***

Vertex labeling is important for subgraph isomorphism problem. In FROSTY, we exploit vertex local characteristics as well as none-local informations to label vertices. Since every vertex is a gate in the graph, the local characteristics of the gate include: 1) the type of the gate (standard gate, pseudo gate, pass gate); 2) the logic function of the p/n-tree; 3) the number of input of the gate. Vertex surrounding information involves: 1) fanin number; 2) fanout number; 3) gate properties in the fanin stage; 4) gate properties in the fanout stage. As an example, the vertex that represents gate 4 in Figure 12 has the following vertex label in Table 2.

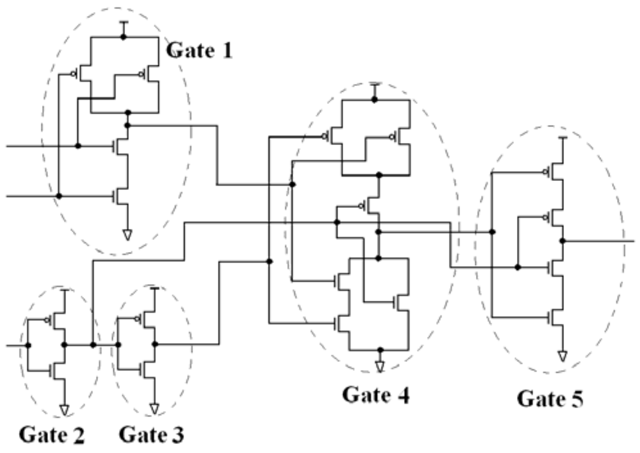


Figure 12. An example of labeling gate 4.

Table 2. Vertex label of gate 4 in Figure 12.

Local Property	gate type		standard gate
	number of inputs		3
	logical function	p-tree	$f = (\overline{a} + \overline{b}) \cdot \overline{c}$
		n-tree	$f = a \cdot b + c$
Surrounding Property	gate in the fanout stage	gate 5	gate type: pseudo gate num of inputs: 2 p-tree: $f = \overline{a} \cdot \overline{b}$ n-tree: $f = a \cdot b$
	gates in the fanin stage	gate 1	gate type: standard gate num of inputs: 2 p-tree: $f = \overline{a} + \overline{b}$ n-tree: $f = a \cdot b$
		gate 2	gate type: standard gate num of inputs: 1 p-tree: $f = \overline{a}$ n-tree: $f = a$
		Gate 3	gate type: standard gate num of inputs: 1 p-tree: $f = \overline{a}$ n-tree: $f = a$

Although this labeling algorithm can not guarantee totally discriminative level in the gate-level graph, it does achieve ***local discriminativity*** that is good enough for recognition subcircuits. To understand the meaning of local discriminativity, just imagine that in each step of the matching process of the gate-level directed graph, program starts from an already matched vertex and gets a group of its fanout vertices for next step matching. Generally, in CMOS circuits, a gate has at most 4-5 fanout gates (normally 1-3), so among this group of fanout vertices, our labeling algorithm can have good discriminativity to distinguish them, making the matching process smooth. If an ambiguity arises due to the symmetric structure of the subcircuit (making the vertex labeling same), FROSTY chooses any of the symmetric vertices, guesses a initial match, and then continues the pattern matching process. If the match fails, FROSTY will try another symmetric vertex for further matching. SubGemini also has a similiar ambiguity guessing solution for symmetric subcircuits, but its transistor-level operation is much slower than the gate-level.

Now let's use one example for illustration: a D-flip-flop subcircuit in Figure 13 to show how to generate the directed graph. Using the partition and gate recognition algorithm in step 1, the circuit can be divided into 10 gates. The gate names and edge names are also labeled in Figure 13. Notice that gate 3 is a pseudo gate made up of two tri-state inverters controlled by the clock signal. Because the two tri-state inverters have the same P-N connection node (the two gates outputs are connected together), they are considered as one CCC. The logic functions of the p-tree and n-tree are $f = \overline{a} \cdot \overline{b} + \overline{c} \cdot \overline{d}$ and $f = a \cdot b + c \cdot d$, respectively. Gate 10 is a T-gate, and other gates are standard gates.

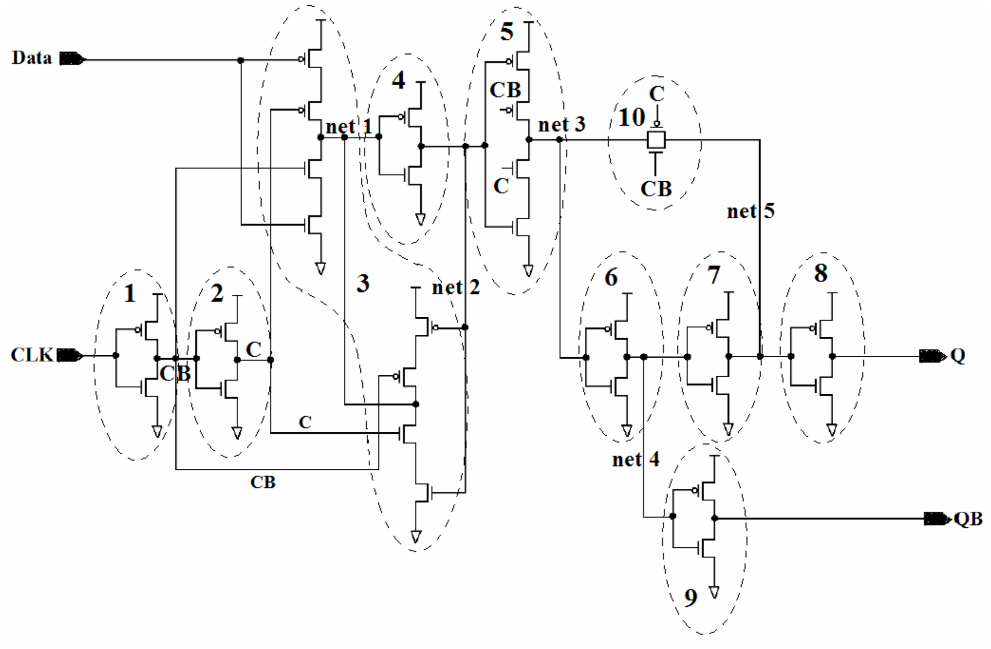


Figure 13. D flip-flop (DFF) circuit after gate recognition.

According to the connection relationships among the gates, a directed graph can be constructed as shown in Figure 14.

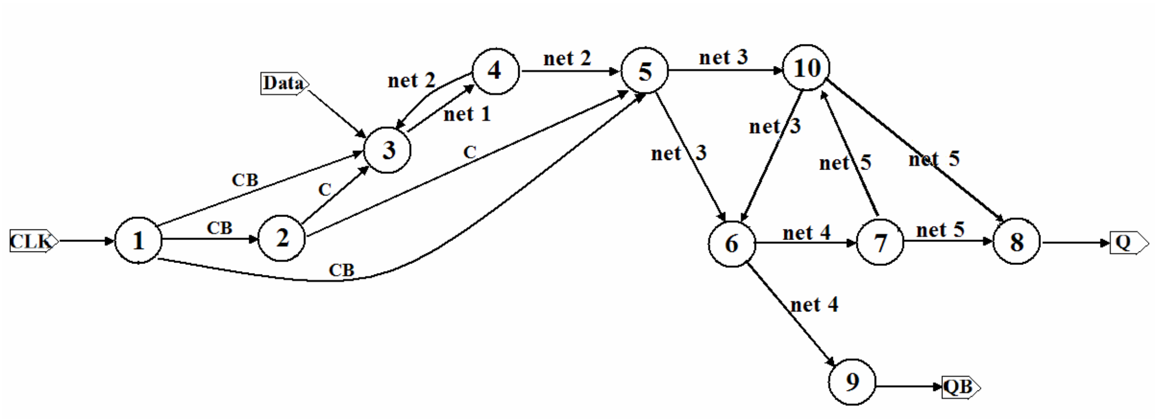


Figure 14. Graph representation of the DFF.

3.3.2 Pattern Matching

After the equivalent gate-level graphs are constructed for the object circuit and the library subcircuits, a pattern matching algorithm is employed to locate all of the defined subcircuits in the object circuit.

The basic pattern matching algorithm is called gradual matching. For each subgraph, an initial vertex K called "source vertex" is chosen to be the key vertex of the subgraph, from the source vertex K all the other vertices can be exhausted through the directed edges in the subgraph. Any object graph vertex C_i that has the same label as vertex K is identified as one possible location of the subcircuit. Then a breadth-first searching process is performed to verify whether there is an subcircuit in this possible location. This is done by initially postulating a match between K and C_i. Starting from this first matching, the algorithm compares K's fanout vertex set F_K in the subgraph and C_i's fanout vertex set F_C in the object graph to find vertex matched pairs. From those matched pairs, program continues to find other matched pairs through comparing their fanout vertices. However, before the comparison in every step, a decision rule is checked to decide that the gradual matching process goes on or stops. The rule can be described that if K's PN vertex is not an output pin of the subcircuit, F_k should equal to F_c ($F_k = F_c$); if K's PN vertex is an output pin, $F_k \subseteq F_c$. If this decision rule is violated, searching process fails and program will try the next candidate C_{i+1}, otherwise the searching process will continue until every vertex in the subgraph corresponds to a vertex in the object graph, at which time searching process ends and one subcircuit is found.

The above pattern matching process can be illustrated with the following example. Consider the DFF in Figure 13 as a subcircuit block defined in the library file. The object circuit, shown in Figure 15, contains the DFF. The corresponding graph of this object circuit is shown in Figure 16. The final pattern matching result is shown in Figure 17.

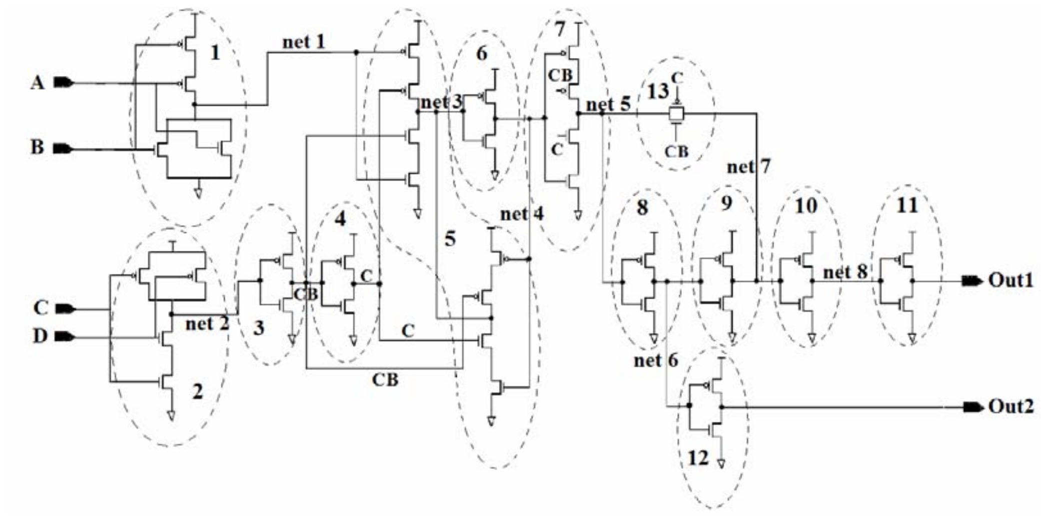


Figure 15. The object circuit after the gate recognition.

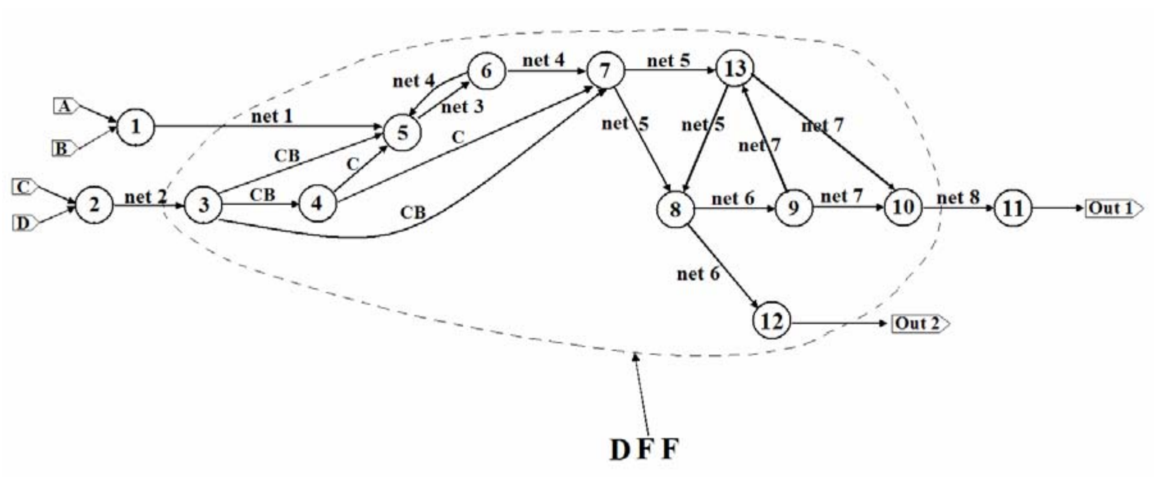


Figure 16. Graph representation of the object circuit.

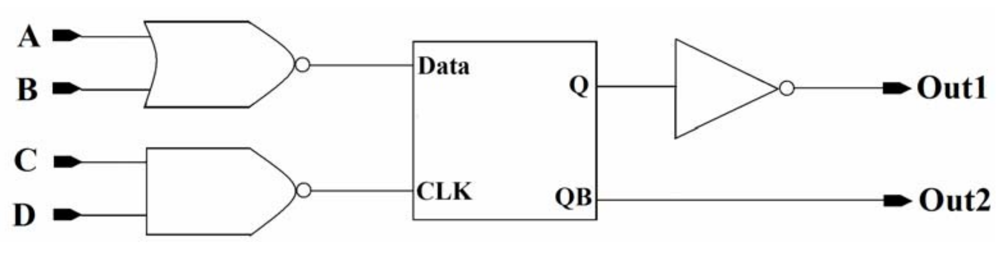


Figure 17. The extracted block-level structure.

In order to find the subcircuit DFF in the object circuit, we apply pattern matching to the subgraph (shown in Figure 14) in the object graph (shown in Figure 16). This means that for every vertex in the subgraph, we attempt to find its corresponding vertex in the object graph, this process is called ***vertex match***. Meanwhile for every input port or output port in the subgraph, we should find its corresponding net in the object graph, which is called ***edge match***. Every time one pair of vertices is matched, the edges connected to them in subcircuit graph and object graph should also be checked for edge matching. In FROSTY, two vertex-stacks and two edge-stacks are set up to hold all the vertex matched pairs and edge matched pairs.

The first step of the gradual matching process is to locate the source vertex *K* in the subgraph. From this source vertex, all other vertices can be reached through directed edges. In order to locate the source vertex, we pick up an output vertex in the subgraph first, and then backtrack to its parent vertices. This backtracking is done recursively until a vertex that has no parent vertices is reached. This vertex is a “source vertex”. In the DFF subgraph in Figure 13, the source vertex is vertex 1. However, sometimes we cannot find the source vertex because the graph may be a ring, as shown in Figure 18. In this case, we can pick any vertex to be the source vertex.

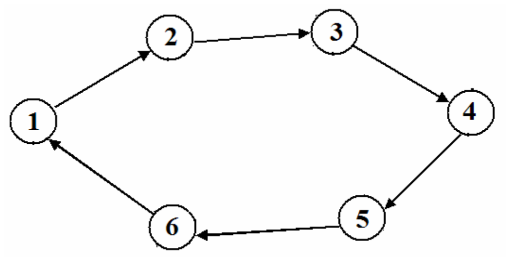


Figure 18. Ring structure of a graph.

Table 3. An example of pattern matching process.

Matching process	**Subgraph Vertex-Stack**	**Object Graph Vertex-Stack**	**Subgraph Edge-Stack**	**Object Graph Edge-Stack**
Loop 1 (source vertex)	1	3	CLK CB	net2 CB
Loop 2	2	4	C	C
	3	5	net 1	net 3
	5	7	net 3 net 2	net 5 net 4
Loop 3	4	6	—	—
	6	8	net 4	net 6
	10	13	net 5	net 7
Loop 4	9	12	QB	Out 2
	7	9	—	—
	8	10	Q	net 8

Loop 5	—	—	Data	net 1

After the source vertex in the subgraph is found, all the vertices in the object graph are checked to locate vertices with the same labels as the source vertex. Any one of these vertices is a possible location of the subgraph. For every such vertex, the source vertex and this possible vertex will be pushed into the subgraph vertex-stack and object graph vertex-stack, respectively, to begin the gradual matching process. For example, vertex 3 in the object graph is found to have the same vertex lable as the source vertex 1 in the subcircuit graph, and they are thus set as the first vertex matched pair to begin the matching process (The whole process is shown in Table 3):

Loop 1:

Vertex 1 and vertex 3 are considered to be the first matched pair, and inserted in the subgraph vertex stack and object graph vertex stack respectively. From this matched pair, the input and output edges should be checked and matched. Because vertex 1 and vertex 3 represent inverters, both of them have only one input and one output, so the input of vertex 1 (CLK) matches input of vertex 3 (edge 2); the output of vertex 1 (CB) matches output of vertex 3 (CB). This edge matching information are written to the edge stack.

Loop 2:

In loop 2, the matching process starts from the pair of matched vertices in the stacks (vertex 1_{subgraph}-vertex $3_{\text{object graph}}$). In the subgraph, vertex 1 connects with vertex 2, 3, and 5, while in object graph vertex 3 connects with vertex 4, 5, and 7. After comparing the labels of the two sets of vertices, we find that the following vertex pairs, vertex 2_{subgraph}-vertex $4_{\text{object graph}}$, vertex 3_{subgraph}-vertex $5_{\text{object graph}}$, vertex 5_{subgraph}-vertex $7_{\text{object graph}}$, have the same labels, respectively. Those pairs are identified to be matched pairs and pushed into the stacks.

After the above vertices match, the edges connected to them are checked and matched. Since vertex 2_{subgraph}-vertex $4_{\text{object graph}}$ are inverters, their inputs CB $_{\text{subgraph}}$-CB $_{\text{object graph}}$, and outputs C $_{\text{subgraph}}$ – C $_{\text{object graph}}$ are matched. For vertex 3_{subgraph}-vertex $5_{\text{object graph}}$, their outputs, edge 1 $_{\text{subgraph}}$ -edge 3 $_{\text{object graph}}$ are matched. However, they both have 4 inputs, only 2 of them CB-CB, C-C have been matched. So it cannot be decided to match the left 2 edges (Data, edge 2) in the subgraph with the left 2 edges (edge 4 and edge 1) in the object graph. For vertex 5_{subgraph}-vertex $7_{\text{object graph}}$, their outputs edge 3_{subgraph} – edge $5_{\text{object graph}}$ are matched. They both have 3 inputs, however, 2 of them CB-CB, C–C have been matched, so the left one edge 2_{subgraph} -edge $4_{\text{object graph}}$ can be matched with each other.

Loop 3:

In loop 3, the matching process starts from all of the newly found matched vertex pairs in loop 2. From matched vertex pair vertex 3_{subgraph}-vertex $5_{\text{object graph}}$, we can match vertex 4_{subgraph}-vertex $6_{\text{object graph}}$; from matched vertex pair vertex 5_{subgraph}-vertex $7_{\text{object graph}}$, we can match vertex 6_{subgraph}-vertex $8_{\text{object graph}}$, vertex 10_{subgraph}-vertex $13_{\text{object graph}}$; from vertex pair vertex 2_{subgraph}-vertex $4_{\text{object graph}}$, we can match vertex 5_{subgraph}-vertex $7_{\text{object graph}}$. All these newly found vertex pairs are also pushed into the stacks.

For edge matching, from vertex 4_{subgraph}-vertex $6_{\text{object graph}}$, edge 1_{subgraph}-edge$3_{\text{object graph}}$, edge 2_{subgraph} -edge $4_{\text{object graph}}$ can be matched. From vertex 6_{subgraph}-vertex $8_{\text{object graph}}$, we can match edge 4_{subgraph}-edge $6_{\text{object graph}}$. And from vertex 10_{subgraph}-vertex $13_{\text{object graph}}$, we can match edge 5_{subgraph}-edge $7_{\text{object graph}}$.

Loop 4:

From the newly matched vertex pair vertex 6_{subgraph}-vertex $8_{\text{object graph}}$ in loop 3, vertex 9_{subgraph}-vertex $12_{\text{object graph}}$ and vertex 7_{subgraph}-vertex $9_{\text{object graph}}$ are matched. From vertex 10_{subgraph}-vertex $13_{\text{object graph}}$, vertex 8_{subgraph}-vertex $10_{\text{object graph}}$ are matched.

For edge matching, starting from vertex 9_{subgraph}-vertex $12_{\text{object graph}}$, edges $\text{QB}_{\text{subgraph}}$ – Out $2_{\text{object graph}}$ are matched. From vertex 8_{subgraph}-vertex $10_{\text{object graph}}$, $\text{Q}_{\text{subgraph}}$ – edge $8_{\text{object graph}}$ are matched.

Loop 5:

In the previous 4 loops, all the vertices in the subgraph have been matched. This means that the DFF block has been found in the object circuit. However, one of the input pins, Data, has not been matched yet. Because Data is the input of vertex 3 in the subgraph, and vertex 3_{subgraph} and vertex $5_{\text{object graph}}$ are matched pairs. Vertex 3_{subgraph} and vertex $5_{\text{object graph}}$ both have 4 input pins, three pairs of them, $\text{CB}_{\text{subgraph}}$-$\text{CB}_{\text{object graph}}$, $\text{C}_{\text{subgraph}}$ – $\text{C}_{\text{object graph}}$ have been matched, so the left pair $\text{Data}_{\text{subgraph}}$ and edge $1_{\text{object graph}}$ must be matched with each other.

The above loops show the gradual matching process for vertex match and edge match of the DFF block, as shown in Table 3. In every step when we search from the newly matched pair K_{subgraph} – $C_{\text{object graph}}$, their fanout set F_K and F_C are prechecked according to the decision rule stated before. If the rule is violated, this breadth-first searching stops, and the vertex-stacks are emptied for next matching process.

3.4 Overall Algorithm and Complexity

The entire algorithm of FROSTY is shown in Table 4. In FROSTY, hash tables are used wherever possible to make its search time linear. The complexity of the program is O(k*n + g1*g2), where k*n represents the complexity of step 1, n is the number of transistors in the circuit, k is an integer number, so that the run time for step 1 is linear in n; g1*g2 represents the complexity of step 2, where g1 and g2 are respectively the number of gates in all library subcircuits and the number of gates in the object circuit after step 1.

Table 4. Algorithm FROSTY.

```
PREPROCESS FOR LIBRARY:
LOOP: for i = Subcircuit $_1$ : Subcircuit $_n$ (in library file) {
        Partition the subcircuit into p-trees and n-trees
        Combine p-trees and n-trees to channel-connected-components(CCCs)
        Recognize transmission gates in the left transistors
        Recognize the logic function of every CCC in Subcircuit$_i$
        }
INITIALIZATION:
Construct hash table to store transistors and vertices of the object circuit
Partition the object circuit into p-trees and n-trees
Combine p-trees and n-trees to channel-connected-components(CCCs)
Recognize transmission gates in the left transistors
Recognize the logic function of every CCC
LOOP: for i = Subgraph $_1$ : Subgraph $_k$ (in library file){
OuterLoop:
    Find "source vertex" in the Subgraph$_i$ and push it into vertex-stack
    InnerLoop:
    for j = Vertex$_1$ : Vertex $_n$ (in the object graph) {
        if (Label (Vertex $_i$ ) = Label (Source Vertex)) {
            Push source vertex and Vertex $_i$ into the vertex-stack.
            Do {
                Searching from matched vertices in vertex-stack to find new matched vertices, and push them into vertex-stack.
                Match edges connected with the vertices and push them into edge-stack.
            }
            while {checking rule violates or all vertices and I/O pins in Subgraph$_i$ has been matched}
        }
        }
    }
    Output the blocks and gates to a Verilog or VHDL block-level netlist
```

4. Discussion of the Algorithm

4.1 Backtracking in pattern matching

FROSTY exploits backtracking mechanism to recognize the subcircuits containing several source vertices. In the process of recognizing these subcircuits, vertices that are not on the searching path will be not exhausted, because the gradual matching process always searches through the directed edges, so backtracking steps are needed to recognize them. One backtracking example is shown in Figure 19. In this example, both vertex 0 and vertex 2 can both be source vertices. If vertex 0 is picked to be the source vertex, vertex 2, 4 are left unexhausted; If vertex 2 is the source vertex, vertex 0, 1, 3, 5 are left unexhausted.

The algorithm of finding a subcircuit containing backtracking process can be described as follows:

1) Decide which vertex is the source vertex.

From an output vertex, program backtracks and find its parent vertices. This backtracking is done recursively until a vertex that has no parent vertices is reached. This vertex is a source vertex. However, if several source vertices are found, the vertex which has the longest backtracking steps is picked to be the source vertex. In the example shown in Figure 19, suppose backtracking starts from vertex 8. Then vertex 0 is found to be source vertex after 5 backtracking steps, while vertex 2 is found to be source vertex after 4 steps. So, vertex 0 is picked to be the source vertex.

2) Gradual matching process.

From vertex 0, the gradual matching process will match vertex 0—1—3—5—6—7—8 step by step.

3) Backtracking process.

If the gradual matching process stops and there are still several unmatched vertices left, the program will check for each unmatched vertex to find if it has children vertices that are already matched. If such matched children vertices exist, FROSTY backtracks from them and matches this unmatched vertex. In the example, after gradual matching process, vertex 2 and 4 are left unmatched. After checking the vertex stack, program found matched vertex 6 is the children vertex of vertex 4, so after this backtracking step, vertex 4 can be matched. Performing the same algorithm, vertex 2 is backtracked and matched.

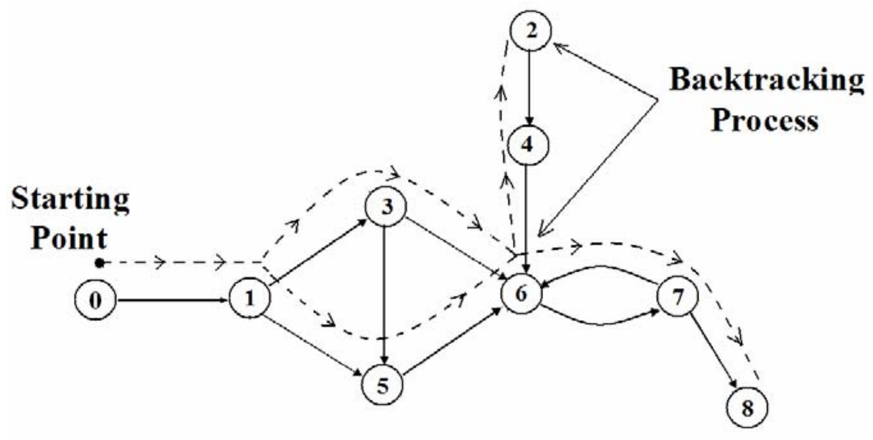


Figure 19. An illustration of the backtracking process.

Generally backtracking is a much slower process in the subgraph isomorphism and degrades the program efficiency. However, in FROSTY, backtracking is the reverse process of gradual matching, and it is as fast as the gradual matching. Gradual matching starts from an already matched vertex, gets a group of its fanout vertices for next step matching. While backtracking starts from an already matched vertex, gets a groups of its fanin vertices for next step matching. In the example shown in Figure 19, backtracking from vertex 6, it has three fanin vertices, vertex 3, 4, 5. Because vertex 3 and 5 have already been matched, vertex 4 is discriminative and can be matched easily.

4.2 Subcircuit input/output ports channel-connected to GND/VDD

The FROSTY step 1 CCC partition algorithm relies on the assumption that in the object circuit, the image of subcircuit input/output ports are not channel-connected to GND or VDD vertex (if gate-connected with GND/VDD, it doesn’t matter). An example of this assumption is shown in Figure 20, the subcircuit is a SRAM cell. In the object circuit, one input port of the subcircuit image channel-connects to GND. After FROSTY step 1, the SRAM subcircuit is partitioned into gate g1, g2, g3 and g4. While in the object circuit, because the n-tree-finding algorithm will search from every GND-channel-connected transistor, thus the subcircuit image in the object circuit is partitioned into 3 gates, G1, G2 and G3, in which G2 is a pseudo gate, which causes mismatch.

If after extracting all the library defined subcircuits, there are still pseudo gates left in the object circuit, that may due to this kind of GND/VDD connection failure. To solve this problem, FROSTY analyzes the structure of every pseudo gate left in the object circuit. For any pseudo gate in which the number of transistors in p-tree and n-tree are not equal, program divides this pseudo gate into one standard CMOS gate plus a pass transistors and re-does the pattern matching. This re-matching process can be very fast because the object gate-level graph is only made of several unrecognized subcircuit images, but it definitely increases the CPU time.

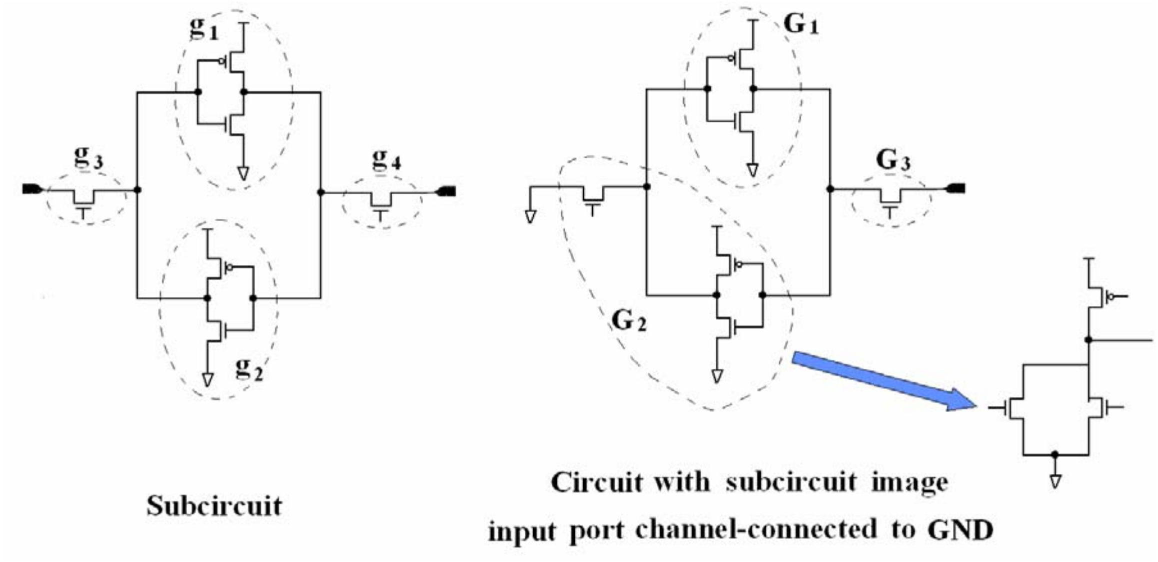

g1
g2
g3
g4
Subcircuit
G1
G2
G3
Circuit with subcircuit image
input port channel-connected to GND

Figure 20. An example subcircuit input channel-connected to GND.

5. Experimental Results

FROSTY was written in C++ and executed on SUN Fire V480 server with 900MHz UltraSparc-III processors and 16GB RAM. Results from applying FROSTY to several industrial circuits from Boeing are presented in this section.

Table 6 shows the statistics of a set of test circuits and the CPU times used by FROSTY for recognizing all the gates and blocks. Test circuits PSM, PSM-7, PSM-17, PSM-43 are a set of digital CMOS designs in Boeing's "Power Supply Monitor ASIC" on F22 airplane. Test circuits CEGRP, CEGRP-3, CEGRP-5, CEGRP-7, DFGRP, DFGRP-2, DFGRP-4, DFGRP-6 are a set of digital designs in Being's "Pressure Belt Chip". This chip is used in Boeing's airplane to determine the structural load on aircraft wings by measuring the pressure distributed on the top and bottom of the wing. These test circuits contain a lot of Boeing defined behavior blocks, such as DFFs, latches, MUXs, adders, etc. None of the Boeing test circuits contain blocks that couldn't be partitioned into CCCs. Table 5 shows in detail how many blocks are contained in the circuits, for example, in PSM, there are 3 different structures of DFFs and the total number of DFFs is 122. Using the library file provided by Boeing, FROSTY extracts all of the blocks in the library file and outputs a behavior block level Verilog/VHDL netlist containing the recognized blocks as well as standard CMOS gates that do not belong to any block.

Table 5. Subcircuits types and numbers in test circuits.

		PSM	**CEGRP**	**DFGRP**
DFF	Types	3	10	5
	Number	122	1304	1436
Latch	Types	1	—	—
	Number	15	—	—
Adder	Types	—	3	5
	Number	—	118	646
MUX	Types	1	3	4
	Number	27	1049	508
XNOR	Types	—	2	2
	Number	—	290	74

Table 6. Results of FROSTY and SubGemini.

Circuits	#Transistors	# CMOS gates	# Behavior blocks	FROSTY CPU Time (s)			SubGemini Results	
				Setup and Step-1	Step 2	Total	# Extracted Behavior blocks	CPU Time (s)
PSM	4520	1516	164	1.17	0.58	1.75	156	1.4
PSM - 7	31640	10612	1148	7.83	3.97	11.8	1091	13.6
PSM - 17	76840	25772	2788	18.98	9.88	28.86	2651	48.4
PSM - 43	194360	65188	7052	49.9	25.2	75.1	6714	277.1
CEGRP	104236	10714	2761	26.45	15.03	41.48	1419	94.1
CEGRP - 3	312708	32142	8283	81.57	46.83	128.4	4568	1113.5
CEGRP - 5	521180	53570	13805	135.0	78.31	213.35	7093	3178.1
CEGRP - 7	729652	74998	19327	191.63	113.67	305.32	9926	5856.2
DFGRP	119257	10048	2664	30.1	45.8	75.9	2001	108.0
DFGRP - 2	238514	20096	5328	60.1	91.5	151.6	3074	1113.9
DFGRP - 4	477028	40192	10656	121.2	169.96	219.2	5652	2456.9
DFGRP - 6	715542	60288	15984	182.45	233.7	416.2	8802	4317.2

In Table 6, we compare our results with SubGemini [12]. For each test circuit in Table 5, we try to use SubGemini to extract all of the behavior blocks in Boeing's library. However, SubGemini fails to recognize some blocks. Part of the reason is that SubGemini can not recognize the subcircuits with shoring inputs. The recognized block numbers and CPU time of SubGemini are listed in columns 8 and 9 in Table 5. The CPU time comparison between FROSTY and SubGemini is shown in Figure 21.

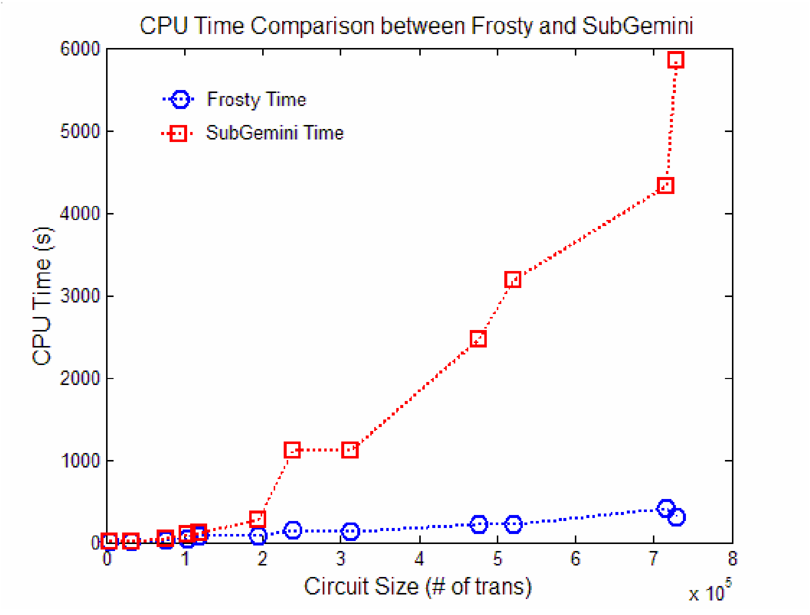


Figure 21. FROSTY and SubGemini CPU time comparison

To analyze why FROSTY is faster than SubGemini, we first recognize that they are both two-phase algorithms. In phase I, SubGemini exploits relabeling algorithm to decide a key vertex *K* in subgraph and a set of candidate vector *CV* in the object graph, thus reducing the searching space. While FROSTY performs backtracking algorithm to determine a key vertex *K* in the subgraph, and considers every vertex C_i with the same label as *K* in the object graph to be a candidate vertex, so its searching space is not so reduced. However, FROSTY's fast graduate matching algorithm in phase II can compensate it because: 1) It is breadth-first search, so a candidate vertex C_i can be denied quickly after the decision rule check; 2) Starting from the source vertex, FROSTY can exhaust the subgraph through the directed edge much fast than SubGemini; 3) FROSTY gate-level graph is much smaller than SubGemini; 4) SubGemini's vertex match time is reasonable, but its edge match process is pretty slow, sometime even resulting in recognition failure.

From Table 6 we can see, for test circuit CEGRP-7 (729652 transistors), recognizing 74998 gates and 19327 behavior blocks only takes 305.32 seconds. Normally, FROSTY is faster than SubGemini for larger circuits and libraries. For example, it is 20 times faster than SubGemini for CEGRP-7 circuit.

The run time of FROSTY depends on two factors: 1) the size of circuits and 2) the number of behavioral blocks in the library file. To observe the relationship between circuit sizes and CPU times, we use a set of PSM circuits and perform the extraction with the same library file.

In Figure 22 the relationship between the CPU time and the circuit size is displayed. We can see that FROSTY running time is linear in the size of a circuit.

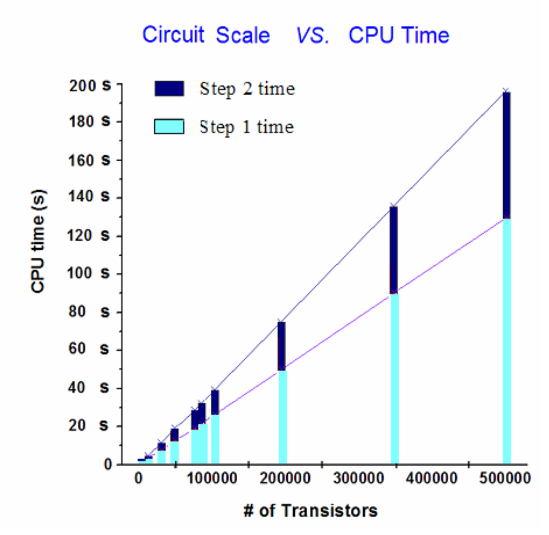


Figure 22. CPU time vs. circuit size.

6 Conclusions and Future Work

This paper presented FROSTY, a computer program for the automatic extraction of circuit hierarchy targeted for the post-layout simulation and verification of library-based large-scale CMOS circuit design. By condensing both the object circuit and library circuits into graphs of blocks and then applying the pattern matching algorithm at the gate level, FROSTY has demonstrated that it can extract an industrial design with seven hundred thousand transistors in less than five minutes on a modern Sun workstation. By representing the extracted hierarchy using high-level descriptions such as VHDL and Verilog, the output netlist can be simulated efficiently by any digital simulator.

Our future efforts will focus on extending the FROSTY algorithm to make it work on dynamic CMOS circuits. Also we will try to exploit some optimization strategies, such as PMAA algorithm in [13] in the pattern matching process.

Acknowledgements

The authors thank Prof. Carl Ebeling of Computer Science at the University of Washington for providing SubGemini and several help discussions, Scott Davis and Daniel

Chaney of Boeing Solid State Electronics for providing us industry test circuits, and Dr. Pavel Nikitin and Dr. Alicia Manthe for careful proof reading of this paper.

Reference

[1] M. S. Abadir and J. Ferguson, "An improved layout verification algorithm (LAVA)", *Proc. European Design Automation Conference*, pp. 391-395, 1990.

[2] T. J. Thatcher and R. A. Saleh, "Automatic partitioning and dynamic mixed-mode simulation", *Proc. IEEE Custom Integrated Circuits Conference*, pp. 12.7.1-12.7.4, 1992.

[3] K.-T. Huang and D. Overhauser, "A novel graph algorithm for circuit recognition", *Proc. IEEE International Symposium on Circuits and Systems*, pp. 1695-1698, 1995.

[4] I. Parulkar, M. A. Breuer, and C. A. Njinda, "Extraction of a high-level structural representation from circuit description with applications to DFT/BIST", *Proc. IEEE/ACM Design Automation Conference*, pp. 345-350, 1994.

[5] S. Jolly, A. Parashkevov, and T. McDougall, "Automated equivalence checking of switch level circuits", *Proc. IEEE/ACM Design Automation Conference*, pp. 299-304, 2002.

[6] S. Kundu, "GateMaker: A transistor to gate level model extractor for simulation, automatic test pattern generation and verification", *Proc. International Test Conference*, pp. 372-381, 1998.

[7] M. Boehner, "LOGEX – an automatic logic extractor from transistor to gate level for CMOS technology*", Proc. IEEE/ACM Design Automation Conference*, pp. 517-522, 1988.

[8] A. Lester, P. Bazargan-Sabet and A. Greiner, "YAGLE, a second generation functional abstractor for CMOS VLSI circuits", *Proceedings of the Tenth International Conference on Microelectronics*, pp. 265-268, 1998.

[9] F. Luellau, T. Hoepken and E. Barke, "A technology independent block extraction algorithm", *Proc. IEEE/ACM Design Automation Conference*, pp. 610-615, 1984.

[10] G. Pelz and U. Roettcher, "Pattern matching and refinement hybrid approach to circuit comparison", *IEEE Transactions on Computer-Aided Design*, vol. 13, no. 2, pp. 264-275, Feb. 1994.

[11] H. Graeb, S. Zizala, J. Eckmueller and K. Antreich, "The sizing rules method for analog integrated circuit design", *Proc. IEEE/ACM International Conference on Computer-Aided Design*, pp. 343-349, 2001

[12] M. Ohlrich, C. Ebeling and E. Ginting, "SubGemini: Identifying subcircuits using a fast subgraph isomorphism algorithm", *Proc. IEEE/ACM Design Automation Conference*, pp. 31-37, 1993.

[13] N. Rubanov, “SubIslands: The probabilistic match assignment algorithm for subcircuit recognition”, *IEEE Transactions on CAD of Integrated Circuits and Systems*, vol. 22, pp. 26-38, Jan. 2003.

[14] G. Pelz and U.Roettcher, “Circuit comparison by hierarchical pattern matching”, *Proc. IEEE/ACM International Conference on Computer-Aided Design*, pp.290-293, 1991.

[15] S. Gold and A. Rangarajan, “Graduated assignment algorithm for graph matching”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 4, pp.377-388, 1996.

[16] S. W. Rosenfeld, A. Hummel and R. A. Zucker, “Scene labeling by relaxation operations”, *IEEE Transactions on Man and Cybernetics*, vol. SMC-6, no. 6, 1976.

Lei Yang received the B.S. degree in HuaZhong University of Science and Technology, Wuhan, China, in 1998, and the MS. degree in TsingHua University, Beijing, China, in 2001. During the MS. Degree period, he worked on RF SAW filter design, FPGA design and digital ASIC design. He is currently working towards his Ph.D degree in the Mixed-Signal CAD lab, Electrical Engineering department at the University of Washington. His research interests are in the field of mixed-signal VLSI and analog IC design automation.

C.-J. Richard Shi (M’91-SM’99) is currently an Associate Professor in Electrical Engineering at the University of Washington. His research interests include several aspects of the computer-aided design and test of integrated circuits and systems, with particular emphasis on analog/mixed-signal and deep-submicron circuit modeling, simulation and design automation.

Dr. Shi is a key contributor to IEEE std 1076.1-1999 (VHDL-AMS) language standard for the description and simulation of mixed-signal circuits and systems. He founded IEEE International Workshop on Behavioral Modeling and Simulation (BMAS) in 1997, and has served on the technical program committees of several international conferences. Dr. Shi has authored or co-authored over 100 papers published in international journals and conferences, and has served as the principal investigator of several research projects supported by DARPA, SRC and NSF with over $8M funding.

Dr. Shi received a Best Paper Award from the IEEE/ACM Design Automation Conference, a Best Paper Award from the IEEE VLSI Test Symposium, a National Science Foundation CAREER Award, and a Doctoral Prize from the Natural Science and Engineering Research Council of Canada. He has been an Associate Editor, as well as a Guest Editor, of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II, ANALOG AND DIGITAL SIGNAL PROCESSING. He is currently an Associate Editor of IEEE Transactions on Computer-Aided Design of Integrate Circuits and Systems.

An FPGA Implementation of Low-Density Parity-Check Code Decoder with Multi-Rate Capability

Lei Yang, Manyuan Shen, Hui Liu and C.-J. Richard Shi
Department of Electrical Engineering,
University of Washington, Seattle, WA 98195
{yanglei,mshen, hliu, cjshi}@ee.washington.edu

Abstract: With superior error correction capability, low-density parity-check (LDPC) has initiated wide scale interests in wireless telecommunication fields. In the past, various structures of single code rate LDPC decoders have been implemented for different applications. However, in order to cover a wide range of service requirements and diverse interference conditions in wireless applications, LDPC decoders that can operate in both high and low code rates are desired. In this paper, a new multi-rate LDPC decoder architecture is presented and implemented in a Xilinx FPGA device. Through selection pins, three operating modes with the irregular 1/2 rate, regular 5/8 rate and regular 7/8 rate are supported. The measurement results show LDPC decoder can achieve BER below 10^{-5} at SNR of 1.4dB in the most critical case with the irregular 1/2 mode.

1. INTRODUCTION

Recently, low-density parity-check (LDPC) codes have attracted an increasing amount of attention due to their amazing error correction capacity. It has been shown that, in the infinite LDPC block length, 0.0045 dB from the Shannon limit is possible, and with block length 10^7, the code can achieve 0.04 dB from the Shannon limit at a bit error rate of 10^{-6} [1]. Furthermore, the LDPC decoding algorithm is inherently parallel and is much easier to be implemented than its comparator turbo coding, thus makes it more attractive.

LDPC code is a linear block code described with a binary sparse M $\times$ N parity-check matrix H. Each row of the matrix H corresponds to a parity check and each column represents a demodulated symbol [2]. The number of non-zero elements in each row or column is defined as a row weight or column weight. The LDPC code with uniform row weight and column weight is called a regular code. Otherwise it is an irregular code. Normally, Tanner graph is widely used to represent LDPC codes. Tanner graph is a bipartite graph with variable nodes on one side and constraint nodes on the other side. Each variable node in the graph corresponds to a received symbol, each constraint bit corresponds to a set of parity check constraints, and each edge corresponds to a non-zero entry in the parity check matrix.

With consistency in the above structure, iterative logarithmic belief propagation (Log-BP) algorithm [3] is considered to be the best algorithm suitable for hardware implementation so far. During the decoding process, logarithmic messages are exchanged along the graph edges, and computed at the variable/check nodes. Unfortunately, efficiently mapping the algorithm to a VLSI implementation is challenging [2]. Several approaches instantiate the BP algorithm to hardware using the most natural way, where each variable node is mapped to a variable processor, each check node is mapped to a check node processor, and all the processors are connected through Tanner graph interconnection network. This architecture can achieve amazing parallelism, but it is not scalable to LDPC codes with large block lengths due to the heavy burden of hardware resource usage. To elude this problem, partial parallel architectures compromising between operating speed and hardware load are proposed, in which a certain number of variable nodes and check nodes are mapped to one hardware unit in time-division multiplexing mode [3]. However, all the implementations up to now have been limited to single rate LDPC code designs. In reality, especially in wireless applications, it is more desirable that a design scheme could adopt different coding rates in order to meet various service requirements and interference conditions.

In this paper, a multi-rate LDPC design architecture is presented. This architecture is not only suitable for different regular LDPC coding rates, but also can be employed to irregular LDPC codes. The proposed architecture is demonstrated through the design and implementation of a 10k bit, multi-rate LDPC decoder in Xilinx FPGA, which is the first published implementation of a multi-rate LDPC code decoder. Through configuring two pins of the FPGA device at "00", "01" or "10", the decoder works on three coding rate modes: 1/2 as irregular code, 5/8 and 7/8 as regular codes, showing the great flexibility of our scheme.

The remainder of this paper is organized as follows. An overview of the LDPC Log-BP algorithm is given in Section 2. The detail of designing the three rate codes is described in Section 3. The architecture of the multi-rate decoder is presented in section 4. Measurement results of the implemented FPGA device are shown in section 5. Concluding remarks are given in Section 6.

2. LDPC DECODING ALGORITHM

If we use μ to represent the log-likelihood ratio (LLR) messages exchanged between variable nodes and check nodes, and γ_i to stand for intrinsic probability information for every bit from a demodulator. The iterative Log-BP algorithm is summarized in the following steps.

1) Initialization

All variables nodes and their outgoing variable messages are initialized to the value of the intrinsic messages. The intrinsic message is defined as:

$$\gamma_i = \log\left[\frac{P(x_i = 0 \mid y_i)}{P(x_i = 1 \mid y_i)}\right] \qquad (1)$$

Here, y_i is the received symbol and x_i is the transmitted symbol.

2) Check Node Computation

After the incoming messages are gathered in each check node from its connected variable nodes in the Tanner graph, the following check node computation is performed:

$$\mu_{CV} = \mathrm{sgn}\left(\prod_{j=1}^{d_C-1} \mu_{V_jC}\right)\Psi^{-1}\left(\sum_{j=1}^{d_C-1} \Psi(\mu_{V_jC})\right) \qquad (2)$$

Here d_C is the degree of the check node *C*, $\mu_{V_j C}$ represents the incoming message from neighbor variable node $V_j \neq V$ to check node *C*, and μ_{CV} is the outgoing message from check node *C*. Function Ψ is equal to Ψ^{-1}, and is expressed in the following:

$$\Psi(x) = \Psi^{-1}(x) = \log(\frac{1+\exp(-|x|)}{1-\exp(-|x|)}) \qquad (3)$$

After the check node computation, the outgoing messages are passed to variable nodes along the edges.

3) Variable Node Computation

The variable node computation is expressed in the comparatively simple equation as follow:

$$\mu_{VC} = \sum_{i=0}^{d_V-1} \mu_{C_i V} \qquad (4)$$

Where $\mu_{C_i V}$ is the incoming message from the neighbor check node $C_i \neq C$ to *V*, d_V is the number of check nodes connected to node *V*, and μ_{VC} is the outgoing message from variable node *V*.

4) Check Stop Criterion

When the variable node computation is finished, the LLR of every symbol *i* is updated as:

$$\lambda_i = \gamma_i + \sum_{i=0}^{d_V-1} \mu_{C_i V} \qquad (5)$$

From the updated LLR vector $\lambda = \{\lambda_1, \lambda_2, \ldots \lambda_i, \ldots \lambda_N\}$, a hard decision result $X = \{x_1, x_2, \ldots x_i, \ldots x_N\}$ is calculated as:

$$x_i = \begin{cases} 1, & if\ \lambda_i \leq 0 \\ 0, & if\ \lambda_i > 0 \end{cases} \qquad (6)$$

The calculated hard decision vector *X* is then checked against the parity check matrix *H*. A case of $H \cdot X = 0$ means the iterative process has converged to the correct codeword and decoding stops. Otherwise, step 2) and 3) have to be repeated until $H \cdot X = 0$ or a fixed number of iterations is reached.

3. LDPC CODES DESIGN

As stated previously, three rate codes are implemented: irregular rate 1/2, regular rate 5/8 and 7/8. Based on many publications such as [8], irregular codes could outperform regular codes in term of coding gain, thus rate 1/2 is implemented as irregular code dealing with the worst transmission situation. While rate 5/8 are 7/8 are designed as regular structures to simplify hardware complexity.

In our design, similar code construction algorithms as [3][4] are adopted to attain the optimal balance between speed and hardware complexity. The main point of these methods is to build a parity check matrix H made of a set of square permutation L×L matrices. Each permutation matrix is an identity matrix whose rows have been circularly shifted by a set of amount. This kind of code structure can be smoothly mapped to a partial decoder structure implementation, in which there are N_P variable node computation units (VNU) and M_P check node computation units (CNU), and every VNU and CNU contain time-multiplexed N/N_P variable nodes and M/M_P check nodes respectively.

1) Regular rate 5/8 code

A (3, 8) regular LDPC structure is adopted to construct the rate 5/8 code (every column has 3 non-zero elements, and every row has 8 non-zero elements), because this structure can provide good BER performance for moderate lengths. Similar as [3], H is made of three submatrixes $[H_1^T, H_2^T, H_3^T]^T$. First a $H' = [H_1^T, H_2^T]$ is constructed as a high girth (2, 8) regular structure. Submatrix H_1 and H_2 contain $8^2=64$ permutation 149×149 matrixes. The codelength $N=149\times8^2$ is equal to 9536, and the check node number $M=3\times8\times149$ is 3576. However, unlike [3], in which H_3 is specified by certain configuration parameters to realize the rather “random” function, we design H_3 as the structure with permutation matrixes randomly located. The simulation results show that our method has almost no gain loss compared to [3], but uses much less hardware resources.

Because there are total $8^2=64$ small matrixes in each row. One natural way to design the hardware architecture is to generate $N_P=64$ VNUs. However, the parallelization factor $N_P=64$ is too high to fulfill the latency requirement, so every two 149×149 matrixes are folded into one VNU to form a decoder structure with $N_P=32$. Each of the VNU has 3 RAMs with depth 2×149=298. The check node parallelization factor M_P is designed to be 12. Overall, 298 clock cycles are needed to finish one VNU computation process and one CNU computation process.

2) Regular rate 7/8 code

(3, 24) regular code structure is used to design the regular rate 7/8 code. The rate 7/8 matrix construction method is similar to the rate 5/8. First, a high girth (2, 24) structure $H' = [H_1^T, H_2^T]$. Both H_1 and H_2 contain $24^2=576$ permutation 17×17 matrix. So the code length N= 17×24^2 is 9792, and check node number M=3×17×24 is 1224. Then, a substructure H_3 is constructed with permutation matrixes randomly located.

The parallelization factors N_P and M_P are 24 and 3 respectively. Each VNU contains 3 RAMs, every RAM fold 24 17×17 small permutation matrixes, hereby the RAM depth is 24×17=408. In one clock, every VNU performs 3 variable nodes computation, and every CNU performs 24 check node computation. Totally, 408 clock cycles are needed for one iteration of VNU computation process and CNU computation process.

3) Irregular rate 1/2 code

Irregular LDPC codes do not outperform regular ones unless their degree distribution and girth are carefully designed. In this paper, the method to design irregular code is the modified bit-filling algorithm [5]. In order to be compatible with the rate 5/8 and the rate 7/8, $N_P=36$, $M_P=18$ are chosen, and every small permutation matrix is 251×251, so the code length is 36×215=9036, and check node number M= 18×251=4518.

As to the degree distribution design, there are some related works shown in [6]. In our implementation, a {2, 3, 7} degree distribution is chosen with average degree of 3.25. This distribution achieves comparably good results through fix point simulations. Once the codelength and degree distribution is determined, a modified bit-filling algorithm similar as [5] is performed to construct a $N_P\times M_P$ matrix, which is then expanded to a $(251\times N_P)\times(251\times M_P)$ matrix.

4. DECODER ARCHITECTURE

In this part, a configurable partial parallel decoder architecture is described. This architecture is not only suitable for different rates, but also can be used for both regular and irregular LDPC codes. Compared with the structure in [3][4], the multi-rate architecture has the following distinct characteristics:

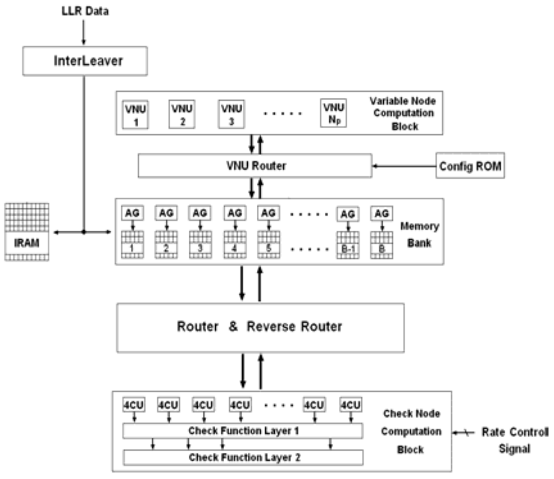


Figure 1. The multi-rate decoding architecture.

- It exploits configurable structures for the VNU block, the CNU block and the memory banks so different rate LDPC regular/irregular codes can be fit.
- It contains a built-in interleaver, which eliminates extra interleaver/deinterleaver need in a communication system.
- Min-sum with correction (MSC) algorithm [7] is adopted to replace normally applied table-lookup quantization method in order to diminish large performance loss.

4.1 FINITE PRECISION IMPLEMENTATION

Up to now all publications on LDPC VLSI implementation employ look-up tables (LUT) to quantize function Ψ. However, the LUT quantization method suffers from the tradeoff between dynamic range and precision. To eliminate this tradeoff, [7] present the MSC algorithm to emulate $\Psi(x)$.

$$L(x \oplus y) = \mathrm{sgn}(xy)\min\{|x|,|y|\} + \ln\left(\frac{1+e^{-|x+y|}}{1+e^{-|x-y|}}\right) \qquad (7)$$

The correction term in equation (7) can be further simplified as:

$$\ln\left(\frac{1+e^{-|x+y|}}{1+e^{-|x-y|}}\right) = \begin{cases} 0.5, & |x+y| \le 1, |x-y| > 1 \\ -0.5, & |x+y| > 1, |x-y| \le 1 \\ 0, & else \end{cases} \qquad (8)$$

The above two equations describe the MSC algorithm and elude the problem of computing non-linear function $\Psi(x)$, thus gives no requirement on the precision and enables freedom to increase the dynamic range. To show the performance of MSC algorithm, both LUT and MSC methods are used to simulate the 3 codes in our design. (6:3) and (6:1) quantization schemes are used for LUT and MSC respectively, here 6 means totally 6 bits are utilized, in which 3 or 1 bits are used for the fractional part of the value. The simulation results are shown in Fig. 2. From the curves we can see that the (6:3) LUT method performs badly at high rate regular 7/8 code and irregular 1/2 code, and other quantization schemes than (6:3) show even worse results. Fortunately, the MSC algorithm achieves good results for all the three cases.

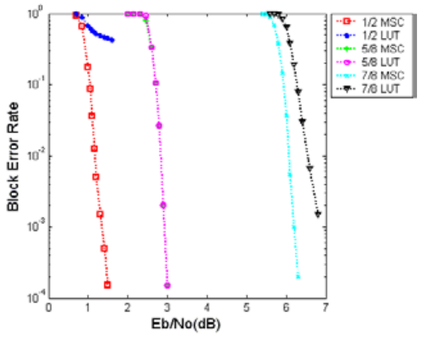


Figure 2. Comparison of LUT and MSC simulation result

4.2 CHECK NODE COMPUTATION BLOCK

Because MSC algorithm is adopted in our design, every pair of messages needs to be performed by the check functions (7) and (8). For example, one check node with degree k follows the equation below to obtain the outputs:

$$OUT_n = \sum_{i=1->k}^{i \neq n} \oplus IN_i \qquad (9)$$

Every output OUT_n is equal to the checking result of all the other k-1 input messages in the check node. In our architecture, a multi-layer tree structure is proposed for fast parallel check node computation instead of normally proposed serial way, this structure is configured to adapt to different rates.

The first layer of the network contains a set of 4-check-node-computation-units (4CUs). Each 4CU computer checking results of the 4 input messages. In the second layer, 4CU sets are connected to a set of 8CUs or 12CUs. Fig. 3 shows an example of an 8CU made of two 4CUs. The 8CUs and 12CUs can be further connected to 24CUs or 36CUs in the third layer.

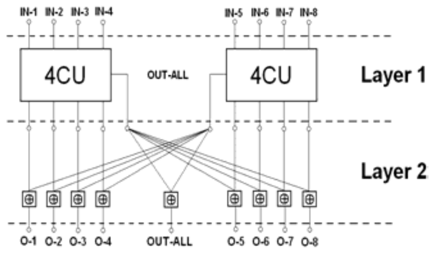


Figure 3. Architecture of the 8CU made of two 4CUs.

4.3 VARIABLE NODE COMPUTATION BLOCK

Variable node computation in our configuration becomes simpler than in [3][4] because of employing MSC check function operators in CNUs. This avoids the use of lookup-tables and fix-point format conversion (between sign-magnitude and two's complement). Fig. 4 shows the VNU architecture with node degree 3. The input of this VNU is one intrinsic message and three check-to-variable messages. The output is three computed variable-to-check messages and 1-bit hard decision result.

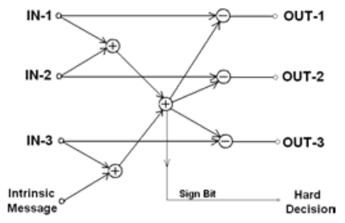


Figure 4. Architecture of j=3 VNU without look-up table

In rate 5/8 and 7/8 regular codes, variable node degree is uniformly distributed and each VNU is connected to 3 RAMs. However, in irregular code, every VNU is associated with different number of RAMs due to the non-uniform distribution of variable node degree. In order to make different rate code designs co-exist in one implementation, a VNU router is inserted between the memory bank and the variable node computation block to adapt to different code rates.

5. FPGA IMPLEMENTATION AND RESULTS

Employing (6 : 1) quantization scheme and the architecture described in section 4, a multi-rate LDPC decoder is implemented on Xilinx Virtex-II XC2V8000 FPGA device. The design is described in VHDL, synthesized by Synplicity, placed and routed using Xilinx development tool ISE6.0. It works at 100MHz clock frequency and has codeword length about 10k bits.

The employed XC2V8000 FPGA belongs to Xilinx Virtex-II family, containing 168 18k-bit dual-port SelectRAM blocks and 46,592 slices, possessing the capacity to handle 8-million-gate design. One of the most challenging problems in the LDPC decoder design is memory usage because LDPC code structure requests large number of parallel working memories to store exchanged messages. Therefore, memory blocks need to be properly partitioned to fully utilize the FPGA SelectRAM resources. In our architecture, memory bank and IRAM are the main sources of memory usage. The memory bank contains 117 512*7 independent RAM blocks, and IRAM is a large 4.5K*32 memory. Those RAMs are carefully combined to utilize the dual-port FPGA RAM resources. The resource utilization statistics is shown in Table 1.

Table 1. FPGA Resource Usage Statistics.

Resource	Number	Usage Rate
Slices	34,127	73%
Slice Flip Flops	24,570	26%
4 Input LUTs	53,327	57%
Block RAMs	102	60%
Bonded IOBs	75	9%
DCMs	1	8%

Another challenging problem of designing LDPC decoder is the routing congestion caused by the complex top-level connections. In our implementation, this problem is alleviated by carefully pipelining the data paths between VNU block, router and CNU block. In addition, the critical delay path is optimized and the decoder is able to operate at 100MHz clock frequency. At this frequency, the decoder achieves a maximum throughput of 66Mbps for regular 5/8 and 7/8 mode when performing maximum 24 decoding iterations. For irregular 1/2 LDPC code, more iteration are required for the code to converge, so maximum decoding iteration is set to 60, and maximum 30Mbps throughput can be achieved.

The measured block error rate verses SNR curve of the three rate is shown in Fig. 5. From the curves we can see that irregular 1/2 design achieves remarkable code gain: block error rate 10^{-4} (bit error rate about 10^{-5}) at SNR 1.4 dB, which outperforms the regular 1/2 code by approximate 0.5 dB. With such amazing error correcting ability, irregular 1/2 mode can be utilized to handle deep fading channel and the worst transmission condition. However, it is also seen from the curve that irregular 1/2 mode encounters the commonly observed error floor problem after 10^{-4} block error rate. This problem is caused by the existence of short cycles in the codes and can be improved in the future through carefully designing the structure of the irregular codes [8]. The measurement curves also show that regular 5/8 and 7/8 modes achieve satisfied results, with threshold SNR at about 3 dB and 6 dB respectively. Although they do not have as good coding gain as irregular 1/2 mode, they enjoy higher throughputs and error floor free capability. So they normally operate at better interference conditions and some particular services.

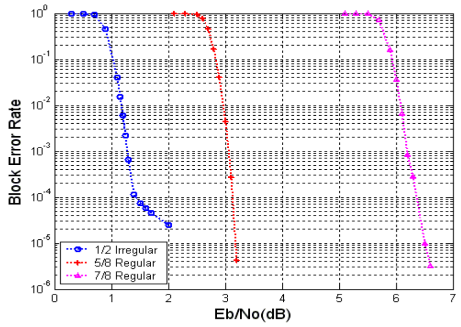


Figure 5. Measurement performance of the decoder

6. CONCLUSIONS

This paper presents a hardware architecture suitable for multi-rate LDPC decoder design. The architecture is implemented on a 10k bit LDPC decoder on the Xilinx XC2V8000 FPGA, and works on three different modes: irregular 1/2, regular 5/8 or regular 7/8. Finite precision effect is also carefully analyzed and the best quantization scheme fit for three rates is found to improve the decoder performance. For the most critical case, irregular 1/2 operating mode can achieve BER 10^{-5} at 1.4dB, which outperform the regular one by about 0.5 dB.

7. REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit Error-Correction Coding and Decoding", *Proc. ICC'93, Geneva, Switzerland,* pp. 1064-1070, May 1993.

[2] A.J. Blanksby and C.J. Howland, "A 690-mW 1-Gb/s 1024-b, Rate-1/2 Low-Density Parity-Check Code Decoder", *IEEE J. Solid-State Circuits,* vol. 37, no. 3, pp. 404-412, March 2002.

[3] T. Zhang and K.K. Parhi, "VLSI Implementation-Oriented (3,k)-Regular Low-Density Parity-Check Codes", *IEEE Workshop on Signal Processing Systems*, pp. 25-36, Sept. 2001

[4] H. Zhang and T. Zhang, "Design of VLSI Implementation-Oriented LDPC Codes", *Vehicular Technology Conference*, vol. 1, pp. 670-673, 2003.

[5] J. Campello, D.S. Modha and S.Rajagopalan, "Designing LDPC Codes Using Bit-Filling*", IEEE International Conf. On Communications*, pp. 55-59, vol. 1, June 2001.

[6] T. Richardson, M.A. Shokrollahi and R.L. Urbanke, "Design of Capacity-Approaching Irregular Low-Density Parity-Check Codes", *IEEE Trans. On Information Theory*, vol. 47, pp. 619-637, Feb 2001.

[7] A. Anastasopoulos, "A Comparison between the Sum-Product and the Min-Sum Iterative Dectection Algorithms Based on Density Evolution", *Global Telecommunications Conference,* vol. 2, pp. 25-29, Nov 2001.

[8] T. Tian, C. Jones, J. D. Villasenor and R. D. Wesel, "Construction of Irregular LDPC Codes with Low Error Floors", *IEEE International Conference on Communication ,* vol. 5 pp. 3125-3129, May 2003.

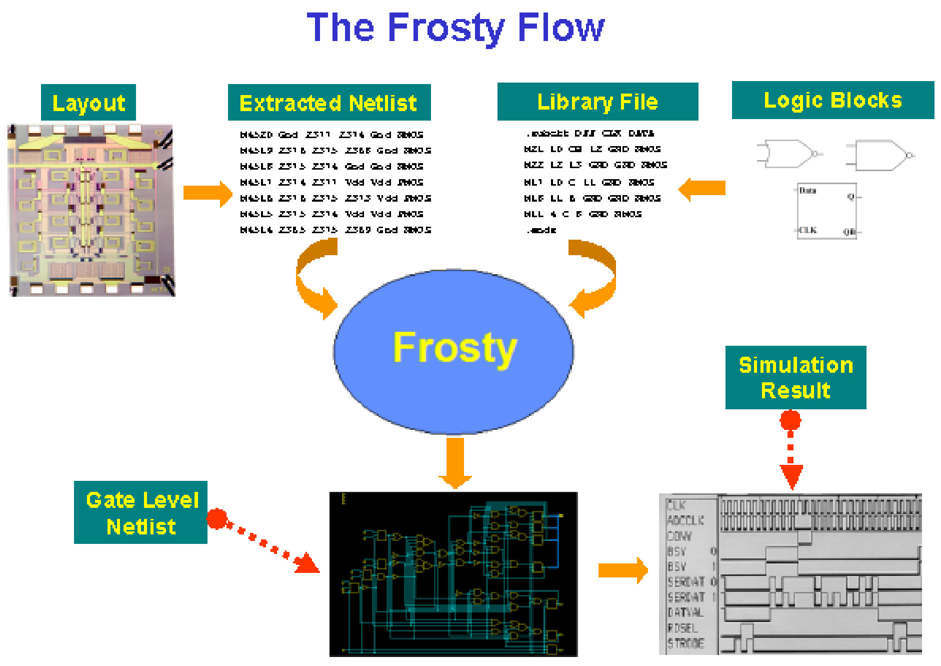
The FROSTY User Guide

Lei Yang
Department of Electrical Engineering
University of Washington
Seattle, WA 98105
May 22, 2003

1. Description

Circuit recognition and extraction is a very important task in VLSI CAD field. This kind of program is widely used in many commercial CAD tools for post-layout simulation, layout function verification, formal verification and design for test.

FROSTY is an automatic CMOS circuit recognition and extraction tool. It reads in two files — file1 and file2. File1 is the description of an ***object circuit***, FROSTY automatically recognize all the standard CMOS gates, such as INV, NAND2, AOI12… in the file1. In file2, user can define some higher level digital blocks, such as DFF, Latch, adder, etc. FROSTY automatically checks whether there are instances of the digital blocks in file1. Finally, FROSTY outputs a Verilog format RTL level netlist and a header file (It contains the functional definitions of all used standard CMOS gates), the two files can be simulated in any digital simulators.



FROSTY read in file format is standard SPICE format, which is compatible with industry standard. The algorithm used by FROSTY is a two-step algorithm. In the first step, structural recognition algorithm is used to recognize all the standard CMOS gates in ***object circuit***. Then in the second step, pattern matching algorithm is performed to extract all the user defined higher level blocks.

FROSTY is written in C++ and runs under the UNIX operating system. It compiles using g++ on Sun SparcStations, but may need some modifications to use other machines and compilers.

FROSTY is research software, please send bug report to yanglei@ee.washington.edu.

2. Usage

There are 2 input files and 2 output files for FROSTY, the file names are specified on the UNIX command line:

%> extractor <flat netlist> <library flie> <Output netlist> <header file>

1) Flat netlist *(input)* — the input flat netlist of the ***object circuit***, SPICE format.
2) Library file *(input)* — SPICE format, user can define digital blocks need to be recognized in this library file.
3) Output netlist *(output)* — Verilog format RTL netlist output from FROSTY.
4) Header file *(output)* — Define the Verilog format behavior model for the standard CMOS gates in the output netlist.

For example, In the working directory, there are 2 files — "PSM.ckt" and "PSM.lib". "PSM.ckt" is the SPICE format description of PSM circuit, and "PSM.lib" defines some digital blocks contained in "PSM.ckt". User can input the following command:

%> extractor PSM.ckt PSM.lib PSM.out header.v

After FROSTY finishes the running, two more output files can be found in the working directory — "PSM.out" and "header.v". "PSM.out" is an RTL level description of PSM circuit, "header.v" defines Verilog format behavior model of all the standard CMOS gates.

3. Installation

The executable file of FROSTY, all the test circuits, one post-layout simulation example are compressed to a “frosty-demo.tar” file. To install FROSTY, just copy the file to your working directory, and run the following command:

%> tar xvf frosty-demo.tar

After running the above command, a new directory call “FROSTY-DEMO” will show up. This directory include the following:

1) Tutorial.pdf

Tutorial file of FROSTY.

2) extractor

This is the executable file of FROSTY, please refer to Part 3 to see the usage of FROSTY.

3) Boeing_Test_Circuit

There are 3 series of test circuits from Boeing in this directory:

--- CEGRP

A series of CEGRP circuits and CEGRP.lib file. In this directory, user can test FROSTY by input the command:

% ../../extractor CEGRP.ckt CEGRP.lib CEGRP.out header.v

or input

% ../../extractor CEGRP3.ckt CEGRP.lib CEGRP.out header.v

--- DFGRP

A series of DFGRP circuits and DFGRP.lib file. To test them, user can input:

% ../../extractor DFGRP4.ckt DFGRP.lib DFGRP.out header.v

--- PSM

A series of PSM circuits and PSM.lib file.

4) PSM-Post_layout_simulation

In this directory, an example - PSM is used to show FROSTY in the flow of post layout simulation. There are 3 directories:

--- VHDL_simulation

PSM VHDL source code and testbench (provided by Boeing). They can be simulated in VHDL simulator, for example, Active-VHDL.

--- Hspice_simulation

PSM Flat netlist extracted from layout (provided by Boeing). It can be simulated in Hspice to get the post-layout waveform.

--- Verilog_simulation

FROSTY extracts all the gates and blocks outputs a Verilog format block level netlist.

4. Input/Output file syntax

The 2 input files are SPICE format:

1) Input file 1

Input file 1 is a flat netlist description of the object circuit. Now FROSTY requires this netlist is a flat netlist. In the future, FROSTY will also support hierarchy netlist.

In this netlist, user need to use ".subckt" command to specify the input ports and output ports of the ***object circuit***. Otherwise, FROSTY can not give the correct port information in the output Verilog format file. Also, in this input netlist, user need to use ".model" file to specify the MOSFET is NMOS or PMOS.

An example is in the following.

```
.subckt sdfa4 a b c e f g Q QBAR
m1 VDD a CLK VDD PMOS L=2.900000e-06 W=6.000000e-07
m2 CLK a GND GND NMOS L=2.900000e-06 W=6.000000e-07
m3 VDD b D VDD PMOS L=2.900000e-06 W=6.000000e-07
m4 1 c GND GND NMOS L=2.900000e-06 W=6.000000e-07
m5 D b 1 GND NMOS L=2.900000e-06 W=6.000000e-07
m6 VDD c D VDD PMOS L=2.900000e-06 W=6.000000e-07
m7 VDD g SCANIN VDD PMOS L=2.900000e-06 W=6.000000e-07
m8 SCANIN g GND GND NMOS L=2.900000e-06 W=6.000000e-07
m9 VDD e 2 VDD PMOS L=2.900000e-06 W=6.000000e-07
m10 TEST f GND GND NMOS L=2.900000e-06 W=6.000000e-07
m11 TEST e GND GND NMOS L=2.900000e-06 W=6.000000e-07
m12 2 f TEST VDD PMOS L=2.900000e-06 W=6.000000e-07
m13 3 4 5 GND NMOS L=1.450000e-06 W=6.000000e-07
m14 6 7 8 VDD PMOS L=1.400000e-06 W=6.000000e-07
m15 8 9 VDD VDD PMOS L=2.200000e-06 W=6.000000e-07
m16 9 6 VDD VDD PMOS L=2.200000e-06 W=6.000000e-07
m17 6 4 10 VDD PMOS L=2.200000e-06 W=6.000000e-07
m18 3 4 11 VDD PMOS L=2.200000e-06 W=6.000000e-07
m19 11 10 VDD VDD PMOS L=2.200000e-06 W=6.000000e-07
m20 10 3 VDD VDD PMOS L=2.200000e-06 W=6.000000e-07
.model NMOS nmos1 level=1 k=0.2 Vth=0.4 lambda=0.002
.model PMOS pmos1 level=1 k=0.2 Vth=0.4 lambda=0.002
```

```
.ends sdfa4
```

2) Input file 2

Input file 2 defines all the digital blocks. The format is also SPICE format, however, keywords ".macromodel" and ".endm" are used to specify the beginning and end of a digital block.

The following is an example of an MUX.

```
.macromodel MUX3 IN0 IN1 IN2 S0 S1 Y
M0 NODEN5 CNODE GND GND NMOS W=2.900U L=0.600U
M1 Y S1BAR NODEN6 GND NMOS W=2.900U L=0.600U
M2 NODEN6 DNODE GND GND NMOS W=2.900U L=0.600U
M3 S1BAR S1 GND GND NMOS W=1.450U L=0.600U
M4 S0BAR S0 GND GND NMOS W=1.450U L=0.600U
M5 CNODE IN2 GND GND NMOS W=1.450U L=0.600U
M6 NODEN3 S0BAR GND GND NMOS W=2.900U L=0.600U
M7 DNODE IN0 NODEN3 GND NMOS W=2.900U L=0.600U
M8 DNODE S0 NODEN4 GND NMOS W=2.900U L=0.600U
M9 NODEN4 IN1 GND GND NMOS W=2.900U L=0.600U
M10 Y S1 NODEN5 GND NMOS W=2.900U L=0.600U
M11 NODEP5 CNODE VDD VDD PMOS W=3.050U L=0.600U
M12 Y DNODE NODEP6 VDD PMOS W=3.050U L=0.600U
M13 NODEP6 S1 VDD VDD PMOS W=3.050U L=0.600U
M14 S1BAR S1 VDD VDD PMOS W=2.200U L=0.600U
M15 S0BAR S0 VDD VDD PMOS W=2.200U L=0.600U
M16 CNODE IN2 VDD VDD PMOS W=2.200U L=0.600U
M17 NODEP3 S0 VDD VDD PMOS W=3.050U L=0.600U
M18 DNODE IN0 NODEP3 VDD PMOS W=3.050U L=0.600U
M19 DNODE S0BAR NODEP4 VDD PMOS W=3.050U L=0.600U
M20 NODEP4 IN1 VDD VDD PMOS W=3.050U L=0.600U
M21 Y S1BAR NODEP5 VDD PMOS W=3.050U L=0.600U
.model NMOS nmos1 level=1 k=0.2 Vth=0.4 lambda=0.002
.model PMOS pmos1 level=1 k=0.2 Vth=0.4 lambda=0.002
.endm MUX3
```

The 2 output files are Verilog format.

3) Output file 1

Output file 1 is a Verilog format RTL level netlist. The following is an example:

```
module sdfa4 (g, f, e, c, b, a, QBAR, Q);
input g, f, e, c, b, a;
output QBAR, Q;
 INV  U1 ( .a(a), .out(CLK) );
 NOR2  U2 ( .a(e), .b(f), .out(TEST) );
 NAND2  U3 ( .a(b), .b(c), .out(D) );
 INV  U4 ( .a(g), .out(SCANIN) );
 DFF  U5 ( .SCANIN(SCANIN), .CLK(CLK), .D(D), .TEST(TEST), .QBAR(QBAR), .Q(Q) );
Endmodule
```

4) Output file 2

Output file 2 defines the behavior models of all the standard CMOS gates. The following is an example of an NOR2:

```
module NAND2(a, b, out);
input a, b;
output out;
assign out = ~(a & b);
endmodule
```

Notice here FROSTY only defines all the behavior models of the standard CMOS gates. For simulation purpose, use should manually add the behavior description of the digital blocks that are defined in the library file. For example, if user defines an DFF in the library file, he should write the Verilog model of the DFF, shown in the following:

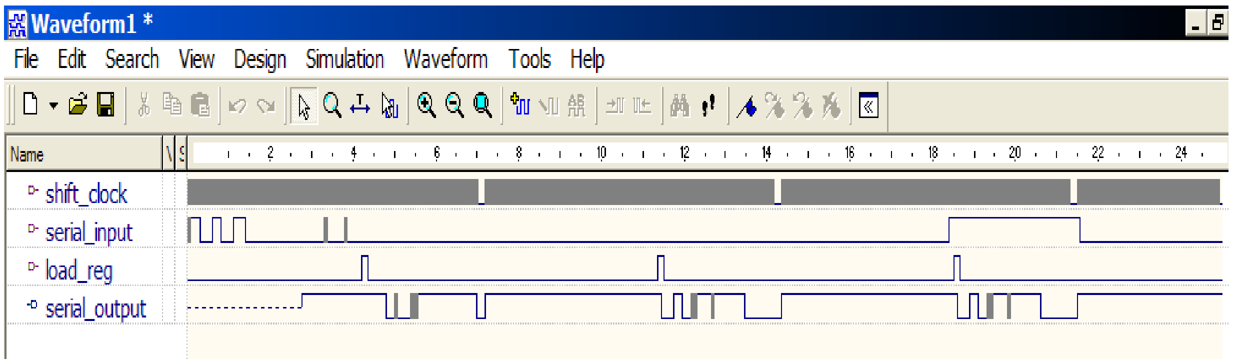
```
module DFFP(DATA, CLK, PRB, Q, QB);
input DATA, CLK, PRB;
output Q, QB;
reg Q, QB;
always @(posedge CLK or negedge PRB)
begin
  if (PRB == 1'b0)
    begin
     Q <= 1'b1;
     QB <= 1'b0;
    end
  else
   begin
    Q <= DATA;
    QB <= ~DATA;
   end
end
endmodule
```

3. An example to shown the flow of post layout simulation

In this part, an example, PSM, is illustrated to show how to use FROSTY in the flow of post layout simulation.

1) VHDL simulation

PSM is written in VHDL in Boeing. Using the PSM VHDL source code and testbench, we can simulate them in Active-VHDL and get the following waveform:

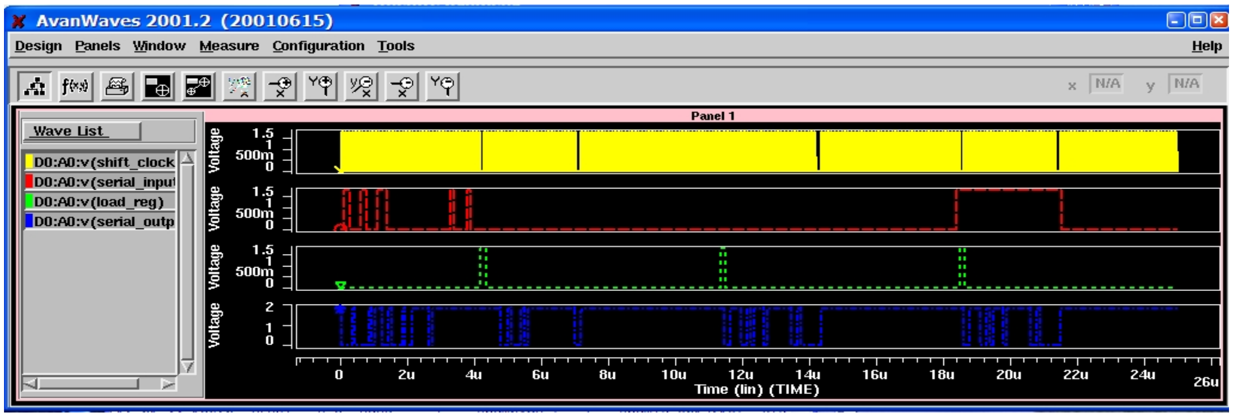


2) HSPICE simulation

Also Boeing provides the flat netlist of PSM extracted from PSM layout. To run the HSPICE simulation, just go to the directory "./FROSTY-DEMO/PSM-Post_layout_simulation/Hspice_simulation" and run the following command:

% hspice psm.sp

This Hspice simulation will take almost 2 hours in our Workstation (900M CPU, 16GB RAM). You can use AWAVES to see the waveform:



3) Verilog simulation

To run the Verilog simulation, we need to get the PSM circuit RTL level netlist from FROSTY, go to directory: "FROSTY-DEMO/Boeing_Test_Circuit/PSM " , run the following command:

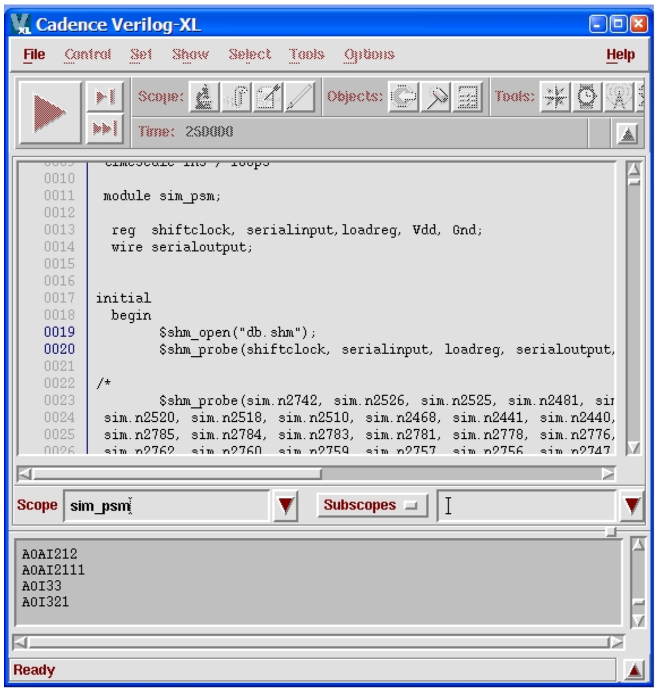
% ../../extractor PSM.ckt PSM.lib PSM.out header.v

After the running, you will see FROSTY output 2 files, one is RTL level netlist "PSM.out", the other one is gate model definition file "header.v", which define the Verilog model of the standard CMOS gates, such as INV, NOR2…..

Copy the above two files "PSM.out" "header.v" to the directory "FROSTY-DEMO/PSM-Post_layout_simulation/Verilog_simulation". In this directory, we have two other files "Subcircuit.model" "sim_psm.v". "Subcircuit.model" define the Verilog model of the higher level blocks in PSM design, such as DFF, latch, MUX and so on. "sim_psm.v" is the simulation file for PSM design. Run the following command:

% verilog sim_psm.v Subcircuit.model header.v PSM.out +gui &

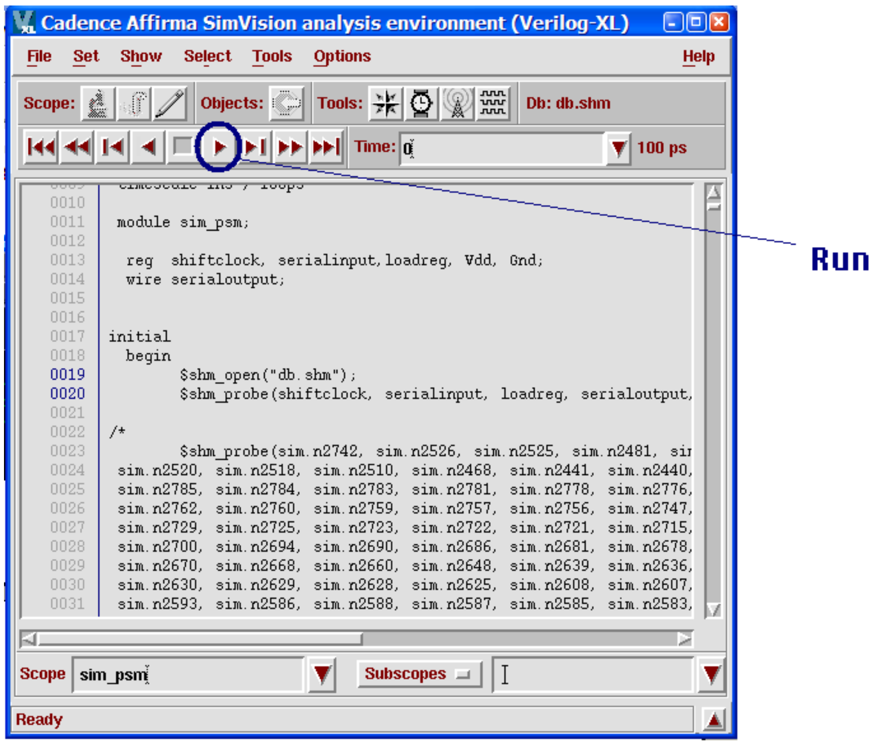
We can see the Verilog-XL GUI interface show up after running the above the command, shown in the following.



```
 module sim_psm;

  reg  shiftclock, serialinput,loadreg, Vdd, Gnd;
  wire serialoutput;


initial
  begin
        $shm_open("db.shm");
        $shm_probe(shiftclock, serialinput, loadreg, serialoutput,

/*
        $shm_probe(sim.n2742, sim.n2526, sim.n2525, sim.n2481, si
 sim.n2520, sim.n2518, sim.n2510, sim.n2468, sim.n2441, sim.n2440,
 sim.n2785, sim.n2784, sim.n2783, sim.n2781, sim.n2778, sim.n2776,
```

```
AOAI212
AOAI2111
AOI33
AOI321
```

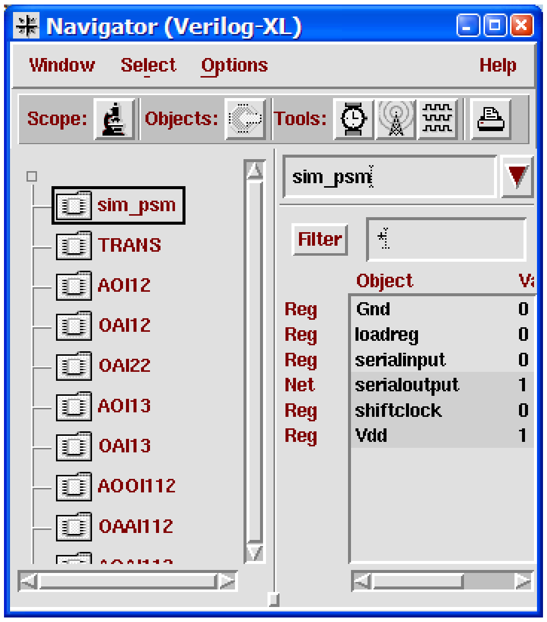
In the GUI interface, select “File-> Post Processing”, a window will pop up, click “Yes”, then we can enter into the “Post Processing” enviroment. Shown in the following:


Cadence Affirma SimVision analysis environment (Verilog-XL)
File Set Show Select Tools Options Help
Scope: Objects: Tools: Db: db.shm
Time: 0 100 ps
Run
Scope sim_psm Subscopes
Ready

```
 module sim_psm;

  reg  shiftclock, serialinput,loadreg, Vdd, Gnd;
  wire serialoutput;


initial
  begin
        $shm_open("db.shm");
        $shm_probe(shiftclock, serialinput, loadreg, serialoutput,

/*
        $shm_probe(sim.n2742, sim.n2526, sim.n2525, sim.n2481, si
 sim.n2520, sim.n2518, sim.n2510, sim.n2468, sim.n2441, sim.n2440,
 sim.n2785, sim.n2784, sim.n2783, sim.n2781, sim.n2778, sim.n2776,
 sim.n2762, sim.n2760, sim.n2759, sim.n2757, sim.n2756, sim.n2747,
 sim.n2729, sim.n2725, sim.n2723, sim.n2722, sim.n2721, sim.n2715,
 sim.n2700, sim.n2694, sim.n2690, sim.n2686, sim.n2681, sim.n2678,
 sim.n2670, sim.n2668, sim.n2660, sim.n2648, sim.n2639, sim.n2636,
 sim.n2630, sim.n2629, sim.n2628, sim.n2625, sim.n2608, sim.n2607,
 sim.n2593, sim.n2586, sim.n2588, sim.n2587, sim.n2585, sim.n2583,
```

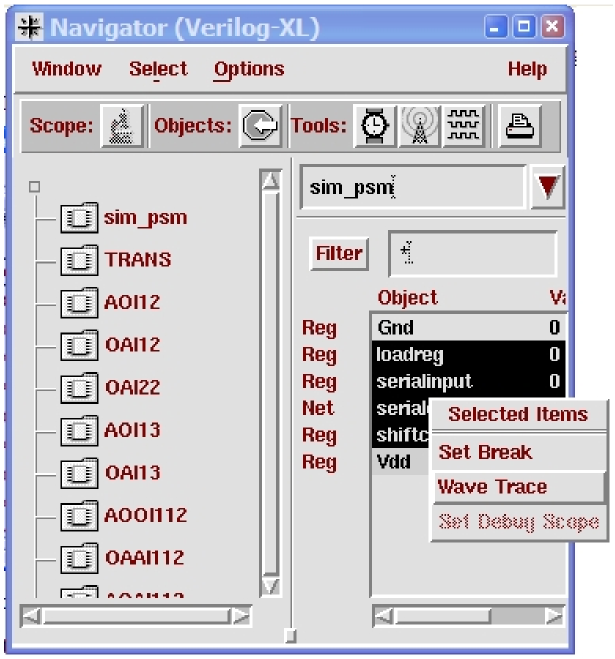
In the Post Processing window, click “Run” button, shown in the above figure, Verilog begins the simulation, we can see the simulation time in the Terminal window, about 0.5 s.

After the simulation is done, we can see the waveforms. Go to “Tools->Navigator”, the following window will pop up.

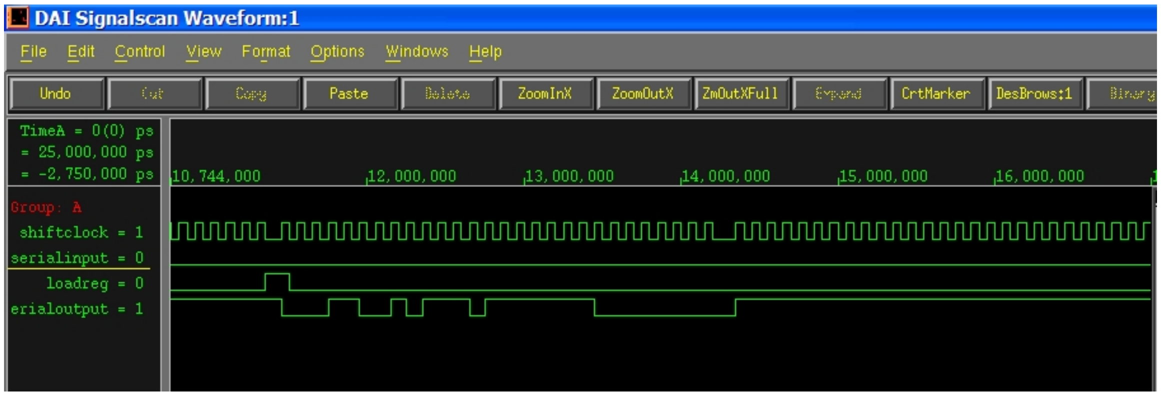


	Object	Va
Reg	Gnd	0
Reg	loadreg	0
Reg	serialinput	0
Net	serialoutput	1
Reg	shiftclock	0
Reg	Vdd	1

In the Navigator window, click the “sim_psm” in the left frame, so there are some signals show up in the right window. Select the signals and click right button, select “”, shown in the following figure:



Then the waveform window will show up. Click the "ZmOutXFull", you can see the waveform show in the following, this waveform is totally same as the VHDL simulation result.



Summary:

From the above flow, we can see that using FROSTY, the post-layout time can be saved greatly. However, in this flow, the real delay information of the design is not inserted, which will be a future work of FROSTY.

A COUPLED ITERATIVE/DIRECT METHOD FOR EFFICIENT TIME-DOMAIN SIMULATION OF NONLINEAR CIRCUITS WITH POWER/GROUND NETWORKS[*]

Zhao Li and C.-J. Richard Shi
Department of Electrical Engineering, University of Washington
Seattle, WA 98195
{lz2000, cjshi}@ee.washington.edu

Abstract: A coupled iterative/direct circuit analysis method is proposed for efficient SPICE-accurate time-domain simulation of nonlinear circuits with large-scale power/ground networks. The system under study is partitioned into a linear part including power/ground networks, a nonlinear part, and an interface between them. The part of power/ground networks is formulated by nodal analysis based on *RCLK* elements, and solved by an efficient conjugate gradient iterative method with an incomplete Cholesky decomposition preconditioner. The nonlinear circuit part is formulated by modified nodal analysis, and solved by the direct method as in SPICE. The iterative method and the direct method are coupled by a Gauss-Seidel like relaxation scheme with SPICE built-in varying time step-size numerical integration. How the condition number of a circuit matrix changes with time step-sizes is further studied. Experimental results on digital circuits with power/ground networks demonstrate that the proposed coupled iterative/direct method yields SPICE-like accuracy with orders of magnitude speedup for circuits with tens of thousands elements.

1. INTRODUCTION

With the increasing operation frequency, lower supply voltage and smaller device feature size, the effects of power/ground networks, such as *Ldi/dt* drop, *IR* drop, resonance, are becoming more and more pronounced [5]. An improper circuit design neglecting power/ground networks and packaging will result in excessive voltage drops and fluctuations in circuit supply nodes. The noise margin for digital circuits is therefore reduced, which may unfortunately disturb gate delays or even produce logic errors. The increasing demand to integrate digital, analog and radio frequency (RF) circuits into one single chip requires accurate analysis of VLSI circuits together with power/ground networks [1][3][5][9]. For such purposes as well as high fidelity coupled circuit and electromagnetic modeling [7], SPICE-like simulators are desirable for accurate transistor-level time-domain simulation.

However, efficient simulation of such systems presents a complexity challenge to SPICE [4]. To accomplish transient simulation, SPICE uses numerical integration formulae at each time point and applies the Newton-Raphson (NR) method to linearize nonlinear devices. Then the circuit system is simulated at each time point by solving a system of linear equations $\boldsymbol{Ax} = \boldsymbol{b}$, where $\boldsymbol{A}$ is typically in the form of a so-called modified nodal analysis (MNA) circuit matrix. It is well known that device evaluation dominates the simulation of small to medium scale circuits and can be speeded up using table-lookup nonlinear device models or parallel computation techniques. However, for large scale nonlinear circuits coupled with power/ground networks, the per-iteration cost of transient simulation with SPICE is dominated by LU factorization of the circuit matrix $\boldsymbol{A}$.

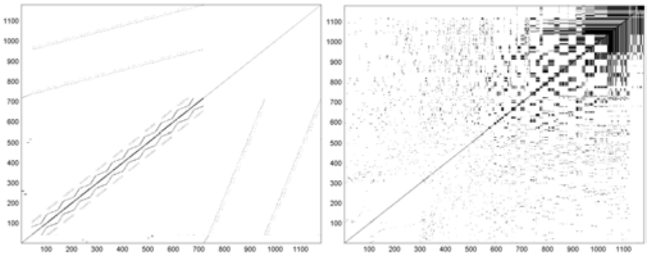


Figure 1. The circuit matrix structure of a power/ground example (a) before LU factorization and (b) after LU factorization.

Figure 1 (a) and Figure 1 (b) show the circuit matrix structure before and after LU factorization for a power/ground analysis example in Section 4. It can be seen that the original circuit matrix before factorization is very regular and sparse (9618 elements in a 1177x1177 matrix, which means the sparsity is 0.70%), while the matrix after factorization becomes irregular and much denser (89733 elements, the element number is increased 9.33*X* due to fill-ins and the sparsity becomes 6.68%). Therefore, a key idea to achieve speedup and save memory is to apply efficient krylov-subspace based iterative methods [1][5][9] on power/ground networks analysis since those methods only require matrix-vector multiplications on the original sparse circuit matrix.

Although iterative methods have been shown to be efficient for transient simulation of large-scale power/ground networks [1], their application to general nonlinear circuits is limited. The reason is that the circuit matrix for a nonlinear circuit is typically not symmetric positive definite, which prohibits the usage of efficient preconditioners for iterative methods. Iterative methods without good preconditioners are well known to have the convergence problem. The direct method based on the Newton-Raphson iteration as in SPICE is still the most efficient way for general nonlinear circuit simulation.

Noticing different application areas of iterative and direct methods, we present a new coupled iterative/direct method capable of analyzing nonlinear circuits with power/ground networks *in SPICE-like accuracy yet orders of magnitude speedup*. The system under study is partitioned into three parts – a linear part including power/ground networks, a nonlinear part and an interface between them. Two key ideas are:

1) For power/ground networks, nodal analysis (NA) formulation of *RCLK* elements is applied so that an efficient iterative conjugate gradient method with an incomplete Cholesky decomposition preconditioner [1][6] can be used. For different circuit formulation methods, how the condition number of a circuit matrix changes with time step-sizes is further studied.
2) For nonlinear circuits, the modified nodal analysis (MNA) formulation is applied and the direct method as in SPICE is

[*] This research was supported by DARPA NeoCAD Program under Grant No. N66001-01-8920, NSF-SRC Joint Initiative on Mixed Signal Electronic Technologies under Grant No. CCR-0120371, and NSF CAREER Award under Grant No. 9985507.

used. The iterative method and the direct method are coupled together by a Gauss-Seidel style relaxation scheme [8].

This paper is organized as follows. Section 2 proposes the new coupled iterative/direct method. The NA formulation for iterative methods and the condition number variation with time step-sizes are described in Section 3. Experimental results on digital circuits with power/ground networks are shown in Section 4. Section 5 concludes this paper.

2. THE COUPLED ITERATIVE/DIRECT METHOD

The system under study is shown in Fig. 2, in which power/ground networks are coupled with nonlinear circuits through a linear interface. It can be seen that parasitic coupling effects between the power network and the ground network are also incorporated. The linear interface is constructed so that only a few linear elements (such as resistors connecting grid nodes of power/ground networks and supply nodes of nonlinear circuits) are introduced.

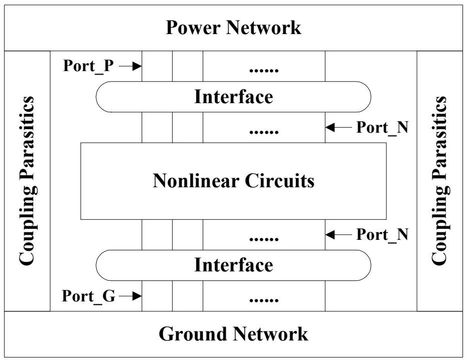


Figure 2. Nonlinear circuits coupled with power/ground networks.

Figure 3 shows the related circuit matrix structure for the system in Fig. 2. $\boldsymbol{Y_{PG}}$, $\boldsymbol{Y_N}$ and $\boldsymbol{Y_I}$ represent circuit matrices of power/ground networks, nonlinear circuits, and the interface, respectively. $\boldsymbol{C_{PG}}$ and $\boldsymbol{C_{PG}^T}$ are coupling matrices between power/ground networks and the interface. $\boldsymbol{C_N}$ and $\boldsymbol{C_N^T}$ are coupling matrices between nonlinear circuits and the interface. All other parts in the circuit matrix are zero. The unknown variables $\boldsymbol{v}$ and the right hand side (RHS) vectors $\boldsymbol{b}$ are also labeled accordingly.

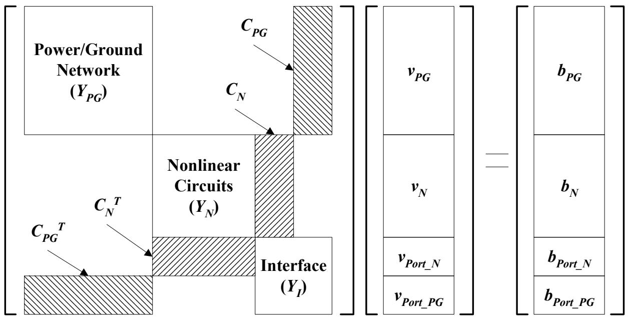


Figure 3. The circuit matrix structure for a system in Fig. 2.

The circuit matrix $\boldsymbol{Y_I}$ for the interface can be further partitioned as below,

$$\boldsymbol{Y_I} = \begin{bmatrix} \boldsymbol{Y_{INN}} & \boldsymbol{Y_{INP}} \\ \boldsymbol{Y_{IPN}} & \boldsymbol{Y_{IPP}} \end{bmatrix}$$

where $\boldsymbol{Y_{INN}}$ and $\boldsymbol{Y_{IPP}}$ are self-admittance matrices for the ports of nonlinear circuits and those of power/ground networks, respectively, $\boldsymbol{Y_{IPN}}$ and $\boldsymbol{Y_{INP}}$ are coupling matrices between the ports of nonlinear circuits and those of power/ground networks. In general, $\boldsymbol{Y_{IPN}} = \boldsymbol{Y_{INP}^T}$.

To introduce the Gauss-Seidel style relaxation scheme [8], we regroup circuit sub-matrices and sub-vectors as below,

$$\boldsymbol{Y}_{\boldsymbol{PG}}^{*} = \begin{bmatrix} \boldsymbol{Y_{PG}} & \boldsymbol{C_{PG}} \\ \boldsymbol{C_{PG}^T} & \boldsymbol{Y_{IPP}} \end{bmatrix} \boldsymbol{Y}_{\boldsymbol{N}}^{*} = \begin{bmatrix} \boldsymbol{Y_N} & \boldsymbol{C_N} \\ \boldsymbol{C_N^T} & \boldsymbol{Y_{INN}} \end{bmatrix}$$

$$\boldsymbol{v}_{\boldsymbol{PG}}^{*} = \begin{bmatrix} \boldsymbol{v_{PG}} \\ \boldsymbol{v_{Port-PG}} \end{bmatrix} \boldsymbol{v}_{\boldsymbol{N}}^{*} = \begin{bmatrix} \boldsymbol{v_N} \\ \boldsymbol{v_{Port-N}} \end{bmatrix}$$

$$\boldsymbol{b}_{\boldsymbol{PG}}^{*} = \begin{bmatrix} \boldsymbol{b_{PG}} \\ \boldsymbol{b_{Port-PG}} \end{bmatrix} \boldsymbol{b}_{\boldsymbol{N}}^{*} = \begin{bmatrix} \boldsymbol{b_N} \\ \boldsymbol{b_{Port-N}} \end{bmatrix}$$

Therefore, the circuit matrix in Fig. 3 can be further written in the following format,

$$\boldsymbol{Y}_{\boldsymbol{PG}}^{*}\boldsymbol{v}_{\boldsymbol{PG}}^{*} = \boldsymbol{b}_{\boldsymbol{PG}}^{*} - \boldsymbol{Y_{IPN}}\boldsymbol{v_{Port-N}} \quad (1)$$

$$\boldsymbol{Y}_{\boldsymbol{N}}^{*}\boldsymbol{v}_{\boldsymbol{N}}^{*} = \boldsymbol{b}_{\boldsymbol{N}}^{*} - \boldsymbol{Y_{INP}}\boldsymbol{v_{Port-PG}} \quad (2)$$

According to Eq. (1), once $\boldsymbol{v_{Port\text{-}N}}$ is fixed, $\boldsymbol{v_{PG}}^*$ can be solely solved. After $\boldsymbol{v_{PG}}^*$ (and therefore $\boldsymbol{v_{Port\text{-}PG}}$) is given, $\boldsymbol{v_N}^*$ (and therefore $\boldsymbol{v_{Port\text{-}N}}$) can be determined by Eq. (2). Then, the new $\boldsymbol{v_{Port\text{-}N}}$ is compared to the old $\boldsymbol{v_{Port\text{-}N}}$ used during solving Eq. (1) to check if this *Gauss-Seidel relaxation* is converged. The coupled iterative/ direct method is summarized in Table I.

Table I. The coupled iterative/direct method.

```
INITIALIZATION:
    Construct $Y_{INP}$ and $Y_{IPN}$
    $t$=0
WHILE ($t<T_{\text{final}}$){
    OUTER LOOP: do{
        Construct matrix $Y_{PG}^*$ and vector $b_{PG}^*$
        Apply ICD-CG to compute $v_{PG}^*$ based on $v_{Port\text{-}N}$ using Eq. (1)
        INNER LOOP: do{
            Construct matrix $Y_N^*$ and vector $b_N^*$
            Apply NR linearization and solve Eq. (2)
        } while ($v_N^*$ not converge)
    } while ($v_{Port\text{-}N}$ not converge)
    Determine the next time step-size $h_n$
    $t = t + h_n$
}
```

It can be seen that the costly simulation of power/ground networks is in the outer loop, while the cheap simulation of nonlinear circuits is in the inner loop. The number of inner nonlinear iterations under the Gauss-Seidel relaxation scheme is generally higher than that with SPICE, since several outer iterations may be required to achieve the final convergence. Even so, a great simulation speedup is still achievable since the cost of each inner nonlinear iteration is much lower than that of one SPICE nonlinear iteration. The reason is that the size of nonlinear circuits is reduced greatly with power/ground networks decoupled in our scheme. Further, the simulation of nonlinear circuits can be speeded up using table-lookup nonlinear device models or parallel computation techniques.

3. ITERATIVE METHODS WITH NA FORMULATION

The MNA formulation for circuit elements is widely used in modern circuit simulators based on direct methods. However, as shown in the Section 1, the simulation of power/ground networks presents a challenge for direct methods. Therefore, the NA formulation of *RCL* elements has been applied for power/ground network simulation based on iterative methods [1]. The NA formulation of *R* and *C* is the same as their MNA formulation. The

NA formulation of *L* with the trapezoid numerical integration formula is shown in Fig. 4. It can be seen that the equivalent conductance is ***h**$_n$**/(2L)*** rather than ***(2L)/ h**$_n$* in the MNA formulation. The mutual inductance can be incorporated easily by so called *K*-elements [2]. It has been proved that the circuit matrix with *RCLK* elements based on the NA formulation is symmetric positive definite [1][2]. Therefore, we have implemented the conjugate gradient (CG) method with an incomplete Cholesky decomposition preconditioner [1][6] (named by ICD-CG).

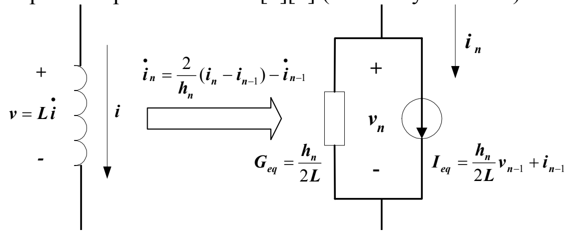

$v = L\dot{i}$
i
+
-
$\dot{i}_n = \frac{2}{h_n}(i_n - i_{n-1}) - \dot{i}_{n-1}$
$G_{eq} = \frac{h_n}{2L}$
v_n
i_n
$I_{eq} = \frac{h_n}{2L} v_{n-1} + i_{n-1}$

Figure 4. NA formulation of a linear inductor.

The *RCL* circuit example in Fig. 5, the structure of which is typical in power/ground networks, is used to study how the condition number of a circuit matrix changes with time step-sizes. As shown in Fig. 6, the condition number for the MNA formulation is becoming worse as the time step-size decreases (***h**$_n$* is less than 1). The reasons are: *1*) The MNA formulation of voltage sources introduces zero diagonal elements; *2*) The self-admittance matrix element at node 1 is only contributed by a fixed resistor. Therefore, a tighter tolerance is required when time step-size becomes smaller with iterative methods [5]. If the voltage source *E* and the serial resistor R_1 in Fig. 5 are replaced by an equivalent Norton current source and a parallel resistor, the condition number is kept relatively small with time step-sizes cahnged, as shown in Fig. 6. Unfortunately, it is not suitable for iterative methods since the MNA formulation of a linear inductor either introduces a negative diagonal element or causes the circuit matrix asymmetric.

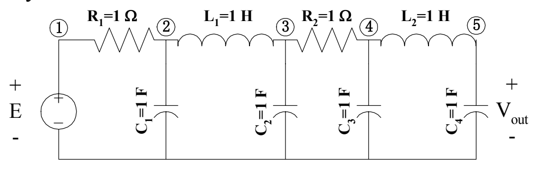


Figure 5. A *RCL* circuit example.

As mentioned previously, the circuit matrix with the NA formulation is symmetric positive definite, which should be suitable for iterative methods. However, in Fig. 6 the condition number for the NA formulation is becoming worse with the time step-size increased (***h**$_n$* is larger than 1). The reason is that the effects of linear inductors become ignorable with an enlarged time step-size – an enlarged equivalent conductance of ***h**$_n$**/(2L)*** means a reduced equivalent resistance. In this case, linear inductors are close to short branches, which will cause excessive numerical errors with the NA formulation. Therefore, proper window-based truncation techniques on inductance matrices [2] should be applied before using the NA formulation so that ignorable (mutual) inductors are not present.

Once ignorable (mutual) inductances are truncated, it will be safe enough to use the NA formulation since the time step-size ***h**$_n$* is determined by time constants with relatively small values in a circuit. Figure 7 shows the histogram of SPICE time step-sizes for the *RCL* circuit in Fig. 5. It can be seen that most time step-sizes are less than 1 and some are even less than 0.1. Therefore, the condition number of the circuit matrix is required to be relatively small for ***h**$_n$* less than 1. According to Fig. 6, the NA formulation does ensure the condition number of the circuit matrix relatively small when the time step-size is decreased for ***h**$_n$* less than 1.

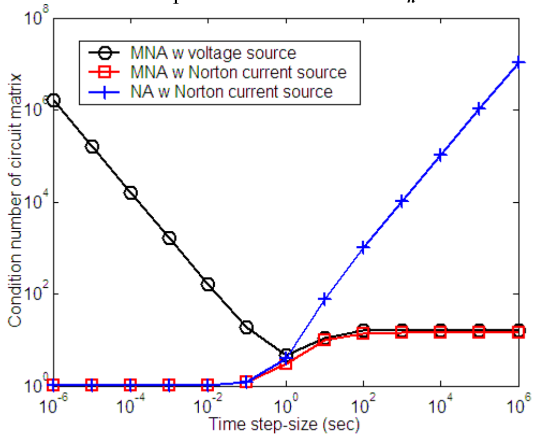


Figure 6. The condition number variation with time step-sizes.

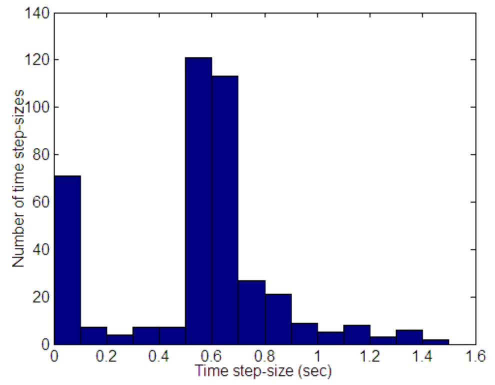


Figure 7. The histogram of time step-sizes.

4. EXPERIMENTAL RESULTS

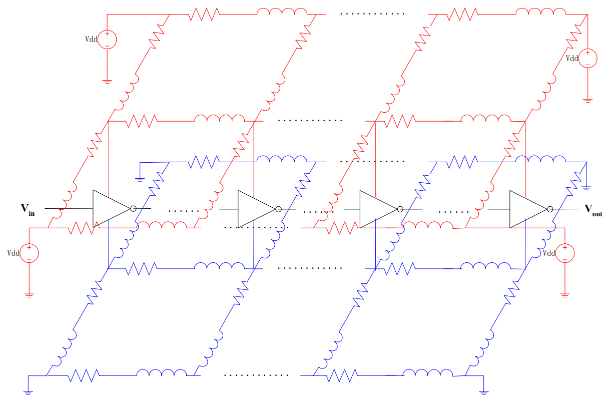


Figure 8. The power/ground analysis example.

In Fig. 8. the power and ground supply networks are modeled as two *RCL* mesh layers (parasitic coupling capacitors are not shown in Fig. 8). Between these two layers is a 20-stage inverter chain, different inverters of which are connected to different power/ground grid nodes. Furthermore, *RCL* loads are added for each inverter to model interconnect lines between adjacent stages.

Figure 9 shows the transient output waveform of the inverter chain when the output signal is digital “1” (the high voltage level). The “1” signal has been disturbed due to the *IR*-drop (the input

Vdd is 3.3v) and *L*dI/dt* effects of the power/ground network. Table II shows the simulation results with varied numbers of elements modeling the power/ground network. In our experiments, the size of two *RCL* meshes is changed to vary the number of elements. The run time comparison between SPICE3 and the proposed method with the tolerance of the iterative method set to 1e-6 and 1e-8 is shown in Fig. 10. We can see that the coupled iterative/direct method achieves more speedup for larger circuits. The maximum overall speed-up reach *85.39X* and *16.74X* (with about 60 thousand elements) with the tolerance set to 1e-6 and 1e-8, respectively. The speedup is comparable to a recent explored direct method [3].

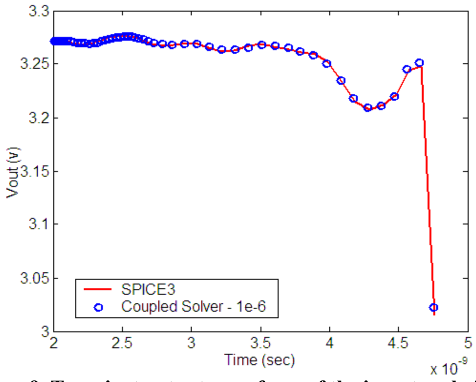


Figure 9. Transient output waveform of the inverter chain for power/ground analysis example.

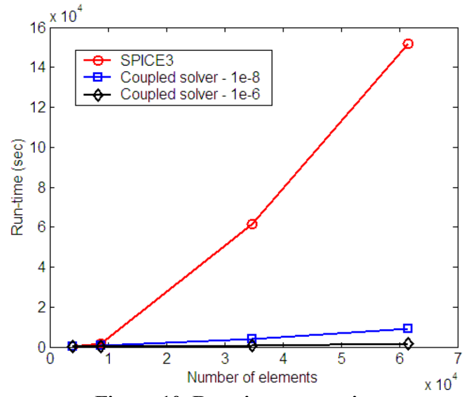


Figure 10. Run time comparison.

It can be seen from Table II that the number of outer Gauss-Seidel iterations is typically increased to 4*X* to 5*X* of that of SPICE nonlinear iterations. When the tolerance is set to 1e-6, the average number of CG iterations for each Gauss-Seidel step is 6.5 to 9, and it becomes 22 to 45 if the tolerance is set to 1e-8. The number of CG iterations increases dramatically to achieve high accuracy, i.e., when the tolerance is set to 1e-8. One way to improve the performance of iterative methods on large-scale power/ground networks for high accuracy is to apply multigrid-like methods [9].

5. CONCLUSION

A coupled iterative/direct time-domain circuit analysis method has been proposed for nonlinear circuits coupled with large-scale power/ground networks. Nodal analysis formulation of *RCLK* elements is applied on power/ground networks and an efficient iterative conjugate gradient method with an incomplete Cholesky decomposition preconditioner is used. Modified nodal analysis formulation is applied on nonlinear circuits and the direct method based on the Newton-Raphson iteration is used. The iterative method and the direct method are coupled together by a Gauss-Seidel style relaxation scheme. We further studied how the condition number of a circuit matrix changes with time step-sizes. Experimental results on digital circuits with power/ground networks show that the proposed method yields SPICE-like accuracy with orders of magnitude speedup over SPICE3.

REFERENCES

[1] T. Chen and C. C.-P. Chen, "Efficient Large-Scale Power Grid Analysis based on Preconditioned Krylov-subspace Iterative Methods", *Proc. IEEE/ACM Design Automation Conference*, pp. 559-562, June 2001.

[2] T. Chen, C. Luk, and C. C.-P. Chen, "INDUCTWISE: Inductance-Wise Interconnect Simulator and Extractor", *IEEE Trans. on CAD*, vol. 22, no. 7, pp.884-894, July 2003.

[3] Z. Li and C.-J. R. Shi, "SILCA: Fast-Yet-Accurate Time-Domain Simulation of VLSI Circuits with Strong Parasitic Coupling Effects", *Proc. IEEE/ACM Int. Conf. on Computer-Aided Design*, pp. 793-799, Nov. 2003.

[4] L. W. Nagel, *SPICE: A Computer Program to Simulate Semiconductor Circuits*, University of California, Berkeley, Tech. Rep., UCB/ERL M520, May 1975.

[5] J. R. Phillips and L. M. Silveira, "Simulation Approaches for Strongly Coupled Interconnect Systems", *Proc. IEEE/ACM Int. Conf. on Computer-Aided Design*, pp. 430-437, November 2001.

[6] Y. Saad, "*Iterative Methods for Sparse Linear Systems*", 2nd Edition, SIAM, 2003.

[7] Y. Wang, V. Jandhyala, and C.-J. R. Shi, "Coupled Electromagnetic-Circuit Simulation of Arbitrarily-Shaped Conducting Structures", *Proc. IEEE Conf. on Electrical Performance of Electronic Packaging*, pp. 233-236, October 2001.

[8] J. K. White and A. Sangiovanni-Vincentelli, *Relaxation Techniques for the Simulation of VLSI Circuits*, Kluwer Academic Publishers, 1987.

[9] J. N. Kozhaya, S. R. Nassif, and F. N. Najm, "A Multigrid-like Technique for Power Grid Analysis", *IEEE Trans. on CAD*, vol. 21, no. 10, pp. 1148-1160, Oct. 2002.

Table II. Simulation results for the power/ground analysis example.

#Elems	SPICE3		Coupled Solver						Speedup	
			ε=1e-6			ε=1e-8			ε= 1e-6	ε= 1e-8
	#Iter	Overall (sec)	#GS Iter	#CG Iter	Overall (sec)	#GS Iter	#CG Iter	Overall (sec)		
4002	4016	456.51	17363	113832	79.30	17350	384444	242.43	5.76	1.88
8851	4171	1.75e3	16432	128561	183.34	16205	475429	647.25	9.55	2.70
34802	3986	6.17e4	17679	156038	869.80	17868	772263	3.88e3	70.94	15.90
61602	4377	1.52e5	20208	162416	1.78e3	22335	1002260	9.08e3	85.25	16.74

SILCA: Fast-Yet-Accurate Time-Domain Simulation of VLSI Circuits with Strong Parasitic Coupling Effects[*]

Zhao Li and C.-J. Richard Shi
Department of Electrical Engineering, University of Washington
Seattle, WA 98195
{lz2000, cjshi}@ee.washington.edu

Abstract: We propose a new circuit analysis method, namely Semi-Implicit Linear-Centric Analysis (*SILCA*), for efficient SPICE-accurate transient simulation of deep-submicron VLSI circuits with strong parasitic coupling effects introduced by interconnect lines, common substrate, power/ground networks, etc. *SILCA* is based on two linear-centric techniques. First, a new semi-implicit iterative numerical integration scheme is developed, which applies dynamic time step control accounting for stiff systems and meanwhile keeps constant equivalent conductance for capacitor/inductor companion models. Its convergence and stability properties are characterized. Second, to achieve constant linearized conductance for nonlinear devices during nonlinear iteration process, a successive variable chord method is introduced as an alternative of the Newton-Raphson method and the rank-one update technique is implemented for fast LU factorization. With these techniques, *SILCA* reduces the number and cost of required LU factorizations dramatically. Experimental results on substrate and power/ground networks have demonstrated that *SILCA* yields SPICE-like accuracy with an over 80*X* reduction in LU factorization cost, and an about 20*X* overall CPU time speedup over SPICE3 for circuits with tens of thousands elements, and the efficiency increases further with the size of a circuit.

1. Introduction

With the increasing operation frequency, lower supply voltage and smaller device feature size, parasitic coupling effects are becoming more and more important in modern deep-submicron VLSI circuit designs [1]. The increasing demand to integrate digital, analog and radio frequency (RF) circuits into one single chip requires accurate analysis of VLSI circuits together with surrounding environments, such as interconnect lines, common substrate, power/ground networks, on-chip and packaging inductance, etc. [1][2][3][4][14]. For such purpose, as well as high fidelity coupled circuit and electromagnetic modeling [16], SPICE-like simulators are desirable for accurate transistor-level time-domain simulation.

However, efficient simulation of such systems presents a complexity challenge to SPICE [5]. To accomplish transient simulation, SPICE uses numerical integration formulae [6][7] to form companion models for capacitors and inductors at each time point, and applies the Newton-Raphson (NR) method [6] to linearize nonlinear devices. Then the circuit system is simulated at each time point by iteratively solving a system of linear equations $\boldsymbol{Ax} = \boldsymbol{b}$, where $\boldsymbol{A}$ is typically a so-called modified nodal analysis (MNA) circuit matrix [5][6]. For strongly coupled systems, the per-iteration cost of transient simulation with SPICE is dominated by LU factorization [6] of circuit matrix $\boldsymbol{A}$. The practical cost for LU factorization by using sparse matrix solvers [8] is $O(n^{1.1\sim1.5})$ for sparse circuits, where n is the circuit matrix size. However, considering strong coupling effects present in deep sub-micron circuits, since the circuit matrix can become much denser, even with model order reduction [10], the cost for LU factorization can approach its worst case $O(n^3)$ [1].

A key idea to improve the efficiency of SPICE-accurate simulation of a large-scale circuit with strong parasitic coupling is to develop innovative simulation approaches to decrease the number of LU factorizations required for the entire time-domain simulation [17][18]. Recently [9] proposes to perform time-domain simulation by using a single fixed time step and the successive chord (SC) method [6] for linearlizing nonlinear devices. Since the MNA circuit matrix for a fixed time step and a fixed chord will not change during transient simulation, only one LU factorization is required. Coupled with model order reduction and table lookup MOSFET models, this idea has been demonstrated to be effective for the simulation of single-stage digital logic gates driving large-size parasitic networks [9]. Unfortunately, there are two principal difficulties that restrict the use of this *linear-centric* idea successfully to the simulation of general VLSI circuits: *1*) Most VLSI circuits have widely distributed time constants, and require dynamic time step control for the simulation efficiency and accuracy. With varying time steps, the circuit matrix is no longer constant for every time point. *2*) The SC method has the linear convergence rate. It often needs excessive amount of iterations to converge, and thus requires a huge number of forward/backward substitutions (FBSs) [6].

This paper presents *SILCA* — ***S****emi-****I****mplicit* ***L****inear* ***C****entric* ***A****nalysis* — a new method capable of analyzing VLSI circuits containing strong parasitic couplings *with SPICE-like accuracy yet orders of magnitude speedup*. *SILCA* consists of two new ideas that can help keep the MNA matrix as constant as possible during transient simulation even with varying time steps:

1) Semi-implicit iterative integration scheme to keep equivalent conductance of capacitor/inductor companion models constant for a relatively large time interval;
2) Successive variable chord (SVC) method to keep linearized conductance of nonlinear devices constant for a relatively large voltage/current range. Rank-one update technique is further applied for fast LU factorization.

With these, the required LU factorizations can be reduced by orders of magnitude with a small increase of iterations. Further, the entire method is stable, accurate, and has been implemented to SPICE3.

This paper is organized as follows. Section 2 proposes the new semi-implicit iterative integration scheme. The SVC method and rank-one update technique are presented in Section 3. Section 4 describes the *SILCA* algorithm. Experimental results on substrate

[*] This research was supported by DARPA NeoCAD Program under Grant No. N66001-01-8920 and NSF CAREER Award under Grant No. 9985507.

and power/ground coupling analysis are shown in Section 5. Finally, Section 6 concludes this paper.

2. Semi-implicit iterative integration scheme

To implement the linear-centric idea for time-domain simulation, we propose a new semi-implicit iterative integration scheme. First, semi-implicit integration formulae are used as a predictor to provide a good initial guess for the present time point. Second, iterative integration formulae are applied as a corrector to achieve the final accurate solution and to ensure numerical stability at the same time. Both of the new integration formulae will keep the equivalent conductance of capacitor/inductor companion models constant for a relatively larger time interval.

2.1 Semi-implicit integration predictor

In [1], semi-implicit integration scheme has been suggested for strongly coupled interconnect systems. In this sub-section, we extend this idea and introduce a generalized semi-implicit integration predictor for dynamic step transient simulation.

Let h be a *basis* time step size. The time step size h_n for the present time point t_n can be represented by $h_n = \alpha h$, where α is a positive scalar. Now let us rearrange the standard trapezoid (TR) formula as follows:

$$\dot{x}_n = \frac{2}{h_n}(x_n - x_{n-1}) - \dot{x}_{n-1} = \frac{2}{\alpha h}(x_n - x_{n-1}) - \dot{x}_{n-1}$$
$$= \frac{2}{h}x_n - \frac{2}{h}x_{n-1} - \frac{2(\alpha-1)}{\alpha h}(x_n - x_{n-1}) - \dot{x}_{n-1} \qquad (1)$$

where $x_n = x(t_n)$, $x_{n-1} = x(t_{n-1})$, $t_n = t_{n-1} + \alpha h$, $\dot{x}_n$ and $\dot{x}_{n-1}$ are first-order time derivatives at t_n and t_{n-1}, respectively. Noting that the first term leads to the constant equivalent conductance, *we would like to represent the x_n in the third term by all the known values from the previous time points.* This can be done using any explicit Adams-Bashforth formula [6]. The simplest is the forward Euler (FE) formula with step size αh as follows:

$$x_n = x_{n-1} + \alpha h \dot{x}_{n-1} \qquad (2)$$

Then the following ***constant-conductance semi-implicit trapezoid integration formula*** for step size αh is derived:

$$\dot{x}_n = \frac{2}{h}x_n - \frac{2}{h}x_{n-1} - (2\alpha - 1)\dot{x}_{n-1} \qquad (3)$$

When $\alpha = 1$, the above formula reduces to the standard TR formula. When $\alpha = 1/2$, it represents the backward Euler (BE) formula with step size $h/2$. We can formally prove the following theorem:

Theorem 1: *The local truncation error ε of the constant-conductance semi-implicit trapezoid formula with time step size αh is given by ($x_\xi = x(t_\xi)$, t_ξ is between t_{n-1} and t_n)*

$$\varepsilon = \left(1 - \frac{1}{\alpha}\right)\left(\frac{\ddot{x}_\xi}{2}\right)(\alpha h)^2 + \left(1 - \frac{1.5}{\alpha}\right)\left(\frac{\dddot{x}_\xi}{6}\right)(\alpha h)^3$$

The proof is a straightforward application of the local truncation error (LTE) estimation for the standard TR formula [6].

The stability property of the semi-implicit TR formula can be proved as below:

Theorem 2: *The absolute stability region of the constant-conductance semi-implicit trapezoidal formula with time step size αh is defined by*

$$\left|\frac{1 + (2\alpha - 1)z}{1 - z}\right| < 1$$

where $z = -h/(2\tau)$ and τ is the time constant of a circuit.

From Theorem 2, several observations can be made on the stability: 1) The semi-implicit TR formula is not A-stable [6] when $\alpha > 1$ since the absolute stability region will approach that of the FE formula, so it cannot be used as a dynamic time step control scheme independently. 2) When $\alpha < 1$, the semi-implicit TR formula is A-stable. Thus *SILCA* implements the semi-implicit TR formula as a predictor when $\alpha < 1$ to provide a good initial guess for the present simulation time point. 3) The semi-implicit TR formula has the "stiff decay" [7] property when $\alpha < 1$. It means that a decent description of the solution in rapidly switching moments could be maintained in the highly stiff case, whereas the standard TR formula generally encounters numerical oscillation phenomena.

2.2 Iterative integration corrector

The LTE and stability problems of the semi-implicit integration formulae come from the approximation step with explicit integration formulae in Eq. (1). In this section, we further propose an improvement of this scheme by using iteration.

Rather than using explicit integration formulae, the x_n in the third term of Eq. (3) is replaced by the (k-1)-th iteration solution $x_n^{(k-1)}$ at the present time point and a new k-th iteration solution $x_n^{(k)}$ is achieved by solving Eq. (3), where k is the iteration number. This leads to the iterative version of Eq. (3), called the ***constant-conductance iterative trapezoid formula***, written as follows:

$$\dot{x}_n^{(k)} = \frac{2}{h}x_n^{(k)} - \frac{2}{h}x_{n-1} - \frac{2(\alpha-1)}{\alpha h}(x_n^{(k-1)} - x_{n-1}) - \dot{x}_{n-1}$$
$$= \frac{2}{h}x_n^{(k)} - \frac{2}{h}x_n^{(k-1)} + 2\frac{x_n^{(k-1)} - x_{n-1}}{\alpha h} - \dot{x}_{n-1} \qquad (4)$$

where $x_n^{(k)}$ and $x_n^{(k-1)}$ are the solution of the present time point for iteration k and k-1 respectively. If the iterative integration formulae converge successfully, the LTE requirement will be satisfied since the final converged solution is the same as that with the implicit integration formulae.

To study the convergence property, let us re-write the circuit equation as below:

$$\boldsymbol{Gx} + \boldsymbol{C}\dot{\boldsymbol{x}} = \boldsymbol{b}$$

where $\boldsymbol{G}$ and $\boldsymbol{C}$ represent the conductance and susceptance matrices, and $\boldsymbol{b}$ is the vector of input sources. Replace time derivatives by the iterative trapezoid formula Eq. (4), we have

$$\boldsymbol{G}\boldsymbol{x}_n^{(k)} + \boldsymbol{C}\left(\frac{2}{h}\boldsymbol{x}_n^{(k)} - \frac{2}{h}\boldsymbol{x}_n^{(k-1)} + 2\frac{\boldsymbol{x}_n^{(k-1)} - \boldsymbol{x}_{n-1}}{\alpha h} - \dot{\boldsymbol{x}}_{n-1}\right) = \boldsymbol{b}$$
$$\left(\boldsymbol{G} + \frac{2\boldsymbol{C}}{h}\right)\boldsymbol{x}_n^{(k)} = \left(1 - \frac{1}{\alpha}\right)\frac{2\boldsymbol{C}}{h}\boldsymbol{x}_n^{(k-1)} + \frac{2\boldsymbol{C}}{\alpha h}\boldsymbol{x}_{n-1} + \boldsymbol{C}\dot{\boldsymbol{x}}_{n-1} + \boldsymbol{b}$$

Clearly the iterative trapezoid formula converges if

$$\left\|\left(\boldsymbol{G} + \frac{2\boldsymbol{C}}{h}\right)^{-1}\left(1 - \frac{1}{\alpha}\right)\frac{2\boldsymbol{C}}{h}\right\| < 1$$

where $\|\bullet\|$ represents the spectral radius of the iteration matrix. In the worst case, to achieve convergence for a decaying system, $0.5<\alpha<\infty$ is required. In practice, to speed up the convergence and ensure accuracy, $0.625<\alpha<2.5$ is used.

The absolute stability regions of iterative integration formulae are also related to the iteration number k. We can formally prove the following Theorem 3:

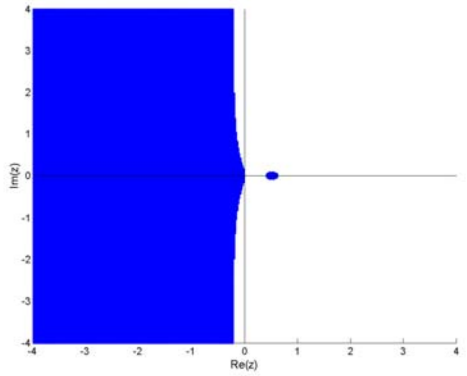


Figure 1. Absolute stability region of the iterative TR formula for α=0.625 and k=2.

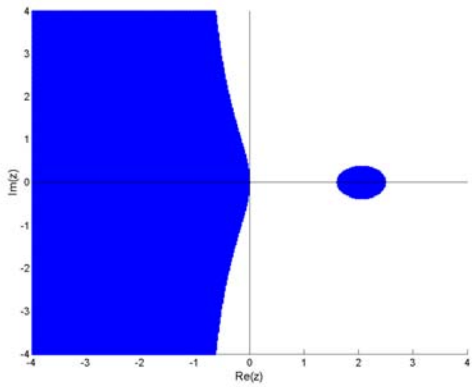


Figure 2. Absolute stability region of the iterative TR formula for α=2.5 and k=2.

Theorem 3: *The absolute stability region of the constant-conductance iterative trapezoidal formula for step size αh is defined by*

$$\left|\left(\frac{1-1/\alpha}{1-z}\right)^k\left(\frac{2z}{z-1/\alpha}\right)+\frac{1/\alpha+z}{1/\alpha-z}\right|<1$$

where $z = -h/(2\tau)$ and τ is the time constant for a circuit. The absolute stability regions for $\alpha = 0.625$ and $\alpha = 2.5$ when $k = 2$ are shown in Fig. 1 and Fig. 2, respectively, which satisfy the "stiff stability" requirements suggested by Gear [11]. Furthermore, if $0.5<\alpha<\infty$ (as required by the convergence property), the absolute stability region will finally reach that of the standard TR formula when $k\rightarrow\infty$. Therefore, the iterative integration formulae can be applied to either decaying or oscillating systems. In practice, to ensure A-stability, a lower time step size limit could be set so that the standard TR formula is applied under the condition that the present time step is less than the lower limit.

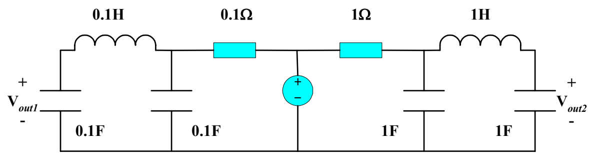


Figure 3. A linear RCL circuit example.

The efficiency of the semi-implicit iterative integration scheme can be illustrated with a simple linear circuit example shown in Fig. 3. It includes two RCL filters with time constants that differ by a factor of 100. The input is a pulse signal (initially in the low voltage level 0v) with 50% duty ratio and 80 sec period. The simulation length is set to 160 sec. Since the minimum time constant is 0.01 sec, at least 16000 time points are required for a fixed-step transient simulation. Simulation results with *SILCA* and SPICE3 are shown in Table I, where *#Total points* represents the number of total simulated time points and *#Accepted points* represents the number of actual accepted time points. It can be seen in Table I that *SILCA* and SPICE3 achieve similar *#Total points* and *#Accepted points*, which are much less than that required by a fixed step method. Furthermore, the number of LUs with *SILCA* is decreased to 1.14% of that with SPICE3 (or *87.63X LU reduction*). The number of iterations has been increased to about *2.5X*.

Table I. Simulation results for a linear RCL circuit example.

	# Total points	# Accepted points	# Iteration	# LU
SPICE3	2630	1965	5258	5258
SILCA	2636	1971	12649	60

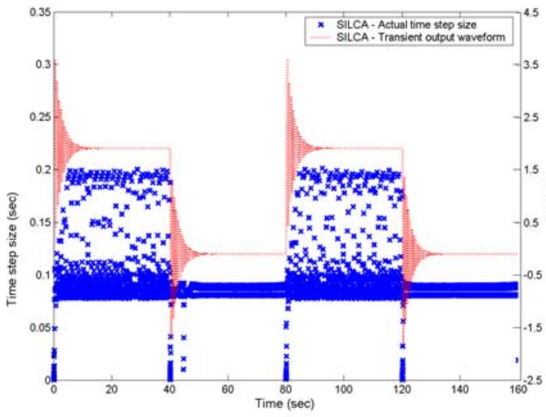


Figure 4. Distribution of actual time step sizes and time-domain output waveform of V_{out1}.

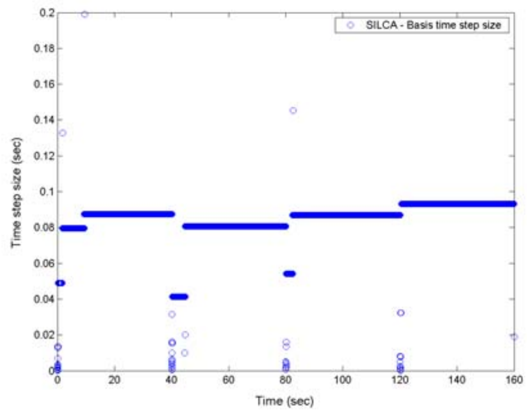


Figure 5. Distribution of basis time step sizes.

Figure 4 shows the distribution of actual simulated time step sizes (αh) and the output waveform of V_{out1}. The distribution of simulated time step sizes is much denser when V_{out1} is close to 0v. The reason is that the relative LTE for a low voltage level is small, so time step sizes are restricted by the relative LTE and cannot change too much when V_{out1} is close to 0v. It can be seen that most of simulated time step sizes are between 0.05 sec and 0.2 sec, centering around 0.08 sec. Recall that the iterative integration formulae can make a relaxation of $0.625<\alpha<2.5$, it is possible that only a few basis time step sizes are required for MNA stamping. Figure 5 shows the distribution of basis time step sizes (h) used for MNA stamping during *SILCA* simulation. It can be seen that the circuit matrix is now kept constant for a larger time interval (i.e., between 45 sec and 80 sec).

3. Successive variable chord method

In SPICE, a new LU factorization is required for each Newton-Raphson iteration. This can be extremely costly for a circuit system with strong parasitic coupling effects. The successive chord method [6] always uses a fixed chord as the first order dI/dV derivative during nonlinear iteration. Hence, at each time point, only one LU factorization is needed for nonlinear iteration. But it is generally difficult to choose a single fixed chord for a (strongly) nonlinear curve to always ensure a good convergence rate.

To achieve a good balance between the number of LUs and that of iterations, we propose to use the successive variable chord (SVC) method. The basic idea is to split a nonlinear curve into different segments, each of which represents a weakly nonlinear curve and the same (local) chord is used for the same segment during nonlinear iteration – so-called *Piecewise Weakly Nonlinear (PWNL) analysis*. As shown in Fig. 6, the nonlinear curve is divided into three PWNL segments with three local chords defined respectively. A new LU factorization is performed only if the nonlinear curve enters a different PWNL segment with the local chord varied.

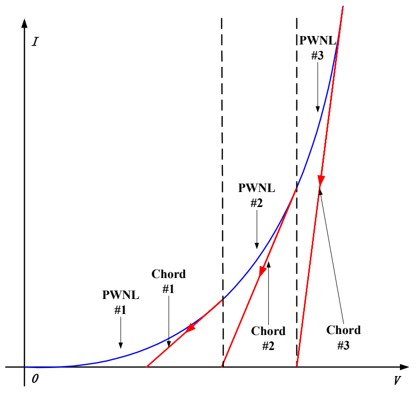


Figure 6. A PWNL example implemented with the SVC method.

The PWNL idea implemented with the SVC method is in practice very effective due to the following facts: *1*) MOSFETs in analog applications generally operate linearly around their operating points, only weakly nonlinearity properties may be present. A fixed chord representing the g_m, $g_{mbs,}$ and g_{ds} of MOSFETs at operating points is generally good enough. *2*) MOSFETs in digital applications reside in two regions most of the time – cutoff region and well-conducted linear region with a very small source-to-drain voltage, both regions have a relatively steady g_m, $g_{mbs,}$ and g_{ds}. The only situation where g_m, $g_{mbs,}$ and g_{ds} change a lot is the time when MOSFETs switch from the cut-off region through the saturation region to the linear region (or vice versa), which only occupies a small fraction of total simulation time for a MOSFET in a large digital system. So, a fixed chord for these situations will not significantly affect the total iteration process.

In our implementation, five MOSFET operating regions for digital applications are defined as shown in Fig. 7, and g_m, g_{mbs}, and g_{ds} for different operating regions are listed in Table II. In Table II, *Reg#0* represents the cut-off region, *Reg#1* and *Reg#3* are saturation regions, and *Reg#2* and *Reg#4* are linear regions. $g_{m\text{-}max}$ and $g_{mbs\text{-}max}$ are maximum values in all the regions (defined by Vdd), and $g_{ds\text{-}i}$ are defined for different regions to ensure nonlinear convergence. It should be noticed here that g_m and g_{mbs} are both zero for *Reg#1* and *Reg#2* since the effect of g_{ds} is dominant for these two regions. Furthermore, this definition can avoid frequent MOSFET switching between normal and reversed modes due to numerical errors, and thus further reduces the number of required LU factorizations.

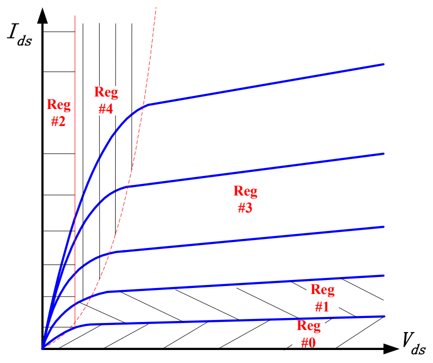


Figure 7. Operating regions of MOSFET for digital applications.

Table II. g_m, g_{mbs}, and g_{ds} for different MOSFET operating regions.

	Reg#0	Reg#1	Reg#2	Reg#3	Reg#4
g_m	0	0	0	$g_{m\text{-}max}$	$g_{m\text{-}max}$
g_{mbs}	0	0	0	$g_{mbs\text{-}max}$	$g_{mbs\text{-}max}$
g_{ds}	0	$g_{ds\text{-}1}$	$g_{ds\text{-}2}$	$g_{ds\text{-}3}$	$g_{ds\text{-}4}$

By the above MOSFET operating region definition, only five sets of g_m, g_{mbs}, and g_{ds} are used during transient simulation for digital systems. We further have the following observations: *1*) At one time point, most MOSFETs in a large digital system will stay in their operating regions as defined above, while only a few may switch from one region to another region. *2*) For a switching MOSFET, the update of g_m, g_{mbs}, and g_{ds} is region-wise. In other words, the change of g_m, g_{mbs}, and g_{ds} from *Reg#i* to *Reg#j* is fixed. Therefore, in the case that a small amount of MOSFETs change operating regions, we could update the L and U matrices directly with the rank-one update technique [12][13], rather than updating the MNA matrix and performing costly LU factorization again.

Suppose that the present MNA matrix is Y, and one MOSFET is now switching from *Reg#1* to *Reg#2*. The MNA matrix for the next iteration can be expressed by:

$$Y' = Y + cr^T$$

where c and r are sparse column vectors representing values of updated elements. In this case, $c = r = [0\ldots0\ e\ 0\ldots0\ {-e}\ 0\ldots]^T$, and $e = \sqrt{(g_{ds-2} - g_{ds-1})}$. The L and U matrices for Y' can be updated from the previous ones for Y efficiently with the rank-one update technique, whose worst case cost is $O(m{*}n^2)$ (m is the number of updated elements, n is the circuit matrix size) and will be much less with sparse matrix solvers. Typically, m is much less than n, so rank-one update can provide a much faster LU factorization.

To illustrate the efficiency of the SVC method and the rank-one update technique, simulations on several digital and RF circuits have been performed and results are shown in Table III. It can be seen that the number of iterations is generally increased to *1.5~2.5X* of that with SPICE. But the number of LUs with the SVC method is decreased to 10%~55% of that with SPICE. After the rank-one update is applied, the number of regular LUs is further decreased to 3%~20% of that with SPICE. It should be noticed that more LU speed-up with rank-one update is achieved for relatively large systems, such as a 20-stage inverter chain, a ring oscillator and a VCO. Very little benefit can be achieved on simple systems, such as a single inverter, a single NAND2 gate, etc. Rank-one update technique will be very efficient for a nonlinear system with strong parasitic coupling effects, since only the L and U matrices for the sparse nonlinear part need to be updated during nonlinear iteration, and the dense linear part remains unchanged.

Table III. Simulation results on test circuits.

Test Circuits	#Total points	#Accept points	#Iter	# LU	
				w/o rnk1	w rnk1
Inv	142	127	351	351	-
	145	129	545	73	65
20-stage inverter chain	369	266	1201	1201	-
	358	260	2401	493	66
Nand2	132	123	313	313	-
	123	114	541	73	60
One-shot trigger	501	421	1542	1542	-
	486	421	3595	438	213
Comparator	145	127	455	455	-
	148	130	1131	130	66
Ring Oscillator	243	173	1031	1031	-
	257	178	2420	571	30
VCO	1506	1045	7630	7630	-
	1468	1042	16146	739	221

Note:** ***For each circuit, the 1st row is the SPICE3 result, the 2nd row is the SILCA result

4. The *SILCA* algorithm

Table IV. Transient simulation flow in *SILCA*.

```
DC operating point analysis
Choose an initial step size $h_0$, the basis step size $h = h_0, t = 0$
WHILE ($t<T_{\text{final}}$){
    OUTER LOOP: do{
        $\alpha = h_n/h$, iter_no = 0
        INNER LOOP: do{
            IF($0.625<\alpha<2.5$){
                IF($\alpha<1$ && iter_no==0) {
                    Apply Semi-Implicit Integration Predictor Eq. (3)
                }
                Apply Iterative Integration Corrector Eq. (4)
            }ELSE{
                IF(iter_no==0) { $h = h_n$ }
                Apply Standard Implicit Integration Scheme
            }
            Apply the SVC method on nonlinear devices
            IF ($0.625<\alpha<2.5$) {
                IF (chord is changed) { Apply Rank-one update & FBS }
                ELSE { Apply FBS}
            }ELSE{ Apply LU factorization & FBS }
            iter_no = iter_no + 1
        } while (not converged)
        Choose a new $h_n$ based on LTE requirement
    } while (LTE greater than predefined error limit)
    $t = t + h_n$
}
```

The basic flow for *SILCA* transient simulation is shown in Table IV. Practical considerations, such as breakpoints [5], are not included in this flow for clarity. In this flow, a new LU factorization is only required when standard implicit integration scheme is used. In case that only local chords of nonlinear devices change, rank-one update is performed for fast LU factorization. No LU factorization is needed in any other case.

5. Experimental results

5.1 Substrate coupling example

The first example is a simple substrate coupling network as shown in Fig. 8. It includes two inverters with pulse inputs in different operating frequencies – the first inverter operates at a low frequency and the second inverter operates at a high frequency. The bulk contacts of nMOSFETs are directly connected to P-substrate ports, and those of pMOSFETs are connected to P-substrate ports through a capacitor between the N-well and the P-substrate [2]. There are four other P-substrate ports connecting to the ground and the backplane of the substrate is also connected to the ground. RCL loads are added at the output of each inverter (not shown in Fig. 8. The substrate is modeled as a dense resistor network [14] that is formed by a 3-dimensional dense resistor mesh with multiple layers; In Fig. 8, a one-layer resistor network is illustrated to model the substrate among four inverter bulk contacts.

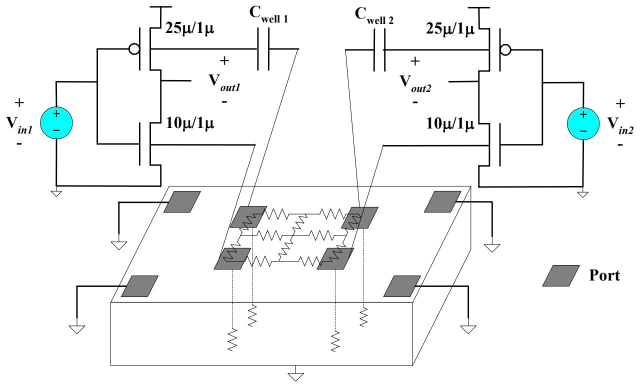


Figure 8. The substrate coupling example.

Although simplified truncated substrate models have been proposed to capture dominant coupling conductance [2][14], they are likely to underestimate coupling effects in circuit systems designed to be noise mature [1]. Furthermore, the accuracy with simplified substrate models may not be sufficient. Therefore, accurate analysis of a circuit with a fully modeled substrate is desirable for high fidelity circuit design and verification.

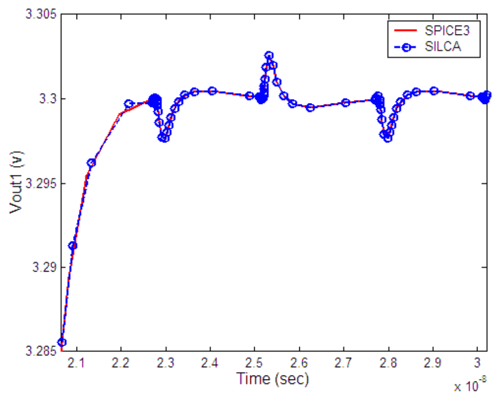


Figure 9. Transient output waveform of the first inverter for the substrate coupling example.

Figure 9 shows the transient output waveform of the first inverter when the output signal is digital "1" (the high voltage level). First, the result from *SILCA* matches that from SPICE3. Second, it can be seen that high frequency feed-through signals from the second inverter are present in Fig. 9. This is an important first-pass design failure reason in deep-submicron digital and analog circuit designs, which may often not be captured by simplified substrate analysis [2][3].

Table V is the statistics of running *SILCA* on a number of substrate analysis examples with varying circuit substrate network complexity. In our experiments, the *number of layers* and the *number of resistors per layer* are changed to vary the total number of circuit elements. A maximum *40.25X* LU speed-up and *17.32X* overall speed-up (with about 35 thousand elements) are achieved for this simple substrate coupling analysis example, and the FBS cost is increased to *2.5~3X*.

Several observations are: *1*) The larger the LU/FBS cost ratio are, the more overall speed-up can be achieved with *SILCA*. Therefore, *SILCA* is very suitable for deep-submicron circuit systems with strong parasitic coupling effects; *2*) Device load cost with *SILCA* is decreased, which is proportional to the LU speed-up, since device load of resistors are only required when a new LU is performed. *3*) The maximum overall speed-up will reach the LU speed-up (around *30X* for this example) for large strongly coupled systems. Figure 10 shows the run time comparison between *SILCA* and SPICE3 with the number of total circuit elements varied. No rank-one update technique is used for the substrate coupling example.

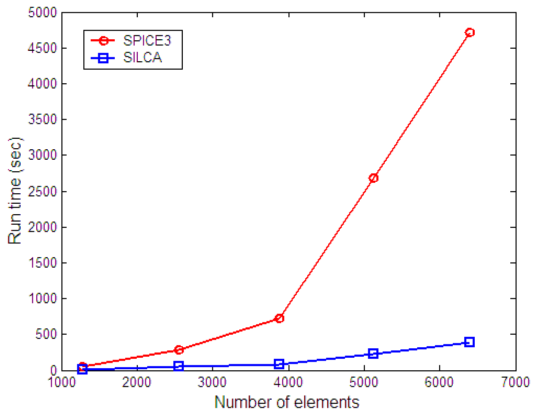


Figure 10. Run time comparison of the substrate coupling example.

Table V. Simulation results for the substrate coupling analysis example.

#Layer x #Res_Per_Layer	#Elements	SPICE3				SILCA			Speed-up	
		LU (sec)	FBS (sec)	Load(sec)	LU/FBS	LU(sec)	FBS(sec)	Load(sec)	LU	Overall
1x1281	1281	32.96	3.99	8.68	8.26	0.96	11.87	1.18	34.33	3.26
2x1281	2562	249.29	14.24	22.56	17.51	9.73	39.80	1.99	25.62	5.55
3x1281	3886	663.86	23.19	35.78	28.63	16.49	63.32	2.40	40.25	8.79
4x1281	5124	2.533e3	59.26	91.77	42.75	78.91	147.27	4.66	32.10	11.63
5x1281	6405	4.496e3	88.37	123.87	50.87	131.22	249.40	7.79	34.26	12.12
6x4961	29766	2.455e5	2.947e3	2.927e3	83.32	8.297e3	7.722e3	112.37	29.59	15.58
7x4961	34727	5.348e5	5.629e3	5.318e3	95.00	1.770e4	1.361e4	195.06	30.21	17.32

5.2 Power/ground analysis example

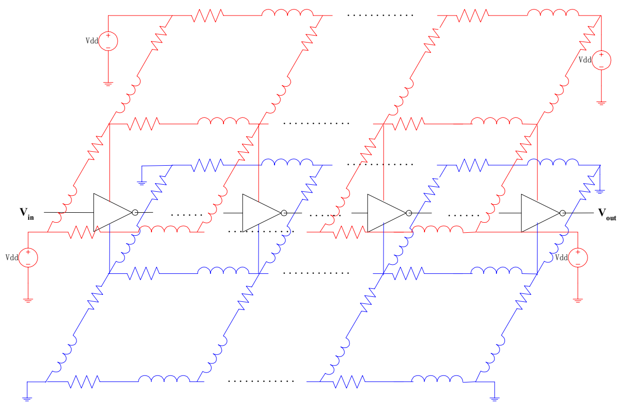


Figure 11. The power/ground analysis example.

The second example is a power/ground network as shown in Fig. 11. The power and ground supply networks are modeled as two RCL mesh layers (parasitic coupling capacitors are not shown in Fig. 11). For fast transient simulation of a nonlinear circuit with power/ground networks, nonlinear devices are generally simplified as (piecewise linear) current sources plus device parasitic capacitors [4][15], to model real nonlinear device behaviors. However, this is generally hard and not accurate for a large circuit. Accurate simulation of a full-scale power/ground network is highly desirable for accurate circuit verification and power/ground optimization. In our example, between these two layers is a 20-stage inverter chain, different inverters of which are connected to different power/ground nodes. Furthermore, RCL loads are added for each inverter to model interconnect lines between adjacent stages.

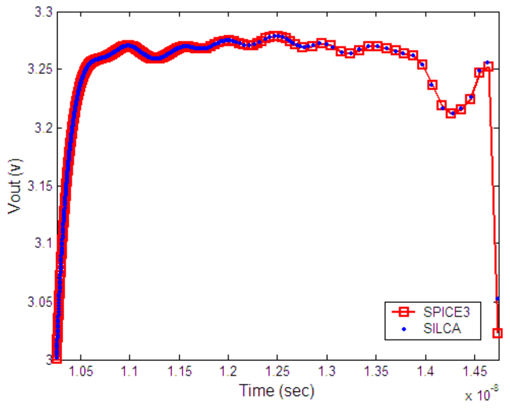


Figure 12. Transient output waveform of the inverter chain for power/ground analysis example.

Figure 12 shows the transient output waveform of the inverter chain when the output signal is digital “1” (the high voltage level).

The "1" signal has been disturbed due to the *IR*-drop (the input Vdd is 3.3v) and *L*dI/dt* effects of the power/ground network. Table VI shows the simulation results with varied numbers of elements modeling the power/ground network. In our experiments, the size of two RCL meshes is changed to vary the number of elements. We can see that *SILCA* achieves more speed-up for larger circuits. It is worthy to notice that, the maximum LU speed-up and overall speed-up reach *87.70X* and *18.97X* (with about 60 thousand elements) respectively with the rank-one update technique, which are *24.88X* and *11.86X*, respectively, with only the SVC method.

6. Conclusions

In this paper, a new time-domain nonlinear circuit simulation method called *SILCA* has been proposed for deep-submicron VLSI circuit design and verification, where requires accurate modeling of parasitic couplings or coupled circuit and electromagnetic modeling. A new dynamic time-step semi-implicit iterative numerical integration scheme was developed to keep constant equivalent conductance for capacitor/inductor companion models. We also proved the convergence and stability property of the new introduced integration formulae. A successive variable chord method was further proposed as an alternative of the Newton-Raphson method and the rank-one update technique has been implemented for fast LU factorization. With these techniques, *SILCA* can reduce the number of costly LU factorization dramatically in transient simulation. Experimental results on substrate and power/ground networks have demonstrated that *SILCA* yields SPICE-like accuracy with orders of magnitude speed-up over SPICE3.

7. References

[1] J. R. Phillips and L. M. Silveira, "Simulation Approaches for Strongly Coupled Interconnect Systems", *Proc. IEEE/ACM Int. Conf. on Computer-Aided Design*, pp. 430-437, November 2001.

[2] A. Samavedam, A. Sadate, K. Mayaram, and T. S. Fiez, "A Scalable Substrate Noise Coupling Model for Design of Mixed-Signal IC's", *IEEE Journal of Solid-State Circuits*, vol. 35, no. 6, pp. 895-904, June 2000.

[3] M. Nagata, J. Nagai, K. Hijikata, T. Morie, and A. Iwata, "Physical Design Guides for Substrate Noise Reduction in CMOS Digital Circuits", *IEEE Journal of Solid-State Circuits*, vol. 36, No. 3, pp. 539-549, March 2001.

[4] H. Su, K. H. Gala and S. S. Sapatnekar, "Fast Analysis and Optimization of Power/Ground networks", *Proc. IEEE/ACM Int. Conf. on Computer-Aided Design*, pp. 477-480, November 2000.

[5] L. W. Nagel, *SPICE: A Computer Program to Simulate Semiconductor Circuits*, University of California, Berkeley, Tech. Rep., UCB/ERL M520, May 1975.

[6] W. J. McCalla, *Fundamentals of Computer-Aided Circuit Simulation*, Kluwer Academic Publishers, 1988.

[7] U. M. Ascher and L. R. Petzold, *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*, SIAM, 1998.

[8] K. S. Kundert and A. Sangiovanni-Vincentelli, *Sparse User's Guide – A Sparse Linear Equation Solver Version 1.3a*, University of California, Berkeley, April 1988.

[9] E. Acar, F. Dartu, and L. T. Pileggi, "TETA: Transistor-Level Waveform Evaluation for Timing Analysis", *IEEE Trans. on Computer-Aided Design*, vol. 21, no. 5, pp. 605-616, May 2002.

[10] A. Odabasioglu, M. Celik, and L. T. Pillegi, "PRIMA: Passive Reduced-Order Interconnect Macromodeling Algorithm", *IEEE Trans. on Computer-Aided Design*, vol. 17, no. 8, pp. 645-654, August 1998.

[11] C. W. Gear, *Numerical Initial Values Problems in Ordinary Differential Equations*, Prentice-Hall, 1971.

[12] T. Fujisawa, E. S. Kuh, and T. Ohtsuki, "A Sparse Matrix Method for Analysis of Piecewise-Linear Resistive Networks", *IEEE Trans. on Circuit Theory*, vol. CT-19, no. 6, pp. 571-584, November 1972.

[13] J. T. J. van Eijndhoven and M. T. van Stiphout, "Latency Exploitation in Circuit Simulation by Sparse Matrix Techniques", *Proc. IEEE int. Symp. on Circuits and Systems*, pp. 623-626, June 1988.

[14] N. K. Verghese, T. J. Schmerbeck, and D. J. Allstot, *Simulation Techniques and Solutions for Mixed-Signal Coupling in Integrated Circuits*, Kluwer Academic Publishers, 1995.

[15] T. Chen and C. C.-P. Chen, "Efficient Large-Scale Power Grid Analysis based on Preconditioned Krylov-subspace Iterative Methods", *Proc. IEEE/ACM Design Automation Conference*, pp. 559-562, June 2001.

[16] Y. Wang, V. Jandhyala, and C.-J. R. Shi, "Coupled Electromagnetic-Circuit Simulation of Arbitrarily-Shaped Conducting Structures", *Proc. IEEE Conf. On Electrical Performance of Electronic Packaging*, pp. 233-236, October 2001.

[17] L. R. Petzold, "A Description of DASSL: A Differential/Algebraic System Solver", *IMACS Trans. Scientific Computing*, R. Stepleman et al. (eds.), vol. 1, pp. 65-68, 1983.

[18] P. F. Cox, R. G. Burch, P. Yang, and D. E. Hocevar, "New Implicit Integration Method for Efficient Latency Exploration in Circuit Simulation", *IEEE Trans. on Computer-Aided Design*, vol. 8, no. 10, pp. 1051-1064, October 1989.

Table VI. Simulation results for the power/ground analysis example.

#Elements	SPICE3 (sec)		SILCA (sec)				Speed-up			
			w/o rank-1		with rank-1		w/o rank-1		with rank-1	
	LU	Overall	LU	Overall	LU	Overall	LU	Overall	LU	Overall
4002	1.532e3	1.666e3	83.19	344.21	19.58	287.42	18.42	4.84	78.24	5.80
34802	5.225e4	5.418e4	2.100e3	5.670e3	685.82	4.236e3	24.88	9.56	76.19	12.79
61602	1.975e5	2.032e5	8.656e3	1.713e4	2.252e3	1.071e4	22.82	11.86	87.70	18.97

Symbolic Analysis of Analog Circuits with Hard Nonlinearity*

Alicia Manthe, Zhao Li, and C.-J. Richard Shi
Department of Electrical Engineering, University of Washington, Seattle, WA 98195
{amanthe, lz2000, cjshi}@ee.washington.edu

ABSTRACT

A new methodology is presented to solve a strongly nonlinear circuit, characterized by Piece-Wise Linear (PWL) functions, symbolically and explicitly in terms of its circuit parameters and is amenable to computer implementation. The method is based on a modified nodal formulation of piecewise linear circuit equations as a *mixed* Linear Complementarity Problem (MLCP). The technique of determinant-decision diagrams is applied to implement the symbolic transformation of the MLCP to the standard LCP. Complementarity-decision diagrams are used to represent the resulting LCP. Examples are presented that demonstrate the accuracy and efficiency of the proposed method.

Categories and Subject Descriptors
T5.3 Analog and mixed-signal design tools and RF

General Terms Algorithms

Keywords
Symbolic Analysis, Circuit Nonlinearity, PWL

1. INTRODUCTION

Analysis of the effect of device nonlinearity on the system performance is critical to high-performance analog/RF systems-on-chip design [5][8]. While a class of nonlinear circuits, known as *weakly nonlinear,* can be analyzed via linearized techniques such as small-signal analysis or techniques based on linearized analysis such as harmonic balance or Volterra series [10], many circuits ranging from switches, mixers, saturation-limited amplifiers to switched-capacitor filters and switching power converters, exhibit strong nonlinearities. Circuits exhibiting strong nonlinearities refer to sudden changes of device behavior, for example, switching of operating regions, sudden changes of device physics, and piecewise I-V characteristics.

Strong nonlinearities also arise in the following two scenarios. First, there is increasing interest in using digital logic signals to control the operations of analog/RF front-ends. As a consequence, more "novel" analog signal processing circuits may change their behaviors abruptly. Second, with the analog hardware description languages such as VHDL-AMS and Verilog-AMS gaining more momentum [5], behavioral models are being developed for systems-on-chip simulation and architecture evaluation. Many behavioral models are characterized as piecewise linear models consisting of sudden behavior changes.

Analysis of circuits demonstrating strong nonlinearities is known to be challenging [8]. Time-varying Volterra series [12], sliding kernels dynamic Volterra series [3], and describing functions [4] have been proposed to handle a certain class of circuits such as mixers, the methods depend highly on the specific circuit structure, require derivatives, and are hard to automate. Further, the complexity increases dramatically when high order series are required. The multi-rate partial differential equation (MPDE) formulation [10] can compute numerically the multi-rate behavior efficiently with strong known linearity such as output spikes. However, the method also requires the computation of a Jacobian matrix, which prohibits its use towards hard nonlinearity analysis.

This paper presents a new method capable of analyzing explicitly and exactly the behavior of circuits with strong nonlinearities characterized by piecewise linear functions. Our work is inspired by the recent work of Bokhoven and Leenaerts [7], which demonstrates that explicit formulae can be derived for a class of PWL circuits that can be formulated as so-called P-class linear complementarity problem (LCP). Our novel contributions are as follows: (1) To be amenable to computer implementation, we first present a formulation of PWL circuits equations using the framework of Modified Nodal Analysis (MNA). This leads to a mathematical problem known as the Mixed Linear Complementarity Problem (MLCP) [2]. (2) We exploit a compact data structure known as determinant decision diagrams (DDDs) [11] to represent all the manipulations from MLCP to LCP symbolically and utilize complementarity decision diagrams (CDDs) [8] to characterize the LCP expressions.

The method is amenable to computer implementation. Furthermore, it represents all the solutions (voltages and currents) explicitly in terms of circuit parameters, input sources based on a special mathematical operator = $\lfloor . \rfloor$. The symbolic expressions can help designers to gain insight on how circuit parameters affect the circuit linearity. A very efficient numerical time-domain and harmonic simulator have been implemented based on the repetitive evaluation of the resulting expressions. The simulator can calculate the harmonics and time-domain responses exactly, while the SPICE-like numerical simulators have to invoke various smoothing functions to compute the approximate solutions. As observed in our experiments, how the PWL is smoothed can lead

[1] This research was supported in part by U.S. Defense Advanced Research Project Agencies (DARPA) under Grant N66001-01-1-8919, in part by an NSF Career Award, and in part by Semiconductor Research Corporation under Contract 2000-NJ-827.

to significant changes in the operating point, linear and nonlinear circuit characteristics.

This paper is organized as follows. Section 2 presents preliminary PWL information, which is followed by an MNA formulation of PWL circuits as the mixed linear complementarity problem (LCP). Experimental results are described in Section 4. Section 5 concludes this paper.

2. PRELIMINARY

Piece-Wise Linear (PWL) functions are used to model devices that exhibit strong nonlinearities. Numerous research by Chua [1] and furthered by van Bokhoven and Leenaerts [7] derived functions to represent networks consisting of nonlinear devices in an *explicit* form. For an *explicit* model the output vector can be obtained simply by substituting the input vector into the description. Therefore, these functions can be solved in a fraction of time needed by other models, such as, table look-up or spline function approximation.

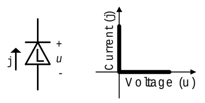


Figure 1. An *orthoator* and its *I-V* curve.

To be able to represent each piece of the PWL function in a behavioral model van Bokhoven and Leenaerts in [7] makes use of an ideal diode. To be amenable to Modified Nodal Analysis (MNA), we will call this "new" basic two-terminal circuit element an *orthoator,* as illustrated in Figure 1. An orthoator describes the behavior of a circuit with "extremely hard" nonlinearities, and it is defined in terms of the current *j* through the orthoator and the voltage *u* across the orthoator as

$$u \geq 0, \quad j \geq 0, \quad u^T j = 0. \tag{1}$$

The relationship between *u* and *j* is defined as the linear complementarity problem (LCP) [7].

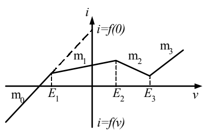


Figure 2: A PWL curve example.

Now consider the one-dimensional continuous PWL function $i = f(v)$ shown in Figure 2. It can be represented by the so-called state-model shown below [7]:

$$\begin{aligned} i &= m_0 v + \begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix} \mathbf{u} + f(0) \\ \mathbf{j} &= \begin{bmatrix} -2 \\ -2 \\ -2 \end{bmatrix} v + \mathbf{I}\mathbf{u} + \begin{bmatrix} g_1 \\ g_2 \\ g_3 \end{bmatrix} \\ &\mathbf{u} \geq 0, \mathbf{j} \geq 0, \mathbf{u}^T \mathbf{j} = 0 \\ b_k &= \frac{m_k - m_{k-1}}{2} \text{ and } g_k = 2 \cdot E_k \text{ for } k = 1 \ldots 3 \end{aligned} \tag{2}$$

The standard LCP resulting from circuit formulation can be re-written from (2) as follows:

$$\mathbf{u} = \mathbf{D}\mathbf{j} + \mathbf{q} \quad \mathbf{j}^T\mathbf{u} = 0 \quad \mathbf{j}, \mathbf{u} \geq 0 \tag{3}$$

where **u** (voltage across *orthoator*), **j** (current through *orthoator*) and **q** (input sources) are column vectors of size *m* x 1 and **D** (linear components of the circuit) is a *m* x *m* square matrix.

It has been shown that there exists a unique solution to (3) if and only if **D** is of class P, i.e., all the principle minors of the matrix are positive [2]. Then explicit solutions of **j** and **u** can be obtained explicitly using an operator called the modulus transform, which is stated here, as [7]:

$$y = \lfloor x \rfloor \rightarrow \begin{cases} y = x, & x \geq 0 \\ y = 0, & x < 0 \end{cases} \tag{4}$$

and is equivalent to the mapping $u, j \rightarrow z$ which satisfies:

$$|z| = \frac{(u+j)}{2} \text{ and } z = \frac{(u-j)}{2} \tag{5}$$

Consider the 1-dimensional (1-D) case ($m = 1$). The solution is $u = \lfloor q \rfloor$, $j = 0$ or $j = \lfloor -q/D \rfloor$, $u = 0$. This result is clearly seen by plugging in a zero for *u* to find *j* and vice versa.

The solution to the case $m = 2$ can be broken down to solve the problem of $m = 1$. Given the 2-D LCP:

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} \begin{bmatrix} j_1 \\ j_2 \end{bmatrix} + \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \tag{6}$$

Assume $j_1 = 0$, then the following is true $u_1 = \lfloor D_{12} * \hat{j}_2 + q_1 \rfloor$ and $\hat{u}_2 = D_{22} * \hat{j}_2 + q_2$. The formulation of $\hat{u}_2$ is equivalent to solving a 1-D case. Assume $\hat{u}_2 = 0$, then $\hat{j}_2 = \lfloor -q_2/D_{22} \rfloor$. Substitute $\hat{j}_2$ into u_1 yields the u_1 expression found in (7). To find j_1 then u_1 must be zero leading to: $0 = D_{11} * j_1 + D_{12} * j_2 + q_1$. Evaluating the function in terms of j_1 leads to the equation found in (7). The solutions for u_2 and j_2 are found the same way and are shown below.

$$\begin{aligned} u_1 &= \left\lfloor D_{12} * \left\lfloor \frac{-q_2}{D_{22}} \right\rfloor + q_1 \right\rfloor & j_1 &= \left\lfloor \frac{-D_{12}}{D_{11}} * \left\lfloor \frac{-q_2}{D_{22}} \right\rfloor - \frac{q_1}{D_{11}} \right\rfloor \\ u_2 &= \left\lfloor D_{21} * \left\lfloor \frac{-q_1}{D_{11}} \right\rfloor + q_2 \right\rfloor & j_2 &= \left\lfloor \frac{-D_{21}}{D_{22}} * \left\lfloor \frac{-q_1}{D_{11}} \right\rfloor - \frac{q_2}{D_{22}} \right\rfloor \end{aligned} \tag{7}$$

In general, an *n*-dimensional (*n*-D) case can be found in the same way by breaking the problem down into smaller matrices. The *n*-D case leads to *n* levels of the modulus transform. Clearly this procedure takes an exponential amount of computation and space.

In [8] a new graph-based method was introduced called the complementarity decision diagram (CDD) to reduce the computation and space by sharing these *n* levels of sub-expressions. For relatively large circuits, this technique can be orders of magnitude more efficient than the original method.

3. MNA FORMULATION OF PWL CIRCUITS AS THE MIXED LINEAR COMPLEMENTARITY PROBLEM

To facilitate the MNA formulation, we can represent the device described by the equations in (2) as a network of linear resistors, ideal voltage sources, and orthoators as shown in Figure 3.

The first piece with slope m_0 in Figure 2, is represented in Figure 3 by a resistor of value $1/m_0$ and a voltage source whose value is in terms of the slope and the extrapolation of that piece to the current axis (*f(0)*). This is the starting piece for PWL modeling. Each branch in this circuit represents a slope update on the previous piece in the PWL curve, which means that a new piece is reached once a new branch is turned on. Using this technique, any PWL circuit can be represented by a circuit consisting of a set of linear elements, (controlled) sources and orthoators.

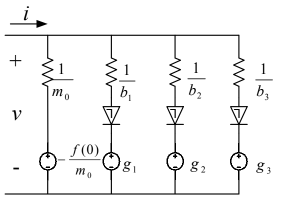
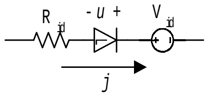


Figure 3. Network representing the PWL curve in Figure 2.

The MNA method then can be applied to solve PWL problems with an appropriate stamping rule for the orthoators. During the MNA stamping, orthoators are treated as special voltage sources. The voltage across an orthoator is *u* while its current is *j*. So, in MNA stamping, *u* goes to the right-hand side (RHS) of the MNA formulation while *j* is treated as an extra current variable. Noting that an orthoator is generally connected in series with a linear resistor and a voltage source (directly coming from the PWL mapping for circuit representation to realize the slope update), we can further treat them together as a macro circuit element. The compact MNA stamping for such a macro circuit element is as in Figure 4:



$$\left[\begin{array}{cc|c} & & 1 \\ & & -1 \\ \hline -1 & 1 & R_{id} \end{array}\right]\begin{bmatrix} V_{n1} \\ V_{n2} \\ j \end{bmatrix} = \begin{bmatrix} \\ \\ u - V_{id} \end{bmatrix}$$

Figure 4. MNA stamping rule for the orthoator.

In general, the MNA formulation of PWL circuit equations can be written in the following *mixed linear complementarity problem* (MLCP) matrix:

$$\begin{bmatrix} \mathbf{M} & \mathbf{A} \\ -\mathbf{A}^T & \mathbf{N} \end{bmatrix}\begin{bmatrix} \mathbf{x} \\ \mathbf{j} \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ \mathbf{u} - \mathbf{o} \end{bmatrix}$$

$$\mathbf{j}^T\mathbf{u} = 0 \text{ and } \mathbf{j}, \mathbf{u} >= 0 \tag{8}$$

where **x** is the vector of MNA nodal voltages and extra current variables, **j** is the vector of current variables of orthoators, **b** is the RHS vector of voltage sources and current sources, **u** is the vector of voltages across orthoators, **o** is the vector of voltages across voltage sources related to orthoators. The matrix **M** is the equivalent admittance matrix with all orthoators open or off. **A** is the incidence matrix of orthoators. **N** is the resistance matrix of linear resistors related to orthoators.

Suppose there are *m* orthoators and *n* MNA variables. Then the matrix **M** is of rank *n*n*, matrix **A** is of rank *m*n*, matrix **N** is of rank *m*m*. It should be noted here that matrix **A** is a very sparse matrix with at most two non-zero elements in each column, and matrix **N** is just a diagonal. If **M** is not singular, we can eliminate **x** from (8). This allows the MLCP matrix to be converted to a standard LCP as considered by van Bokhoven and Leenarets in [7]:

$$\mathbf{u} = \mathbf{D}\mathbf{j} + \mathbf{q} \tag{9}$$

where $\mathbf{D} = \mathbf{A}^T\mathbf{M}^{-1}\mathbf{A} + \mathbf{N}$, $\mathbf{q} = -\mathbf{A}^T\mathbf{M}^{-1}\mathbf{b} + \mathbf{o}$. Noting that **M** and **A** are both sparse admittance matrices, the matrix **D** and the vector **q** can be computed from at most four cofactors of the matrix **M** and its determinant. Since typical analog circuits only require a few orthoator macro circuits to represent the nonlinearity, then, only some cofactors and the determinant of matrix **M** need to be represented symbolically. This can be implemented efficiently using determinant decision diagrams introduced originally by Shi and Tan in [11].

4. Experimental Results

The proposed new method has been implemented into a prototype CAD program. Results from applying the resulting program to a behavioral model of the μa741 and a generic hard nonlinearity is presented in this section. For all the examples, our program reads in the circuit description in the SPICE-like format, sets up the MLCP formulation based on the framework of MNA, and then constructs symbolically all the solutions. Numerical results are obtained by repetitively evaluating the resulting symbolic expressions. We use the numerical simulations as a form to validate the symbolic expressions.

a) Example 1

The first example is a generic circuit that behaviorally models a strong nonlinearity. In other words, our output waveform should exhibit abrupt changes in its behavior. To compare this to SPICE-like algorithms we also implemented a smoothing algorithm, which is commonly performed for numerical simulators. The smoothing algorithm implemented was formulated in [6], which replaces the absolute operator by a hyperbolic cosine as done in [6]. Note that the modulus transform is related to Chua's model by

$$|w| = \frac{(u+j)}{2}, \quad w = \frac{(u-j)}{2} \rightarrow \lfloor x \rfloor = \frac{(|x| + x)}{2} \tag{10}$$

So $\lfloor x \rfloor$ is replaced with the following:

$$\lfloor x \rfloor \rightarrow \frac{\left(\frac{1}{\kappa}\ln(\cosh(\ \kappa x)) + x\right)}{2}, \tag{11}$$

where κ is the smoothing parameter. The closer κ is to zero the closer the expression evaluates to $\lfloor x \rfloor$. To illustrate the effect of the smoothing function the circuit in Figure 5a is simulated. Since there are two orthoators used in this example, then we are solving a 2-D LCP matrix as in (12). The symbolic expression representing the voltage at node V_3 is shown in equation (13), notice the expressions representing the voltage across the orthoators are encapsulated by the modulus transform.

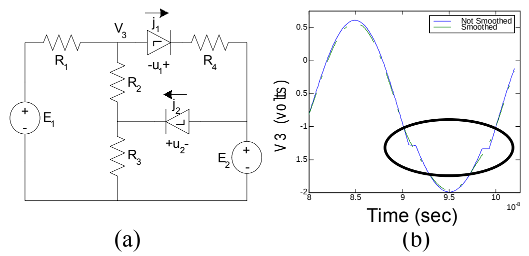


Figure 5. (a) Circuit used for smoothing analysis. (b) A section of the transient analysis.

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} \frac{R_1R_2 + R_1R_3 + R_1R_4 + R_2R_4 + R_3R_4}{R_1 + R_2 + R_3} & \frac{-R_1R_3}{R_1 + R_2 + R_3} \\ \frac{-R_1R_3}{R_1 + R_2 + R_3} & \frac{R_3R_1 + R_3R_2}{R_1 + R_2 + R_3} \end{bmatrix}\begin{bmatrix} j_1 \\ j_2 \end{bmatrix} + \begin{bmatrix} \frac{-(R_2 + R_3)}{R_1 + R_2 + R_3}E_1 + E_2 \\ \frac{R_3}{R_1 + R_2 + R_3}E_1 - E_2 \end{bmatrix} \tag{12}$$

$$V_3 = \frac{\left(\begin{array}{l} E_1R_2R_4 + E_2R_1(R_4+R_2) - \left\lfloor \frac{-R_1R_3}{R_1+R_2+R_3} * \left\lfloor \frac{-R_3E_1 + (R_1+R_2+R_3)E_2}{R_3R_1+R_3R_2} \right\rfloor - \frac{(R_2+R_3)}{R_1+R_2+R_3}E_1 + E_2 \right\rfloor R_1R_2 \\ + \left\lfloor \frac{-R_1R_3}{R_1+R_2+R_3} * \left\lfloor \frac{(R_2+R_3)E_1 - (R_1+R_2+R_3)E_2}{R_1R_2+R_1R_3+R_1R_4+R_2R_4+R_3R_4} \right\rfloor + \frac{R_3}{R_1+R_2+R_3}E_1 - E_2 \right\rfloor R_1R_4 \end{array}\right)}{R_2R_4 + R_1R_4 + R_1R_2} \tag{13}$$

A transient sweep is performed and the waveform at node V_3 is shown in Figure 5b. The solid curve is the results of using the modulus transform, while the dotted curve is the results from using the smoothing function. The smoothing parameter, κ, was set to 1. The smoothing function smoothes out the glitches seen in the modulus transform data as shown in the solid circle in Figure 5b. Taking the Fast Fourier Transform (FFT) of this data reveals differences in the distortion components. The normalized harmonic (HD) and intermodulation (IM) distortion components are shown in Figure 6. *What is opposite to the intuition is that smoothing actually yields larger distortion of the magnitude at third order harmonics and intermodulation distortions.*

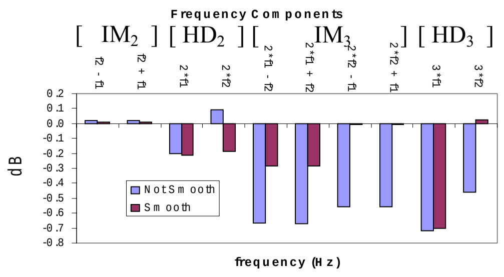


Figure 6. HD and IM distortion components of the Figure 5.

b) Example 2

The second example is a commonly used μa741 behavioral model shown in Figure 7 [13]. Note that it contains a nonlinear output resistor to simulate output limiting and a nonlinear transconductance simulating slew rate limiting. The parameters used for the model are taken from [13]. Figure 8a shows the time-domain waveforms when the input is a small-signal sin waveform computed by our method (PWL) and by SPICE. Clearly we can see that SPICE's smoothing leads to an over-estimation of the signal magnitude. Figure 8b shows the computed nonlinear behaviors when the input is applied to a large signal by both our method and SPICE. In this case, both simulators captured the limiting behaviors.

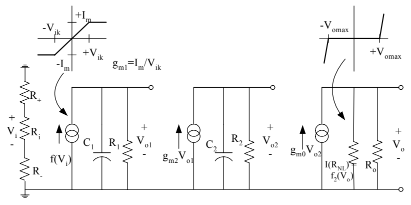


Figure 7. μa741 behavior model.

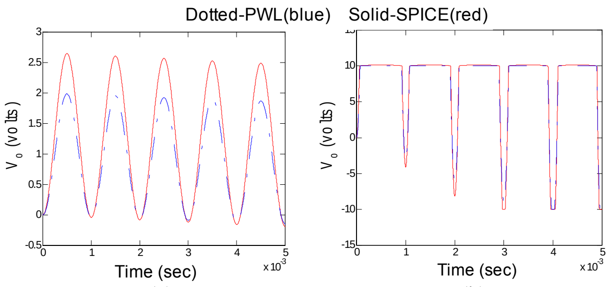


Figure 8. (a) μa741 result with the V_{in} amplitude = 1mV. (b) μa741 result with the V_{in} amplitude = 0.1V.

The harmonic distortion components of the μa741 time-domain results in Figure 8 are shown in Table 1. This clearly shows that SPICE and PWL obtain very similar results.

Table 1. Normalized Harmonic distortion of μa741.

Simulator	Fundamental	HD_2	HD_3
PWL	35.2808	0.9099	0.7274
Spice	35.2362	0.9102	0.7317

5. Conclusions and Future Work

In this paper, we presented a method for analyzing circuits with device and model hard nonlinearity characterized by piece-wise linear (PWL) I-V functions. The method is based on the modified nodal formulation of PWL circuits, where PWL devices are replaced by a network of linear resistors, (controlled) sources and orthoators. The resulting formulation is known as a mixed linear complementarity problem (MLCP), which can be converted to a standard LCP by implementing a determinant-decision diagram based procedure. Complementarity-decision diagrams were used to exploit the sharing of common sub-expressions of the LCP functions. The method has been implemented as a prototype tool and tested on a number of circuits.

6. References

[1] L.O. Chua, R. Ying, "Canonical piecewise-linear analysis", *IEEE Trans. on Circuits and Systems*, vol. 30, pp. 125-140, Mar. 1983.

[2] R. Cottle, J.-S. Pang and R. E. Stone, *The Linear Complementarity Problem*, Academic Press, 1992.

[3] E. Egoya, N. Le Gallou, J. M. Nebus, H. Buret, P. Reig, "Accurate RF and microwave system level modeling of wide band nonlinear circuits", in *Proc. IEEE Intl S. on Micro. Theory and Applics*, 2000.

[4] H. Floberg, S. Mattisson, "Symbolic distortion analysis of nonlinear elements in feedback amplifiers using describing functions", *Intl J. of Cir. Theory & Applications*, v. 23, pp. 345-356, 1995.

[5] K. Kundert, H. Chang, D. Jeffries, G. Lamant, E. Malavasi, and F. Dendig, "Design of mixed-signal systems-on-chips" *IEEE Trans. Computer-Aided Design,* vol. 19, no. 12, pp. 1561-1571, Dec. 2001

[6] Lazaro, Santamaria, Pantaleon, Sanchez, "Smoothing the canonical piecewise-linear model: An efficient and derivable large-signal model for MESFET/HEMT transistors", *IEEE Trans. Cir. & Sys-I: Fundmtl. Theory & Applics.*, vol. 48, no. 2, pp.184-192, Feb. 2001.

[7] D. M. W. Leenaerts and W. M.G. van Bokhoven, *Piecewise Linear Modeling and Analysis*, Kluwer Academic Publishers, 1998.

[8] A. Manthe, Z. Li, C.-J. Shi, K. Mayaram, "Symbolic Analysis of Nonlinear Analog Circuits", *DATE*, Mar. 2003.

[9] K. Mayaram, "Distortion" in *The Electrical Engineering Handbook,* Editor, R. C. Dorf, Second Edition, CRC Press, pp. 147-157, 1997.

[10] J. Roychowdhury, "Efficient methods for simulating highly nonlinear multi-rate circuits", *Proc. IEEEE/ACM DAC*, 1997.

[11] C.-J. Shi and X.-D. Tan, "Canonical symbolic analysis of large analog circuits with determinant decision diagrams", *IEEE Trans. Computer-Aided Design*, vol. 19, pp. 1-18, Jan. 2000.

[12] M. T. Terrovitis and R. G. Meyer, "Intermodulation distortion in current-cummutating CMOS mixers", *IEEE J. Solid-State Circuits*, pp. 1461-1473, vol. 35, no. 10, Oct. 2000.

[13] J. Vlach and K. Singhal, *Computer Methods for Circuit Analysis and Design,* New York: N.Y.: Van Nostrand Reinhold, 1991 (Ch. 11).

Submitted to IEEE Trans. Circuits and Systems Part 1 (in review)

A New Topological Approach to Symbolic Network Analysis*

Guoyong Shi and C.-J. Richard Shi

Department of Electrical Engineering
University of Washington
Seattle, WA 98195, U.S.A.
`E-mail: {gshi,cjshi}@ee.washington.edu`

October 14, 2004

Abstract

Classical topological network analysis algorithms use graph representations of a circuit to compute the determinant and cofactors needed for forming a symbolic network function. A number of methods have been proposed in the literature over the past half century. Most of them are not widely used in application because they are either conceptually too complicated or not easily implementable by a computer program. This paper develops a new topological enumeration algorithm for deriving symbolic network functions without computing any cofactors. The enumeration algorithm applies to circuits containing independent sources, immittances, all types of dependent sources, nullors, and transformers, without the need of element conversion. The terms enumerated are free of cancellation and vanishing terms are automatically excluded. The enumeration rules presented here are conceptually simple and can be easily implemented. An algebraic proof is provided to justify the correctness and completeness of the enumeration rules. Efficient algorithms are developed for implementing the rules. A storage scheme using decision diagram is proposed for internal representation of a symbolic network function. The algorithms and storage scheme have been implemented in a symbolic simulator and its performance is demonstrated by several circuit examples.

Index Terms – Circuit simulation, decision diagram, network topology, symbolic analysis, tree enumeration.

*This work was supported by DARPA NeoCAD Program under Grant No. N66001-01-8920.

I. Introduction

Topological approach to electrical networks analysis is concerned with the determination of the network characteristics from the knowledge of elements and their connections without applying numerical methods and any other intermediate formulation such as tableau matrix or modified nodal analysis. In all topological methods, the network is represented by a graph that resembles the network, and the calculation of any network function is transformed into a subgraph enumeration problem.

Topological method is one of the several key methodologies used in the whole body of symbolic network research. Besides its limitation, symbolic analysis has several inherent advantages such as insight into the network behavior, control of error in numerical calculations, easy generation of sensitivities, and computational reduction in statistical analysis and parameter optimization, etc. While numerous methods have been published in the literature, this subject is still constantly being revisited. A comprehensive review of most classical and recent symbolic analysis methods can be found in the textbook [1] and the survey paper [2].

Most classical topological methods derive the network function of a circuit by representing the cofactors and determinant of the node admittance matrix in graphical form, typically in trees or directed trees. Topological formulas for passive networks can be dated back to the time of Kirchhoff and Maxwell over 100 years ago. Research on topological methods for active networks started from 1950's, along with the rapid progress in the theoretical graph research in the following several decades. The 2-graph method was the first topological method to address active networks [3, 4]. It originally deals with networks containing admittances and voltage controlled current sources (VCCS). For constructing 2-graphs, all admittances are modeled as VCCS's. Then all controlling voltage edges become edges of the voltage graph, and all controlled current edges become edges of the current graph. The voltage and current graphs constitute a 2-graph. The terms of the nodal admittance determinant are obtained by enumerating all common spanning trees of the 2-graph. The cofactors needed for calculating the network functions can also be derived using the 2-graph concept. Although the 2-graph method has the cancellation-free property, i.e. no two terms have the exact symbol combination but opposite signs, the complexity of sign rules makes the 2-graph method not widely used. Yu and Sechen applied the 2-graph methodology in an approximate symbolic analysis framework after extension of certain rules [5].

Coates's [6] paper originated the 2-tree concept for topological network analysis. This paper also discussed sign determination when pairs of edges are involved in linear active networks. It is perhaps the earliest paper that introduced a fundamental concept called *graph pair* which was extended by many authors in the following decades. A slight extension of the topological formula for passive networks was proposed by Brown [7] to include two, three, and four terminal components in the network. The applicability of this method is, however, very limited. Chen [8] proposed a directed tree method based on the theory of *equicofactor* matrix. This method was originally developed for RLC-g_m networks, where determinants and cofactors are found by enumerating all directed trees and 2-trees without the need to consider the signs. However, a serious drawback of this method is the cancellation problem, i.e. some terms resulting from enumeration cancel each other in the final expression, which is not efficient from a computation point of view.

Motivated by the limitation of purely enumeration-based methods, Shieu and Chan in [9] developed a topological method based on k-tree terms which potentially reduces the number of terms in the k-tree method. However, in addition to its conceptual complexity, this method only applies to active elements involving voltage controlled current sources (VCCS). Also cancellation-free is not a direct consequence.

Talbot's paper [10] is one of the papers in the old literature that has the most relevance to the work in this paper. Talbot introduced *tree* and *tree-pair* concepts for analyzing general linear networks. However, except for the selection table technique introduced there, Talbot did not provide systematic rules for enumerating all trees and tree-pairs from a circuit containing commonly used circuit elements, such as dependent sources. Also no systematic algorithms were developed for computer implementation. Another work that has certain relevance to this paper is [11], where the authors formulated topological formulas from tableau matrix by means of tree and cotree. However, their formulas were cofactor-based, and could not exclude vanishing terms due to the lack of a deep analysis of the inherent topological structure among the mutual coupling branches. The rules developed here can be considered as a further development along this direction.

An elementary exposition of some of the representative topological methods is presented in Lin's textbook [1]. A careful inspection of those classical methods (except the 2-graph method) reveals that the cancellation problem essentially originates from the fact that trees are constructed from the nodal admittance matrix, and the product terms obtained from the graphical methods are directly associated with the terms in the Laplace expansion of the nodal matrix determinant, which has the inherent term cancellation problem.

The new set of topological rules to be presented in this paper, in addition to its conceptual simplicity and ease of implementation, avoids the term cancellation problem and the evaluation of vanishing terms; thus can be considered as the most efficient among all exact topological enumeration methods. The topological approach considered in this paper is derived from the tableau matrix formulation. It has been noticed by several authors that expansion of a tableau matrix determinant results in cancellation-free terms. Since the tableau matrix is directly related to the network topology, product terms from the expansion of a tableau matrix determinant can be mapped to certain subgraphs in the original network. Such an idea has been studied recently in several publications. The first work along this line was by Wambacq et al. [12] where an enumeration algorithm was described based on the structure of a full tableau matrix, but no systematic rules were developed for analyzing general circuits. More recently Yin and his colleagues [13, 14] presented a set of enumeration rules using the concept of valid trees and tree-pairs. But the rules described there are ambiguous and are not rigorously proved. Although it is intuitively possible to summarize certain rules by (Laplace) expanding a tableau matrix as did in [12], a formal proof is necessary to guarantee that the rules are *correct* and *complete*.

Recent research efforts on symbolic methods are targeted at large networks which typically contain 20 to 40 transistors. Due to the explosive increasing of symbol combinations along with the circuit size, new techniques such as symbol composition, approximate analysis, and hierarchical analysis have been proposed recently for symbolic analysis of large networks. Representative works along this line include the application of decision diagram in [15], error-controlled term truncation in [5], and a non-topological approach based on network partitioning in [16]. The success of determinant-based approach to large networks is largely attributed to the significant lumping effect resulting from the modified nodal analysis (MNA) formulation and the inherent sparsity of practical networks. But topological information is lost in such a formulation. The advantage of topological methods such as the one developed in this paper is that the symbols can directly be associated with network elements because of the unique symbol-to-element correspondence.

The topological method studied in this paper is a continuation and extension of many previous works. The new contribution contains several aspects. First, we systematically present in Section II a set of rules that are necessary and complete for deriving a symbolic network function of a circuit containing immitance elements, all types of dependent sources, an independent source, and nullors. All of the rules are stated accurately for the purpose of implementation. Second, a rigorous algebraic proof is presented in Section III which shows the correctness and completeness of all the enumeration rules. A new technique is introduced in the proof which is powerful in its own regard for symbolic network research. Third, two efficient algorithms are introduce in Section IV; one for term enumeration and the other for the determination of a term sign. These algorithms are described in detail for computer implementation. Fourth, we propose an efficient scheme for term storage and manipulation using *Decision Diagram* in Section V. Some preliminary experimental results are reported in Section VI. This paper is concluded in Section VII with some concluding remarks.

II. Enumeration Rules

In this section we develop a set of new rules for enumerating all cancellation-free terms directly from a given circuit topology. The rules apply to circuits that contain impedances, admittances, four types of dependent sources, independent sources, and nullators and norators (nullors). The rules to be stated are based on the the following basic assumptions.

Assumption 1 (Basic Assumptions)

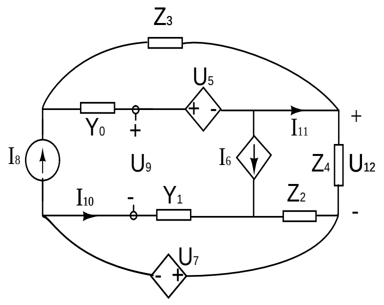
(a) A controlling branch only controls one branch.

(b) A controlled branch is controlled by only one branch.

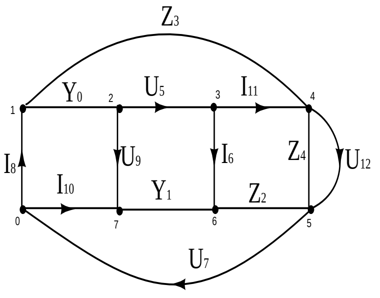
Note that these assumptions are without loss of generality; they are adopted solely for conceptual clarity in the presentation. If multiple branches are involved in the dependent sources, it should be straightforward to remodel them in terms of *one-to-one* dependence.

Before listing the rules, we use an example to introduce some necessary terminologies. The circuit shown in Fig. 1(a) contains three dependent sources, one independent current source, some admittances represented by Y_i, and some impedances represented by Z_j [14]. We would like to find the transfer function from the independent current source I_8 to the voltage across the element Z_4, denoted by U_{12}. The three dependent source pairs are specified as follows:

$$\begin{aligned} U_5 &= E_{5,9}U_9 \quad (VCVS), \\ I_6 &= F_{6,10}I_{10} \quad (CCCS), \\ U_7 &= H_{7,11}I_{11} \quad (CCVS). \end{aligned}$$



(a)



(b)

Fig. 1. A circuit example with four-types of dependent sources: (a) Circuit, (b) Graph.

To find the network transfer function from I_8 to U_{12}, i.e. $H(s) = U_{12}(s)/I_8(s)$, we use an idea similar to the *sorting scheme* (see [1], Section 3.4.5) by introducing another dependent pair

$$I_8 = G_{8,12}U_{12} \quad (VCCS).$$

(Here we cannot treat the pair as a CCVS because by convention the controlling current should have zero voltage, see [1], page 64.) If we can find a symbolic expression for $G_{8,12}$, then the transfer function from I_8 to U_{12} is simply

$$H(s) = \frac{U_{12}}{I_8} = \frac{1}{G_{8,12}}.$$

After including the dependent pair from the input to the output, we have all four types of dependent sources in the circuit. In general a circuit has to to be converted to a graph before the product terms are enumerated. Shown in Fig. 1(b) is the converted graph of the circuit in Fig. 1(a). The graph construction rules summarized below can by implemented by a parser that automatically parses a standard Spice netlist into a graph.

Graph Construction Rules:

(i) The edges associated with controlling or controlled sources are directed, i.e. *a voltage edge is directed from + to −, and a current edge is directed along the current direction assigned.*

(ii) Add an edge for each controlling voltage, such as U_9 and U_{12} in Fig. 1(b). (Remember that no current flows through such edges.)

(iii) Each controlling current also takes a single edge, such as I_{10} and I_{11} in Fig. 1(b). (Note that in Spice netlist, a controlling current source is specified by a voltage name.)

(iv) All the edges are appropriately labelled with edge names and edge types, and linked to their dependent edges.

(v) Ideal opamps are modeled by nullors, i.e. a pair of nullator and norator (Fig. 2). A nullor is represented by two edges, NU (nullator) and NO (norator), in the graph.

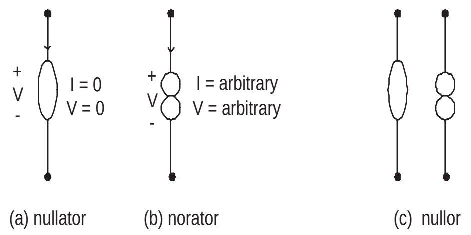


Fig. 2. Nullator, norator, and nullor.

In the rest of the paper, by a *dependent pair* we mean the two edges associated with a dependent source or a nullor. In the context of symbolic analysis, edges in a graph are weighted and the weights come from the physical symbolic coefficients defining V-I relationships, such as admittances, impedances, and controlled source gains. Traditionally, the symbolic term associated to a subgraph is defined to be the product of all symbolic edge weights in the subgraph. Tree is a commonly used subgraph in most classical symbolic analysis methods. In this paper by *tree* we always mean a spanning tree. Because of the dependent sources, we also have to deal with pairs of trees, namely, *tree-pairs*. To be specific, we shall use the shorthand CC, VC, CS, and VS to name those edges with dependent sources, and NU and NO to name those edges with nullors. Since not all trees and tree-pairs of a circuit graph contribute to a product term, we only enumerate those trees and tree-pairs that contribute to nonvanishing product terms. Such contributing trees and tree-pairs are called *term trees* and *term tree-pairs*. The first rule defines a term tree. It is important to note that if a circuit has at least one nullor, then no term tree will be enumerated because all terms are determined by tree-pairs.

Rule 1 (Tree) *Tree exists only if the circuit does not have any nullor. Regardless of the types of dependent sources, a term tree must contain all CC and VS edges (if any), but must <u>not</u> contain any VC and CS edges.*

The product term defined by a term tree is specified by the next rule.

Rule 2 (Tree Term) *The term determined by a term tree is the product of all Y_i's and Z_j^{-1}'s on the tree, i.e. all admittances on the tree. Those CC and VS edges appearing on the tree have weight one.*

Note that the inversion of each Z element does not need any extra treatment in symbolic manipulation. Only in the numerical evaluation should its value be inverted.

The second rule defines a *term tree-pair*. This is a rule with certain degree of delicacy. A general statement is that any Y/Z edges, if appear, must be common to both trees, but edges with the dependent sources could have two cases; either some edge like CC/VS appears as a common edge, or otherwise appears as an edge in pair with its dependent edge appearing on the other tree. To minimize confusion, we specifically name the two trees in a tree-pair the *L-tree* (left tree) and the *R-tree* (right tree). Also, as a rule, we require that all *controlling* edges, if appearing in pairs, be in the R-tree, while all *controlled* edges, if appearing in pairs, be in the L-tree. With these specifications, the next rule defines a term tree-pair.

Rule 3 (Tree-Pair) *A term tree-pair consists of an L-tree and an R-tree.*

(*i*) *All Y and Z edges appearing on a term tree-pair must be common to both trees.*

(*ii*) *All NU and NO edges must appear on all term tree-pairs, with the NU edges on the R-tree and NO edges on the L-tree.*

(*iii*) *All CC and VS edges must appear on the term tree-pair, but they can appear either as common edges or as pairing edges, exclusively. If in pair, CC must be on the R-tree and VS on the L-tree.*

(*iv*) *Any VC and CS edges can either not appear or appear in pair with their dependent edges. If a VC appears it must be on the R-tree, while if a CS appears it must be on the L-tree.*

This rule is elaborated for clarity. For example, for a CCVS dependent pair, it could appear in the tree-pair in one of the two cases: either in (CC, CC), (VS, VS) as common edges or in (VS, CC) as a pair, where by (VS, CC) we mean a dependent pair with VS in the L-tree and CC in the R-tree, and by (CC, CC) we mean CC appears as a common edge on both trees. Because both CC and VS edges *must* appear in a term tree-pair, if CC appears as a common edge in the tree-pair, then the dependent VS edge must also appear as a common edge. Otherwise, only the edge pair (VS, CC) is allowed to appear in the tree-pair. Keep in mind that if a CC or VS edge appears as a common edge, then it cannot show up as a pairing edge on the same tree-pair simultaneously, i.e. the two cases are exclusive. In the case of a CCCS it could either appear as (CC, CC) or as (CS, CC), but not simultaneously, because CS is not a *must-appear* edge. The case for a VCVS is analogous; it could appear either as (VS, VS) or as (VS, VC), but not simultaneously, because VC is not a *must-appear* edge. For illustration, one tree-pair of the graph in Fig 1(b) is shown in Fig. 3.

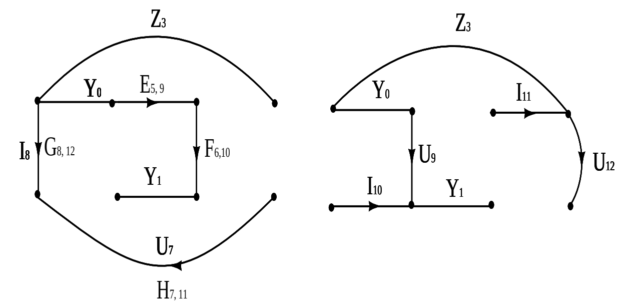


Fig. 3. An example of a term tree-pair.

Remark 1 *The case that CC and VS edges forming a loop can be excluded because, if this is the case, the network is either inconsistent or has no unique solution, a pathological case of no practical importance.*

The product term determined by a term tree-pair is defined in a more delicate way by the following three rules.

Rule 4 (Tree-Pair Term) *The term determined by a term tree-pair is the product of the following three parts:*

(*i*) *A term sign.*

(*ii*) *All common Y_i's and Z_j^{-1}'s on the tree-pair.*

(*iii*) *The signed gains of all dependent sources appearing on the tree-pair.*

Again, the weights for all common CC and VS edges are one. The weight of each nullor pair is also one. The signed gains *are defined by Rule 5 and the* term sign *is defined by Rule 6.*

Rule 5 (Signed Gain) *The following signed gains are used for the dependent sources appearing on a term tree-pair:*

$$\begin{array}{rcl} VCVS & \leftrightarrow & -E_{j,k} \\ CCCS & \leftrightarrow & +F_{j,k} \\ VCCS & \leftrightarrow & +G_{j,k} \\ CCVS & \leftrightarrow & -H_{j,k} \end{array}$$

In other words, those gains associated with a VS are signed minus; otherwise they are signed plus.

Remark 2 *Since all term tree-pairs contain all nullor pairs (if any), the weight one of any nullor need not be signed. The product of all nullor signs is ultimately eliminated by equating the sum of products to zero.*

Remark 3 *The minus sign for VS edges can be removed if we direct the VS edges from '$-$' to '$+$', by reversing the direction stipulated in the Graph Construction Rules. However, we keep the sign rule as stated in Rule 5 to follow the convention.*

The sign of a product term defined by a term tree-pair is determined by two nonzero majors from the reduced incidence matrix of a circuit graph. A reduced incidence matrix results from the graph incidence matrix by deleting one row (normally the ground row) (see [1], page 19).

Rule 6 (Term Sign) *Let A_L and A_R be the two reduced incidence matrices of the L-tree and R-tree, respectively, with their rows (node numbers) in exactly the same order and their columns (labelled by the edge names) aligned as follows: the common Y, Z, CC, VS edges of the tree-pair take the same column numbers, while those edges in pair also take the same column number but with the controlling edges in A_R and the controlled edges in A_L. The directions of those directed edges are kept, while the directions for those undirected Y/Z edges are arbitrary but fixed when writing the incidence matrices. The sign for the tree-pair term is the product of the determinants of A_L and A_R, i.e.* $\det|A_L| \cdot \det|A_R|$*. Note that each determinant takes either $+1$ or -1.*

Remark 4 *According to Rule 6, two determinants have to be evaluated every time a term tree-pair is enumerated. However, in implementation advantage can be taken of the special structure of the matrices A_L and A_R which contain only ±1 as nonzero elements. A fast relabelling algorithm is presented in Section IV.*

III. Proof of the Enumeration Rules

This section is devoted to an algebraic proof of the enumeration rules. The rules will be derived from the Binet-Cauchy Theorem [17]. Before the theorem can be applied, some algebraic manipulation will be carried out. To simplify the derivation, we shall introduce an ε symbol in the algebraic manipulation. The role of ε symbol can be illustrated by a simple example. The evaluation of the following determinant

$$\begin{vmatrix} a & b \\ c & 0 \end{vmatrix} = -bd. \tag{1}$$

can be carried out in an alternate way

$$\begin{vmatrix} a & b \\ c & \varepsilon \end{vmatrix} = \varepsilon(a - b\varepsilon^{-1}d) = a\varepsilon - bc. \tag{2}$$

followed by taking limit $\varepsilon \to 0$. The replacement of 0 by ε makes it possible to apply some determinant identities such as

$$\begin{vmatrix} A & B \\ C & D \end{vmatrix} = |A|\,|D - CA^{-1}B| \tag{3a}$$

$$\begin{vmatrix} A & B \\ C & D \end{vmatrix} = |D|\,|A - BD^{-1}C| \tag{3b}$$

when either A or D is nonsingular. We shall see later on in the proof that the presence of an ε symbol simplifies the algebraic manipulation meanwhile making the identification of the enumeration rules straightforward.

The branch equations of a circuit (converted to a graph) can be written in matrix form

$$ZI + YU = 0, \tag{4}$$

where I and U are the branch current and voltage vectors

$$\begin{aligned} I^{\mathrm{T}} &= \begin{bmatrix} I_1 & \cdots & I_E \end{bmatrix}, \\ U^{\mathrm{T}} &= \begin{bmatrix} U_1 & \cdots & U_E \end{bmatrix}, \end{aligned}$$

where E is the number of edges in the graph created according to the *Graph Construction Rules* stated in Section II. Since a circuit is allowed to have Y, Z elements, all four dependent source, and nullors, each branch equation takes one of the following forms

$$U_i = Z_i I_i \qquad \text{(Z edge)} \tag{5a}$$

$$I_i = Y_i U_i \qquad \text{(Y edge)} \tag{5b}$$

$$U_i = E_{i,j} U_j, \ I_j = 0 \quad \text{(VCVS)} \tag{5c}$$

$$I_i = F_{i,j} I_j, \ U_j = 0 \quad \text{(CCCS)} \tag{5d}$$

$$I_i = G_{i,j} U_j, \ I_j = 0 \quad \text{(VCCS)} \tag{5e}$$

$$U_i = H_{i,j} I_j, \ U_j = 0 \quad \text{(CCVS)} \tag{5f}$$

$$U_j = 0, \ I_j = 0 \qquad \text{(NU edge)} \tag{5g}$$

where we use the Spice convention for the gain factors. Note that a (NO, NU) pair can be viewed as the limiting case of a VCVS by letting the gain $E_{i,j}$ go to infinity. Since the current and voltage associated with a norator edge are arbitrary, there is no specific equation for a norator edge. We shall see that treating all Z edges as Y edges simplifies the algebraic manipulation. Thus the branch equation for a Z edge in (5a) will be written as $I_i = Z_i^{-1}U_i$, i.e. all impedances are treated as admittance in the graph. Since R and L elements directly appear in a circuit, for convenience we keep on naming all R and L edges as Z edges but using Z_i^{-1} for evaluation (see Rules 2 and 4 in Section II).

The appearance of the preceding seven types of elements in the matrices Z and Y can be described by the stamps listed below:

$$\begin{array}{ccccc}
\text{Stamp in } Z & \leftrightarrow & \text{Stamp in } Y & & \\
\begin{bmatrix} \ddots & & \\ & 1 & \\ & & \ddots \end{bmatrix} & \leftrightarrow & \begin{bmatrix} \ddots & & \\ & -Y_i & \\ & & \ddots \end{bmatrix}, & & (Y) \\
\begin{bmatrix} \ddots & & \\ & 1 & \\ & & \ddots \end{bmatrix} & \leftrightarrow & \begin{bmatrix} \ddots & & \\ & -Z_i^{-1} & \\ & & \ddots \end{bmatrix}, & & (Z) \\
\begin{bmatrix} \varepsilon & & \\ & \ddots & \\ & & 1 \end{bmatrix} & \leftrightarrow & \begin{bmatrix} -1 & \cdots & E_{i,j} \\ & \ddots & \vdots \\ & & 0 \end{bmatrix}, & & (VCVS) \\
\begin{bmatrix} 1 & \cdots & -F_{i,j} \\ & \ddots & \vdots \\ & & \varepsilon \end{bmatrix} & \leftrightarrow & \begin{bmatrix} 0 & & \\ & \ddots & \\ & & -1 \end{bmatrix}, & & (CCCS) \\
\begin{bmatrix} 1 & & \\ & \ddots & \\ & & 1 \end{bmatrix} & \leftrightarrow & \begin{bmatrix} 0 & \cdots & -G_{i,j} \\ & \ddots & \vdots \\ & & 0 \end{bmatrix}, & & (VCCS) \\
\begin{bmatrix} \varepsilon & \cdots & H_{i,j} \\ & \ddots & \vdots \\ & & \varepsilon \end{bmatrix} & \leftrightarrow & \begin{bmatrix} -1 & & \\ & \ddots & \\ & & -1 \end{bmatrix}, & & (CCVS) \\
\begin{bmatrix} \varepsilon & & \\ & \ddots & \\ & & 1 \end{bmatrix} & \leftrightarrow & \begin{bmatrix} 0 & \cdots & 1 \\ & \ddots & \vdots \\ & & 0 \end{bmatrix}, & & (Nullor)
\end{array} \tag{6}$$

Here we have assumed that the controlling edges are always numbered greater than the controlled edges (i.e. $i < j$) so that all off-diagonal elements appear in the upper-right triangular part of matrices Y and Z. Note that we have intentionally placed all the minus signs to the diagonal elements of Y and replaced all the *zeros* on the diagonal of Z by ε. Thus the matrix Z is nonsingular for $\varepsilon > 0$. Remember that we shall finally take limit $\varepsilon \to 0$ to recover the original circuit equations. We also point out that there are other forms of stamp for a nullor, e.g.

$$\begin{bmatrix} \varepsilon & \cdots & 1 \\ & \ddots & \vdots \\ & & \varepsilon \end{bmatrix} \leftrightarrow \begin{bmatrix} 0 & \cdots & 0 \\ & \ddots & \vdots \\ & & 1 \end{bmatrix}, \quad (Nullor)$$

However, it is easy to verify that the rules derived would be independent of the stamp forms. Thus we keep on using the previous stamp for a nullor.

Diagonalization of matrix Z greatly simplifies the algebraic manipulation later on. We can apply row transformations to eliminate those nonzero off-diagonals in Z coming with the stamps of CCCS and CCVS. Note that the presence of the ε symbol makes this operation possible. The two stamps after the row transformations now become

$$\begin{array}{l} \begin{bmatrix} 1 & \cdots & 0 \\ & \ddots & \vdots \\ & & \varepsilon \end{bmatrix} \leftrightarrow \begin{bmatrix} 0 & \cdots & -\varepsilon^{-1}F_{i,j} \\ & \ddots & \vdots \\ & & -1 \end{bmatrix}, \quad (CCCS) \\ \begin{bmatrix} \varepsilon & \cdots & 0 \\ & \ddots & \vdots \\ & & \varepsilon \end{bmatrix} \leftrightarrow \begin{bmatrix} -1 & \cdots & \varepsilon^{-1}H_{i,j} \\ & \ddots & \vdots \\ & & -1 \end{bmatrix}, \quad (CCVS) \end{array} \tag{7}$$

In passing we observe the following special features associated with the stamps, revealing of the enumeration rules.

(a) Those diagonal ε's of Z-stamps are only associated with CC and VS edges, except the nullor.

(b) Those diagonal 0's of Y-stamps are only associated with VC and CS edges, except the nullor.

(c) A nullor has both diagonals of the Y-stamp zero, and an ε in the Z-stamp associated with the NO-edge.

(c) The signs of the terms $E_{i,j}$ and $\varepsilon^{-1}H_{i,j}$ associated with VS are both *positive*, while the other terms $\varepsilon^{-1}F_{i,j}$ and $G_{i,j}$ associated with CS are both *negative*. This fact has connection to the gain sign rule stated in Rule 5.

We now ready to proceed to an algebraic proof. Given a connected network, after removing a spanning tree from the network, the remaining branches form a subgraph called *cotree*. Let I_t (resp. U_t) and I_c (resp. U_c) be the vectors of branch currents (resp. voltages) on the tree and cotree, respectively. By tableau formulation, the network equation can be written as

$$\begin{bmatrix} A_t & & & A_c \\ & B_c & B_t & \\ Z_{(1)} & Y_{(2)} & Y_{(1)} & Z_{(2)} \\ Z_{(3)} & Y_{(4)} & Y_{(3)} & Z_{(4)} \end{bmatrix} \begin{bmatrix} I_t \\ U_c \\ U_t \\ I_c \end{bmatrix} = 0 \tag{8}$$

where $A = \begin{bmatrix} A_t & A_c \end{bmatrix}$ is the *reduced incidence matrix* with A_t corresponding to a preselected tree, and $B = \begin{bmatrix} B_c & B_t \end{bmatrix}$ is the *fundamental loop matrix* with B_c corresponding to the cotree (see the textbook [1]). An empty block indicates all-zero entries. Note that $\det|A_t| = \pm 1$ and $\det|B_c| = \pm 1$.

Let $I^{\mathrm{T}} = \begin{bmatrix} I_t^{\mathrm{T}} & I_c^{\mathrm{T}} \end{bmatrix}$ and $U^{\mathrm{T}} = \begin{bmatrix} U_t^{\mathrm{T}} & U_c^{\mathrm{T}} \end{bmatrix}$. The third and fourth block-rows in (8) come from the branch equation (4) written in compatible blockwise form

$$\begin{bmatrix} Z_{(1)} & Z_{(2)} \\ Z_{(3)} & Z_{(4)} \end{bmatrix} \begin{bmatrix} I_t \\ I_c \end{bmatrix} + \begin{bmatrix} Y_{(1)} & Y_{(2)} \\ Y_{(3)} & Y_{(4)} \end{bmatrix} \begin{bmatrix} U_t \\ U_c \end{bmatrix} = 0 \tag{9}$$

Let M be the coefficient matrix in (8). We are interested in computing in symbolic form the determinant of M, i.e.

$$\det|M| = \begin{vmatrix} A_t & & & A_c \\ & B_c & B_t & \\ Z_{(1)} & Y_{(2)} & Y_{(1)} & Z_{(2)} \\ Z_{(3)} & Y_{(4)} & Y_{(3)} & Z_{(4)} \end{vmatrix}. \tag{10}$$

It would be hard to see a connection between the expansion of this determinant and the enumeration rules without algebraic simplifications of M. To this end, we assume that those diagonal zeros of $Z_{(1)}$ and $Z_{(4)}$ have been replaced by ε and the row transformations mentioned before have been carried out to make the Z matrix diagonal. Then the branch equation in (9) becomes

$$\begin{bmatrix} \hat{Z}_{(1)} & \\ & \hat{Z}_{(4)} \end{bmatrix} \begin{bmatrix} I_t \\ I_c \end{bmatrix} + \begin{bmatrix} \widetilde{Y}_{(1)} & \widetilde{Y}_{(2)} \\ \widetilde{Y}_{(3)} & \widetilde{Y}_{(4)} \end{bmatrix} \begin{bmatrix} U_t \\ U_c \end{bmatrix} = 0 \tag{11}$$

where $\hat{Z}_{(1)}$ and $\hat{Z}_{(4)}$ are the diagonal matrices formed by the diagonal elements of $Z_{(1)}$ and $Z_{(4)}$, respectively, with the zeros substituted by ε. Because of the transformation, some off-diagonals of $\widetilde{Y}$ are also ε-dependent (but not explicitly indicated here), see the stamps in (7).

Note that if we apply the same transformation to M, the determinant of M will not change. Therefore it holds that

$$\det|M| = \begin{vmatrix} A_t & & & A_c \\ & B_c & B_t & \\ \hat{Z}_{(1)} & \widetilde{Y}_{(2)} & \widetilde{Y}_{(1)} & \\ & \widetilde{Y}_{(4)} & \widetilde{Y}_{(3)} & \hat{Z}_{(4)} \end{vmatrix}. \tag{12}$$

Noting that the submatrices A_t, B_c, $\hat{Z}_{(1)}$, and $\hat{Z}_{(4)}$ are all nonsingular, we can rewrite $\det|M|$ as

$$\det|M| = \det|A_t| \det|B_c| \det|\hat{Z}_{(1)}| \det|\hat{Z}_{(4)}| \times \begin{vmatrix} I & & & Q \\ & I & -Q^{\mathrm{T}} & \\ I & \hat{Z}_{(1)}^{-1}\widetilde{Y}_{(2)} & \hat{Z}_{(1)}^{-1}\widetilde{Y}_{(1)} & \\ & \hat{Z}_{(4)}^{-1}\widetilde{Y}_{(4)} & \hat{Z}_{(4)}^{-1}\widetilde{Y}_{(3)} & I \end{vmatrix}, \tag{13}$$

where $Q = A_t^{-1}A_c$ and $B_c^{-1}B_t = -Q^{\mathrm{T}}$ (see Corollary 2.8 in [1]).

Let M_1 be the matrix in the last determinant of (13). Applying the identities in (3), we obtain

$$\begin{aligned} & \det|M_1| \\ = & \left| \begin{bmatrix} \hat{Z}_{(1)}^{-1}\widetilde{Y}_{(1)} & \\ \hat{Z}_{(4)}^{-1}\widetilde{Y}_{(3)} & I \end{bmatrix} - \begin{bmatrix} I & \hat{Z}_{(1)}^{-1}\widetilde{Y}_{(2)} \\ & \hat{Z}_{(4)}^{-1}\widetilde{Y}_{(4)} \end{bmatrix} \begin{bmatrix} & Q \\ -Q^{\mathrm{T}} & \end{bmatrix} \right| \\ = & \begin{vmatrix} \hat{Z}_{(1)}^{-1}\left(\widetilde{Y}_{(1)} + \widetilde{Y}_{(2)}Q^{\mathrm{T}}\right) & -Q \\ \hat{Z}_{(4)}^{-1}\left(\widetilde{Y}_{(3)} + \widetilde{Y}_{(4)}Q^{\mathrm{T}}\right) & I \end{vmatrix} \\ = & \left| \hat{Z}_{(1)}^{-1}\left(\widetilde{Y}_{(1)} + \widetilde{Y}_{(2)}Q^{\mathrm{T}}\right) + Q\hat{Z}_{(4)}^{-1}\left(\widetilde{Y}_{(3)} + \widetilde{Y}_{(4)}Q^{\mathrm{T}}\right) \right| \\ = & \left| \hat{Z}_{(1)}^{-1}\widetilde{Y}_{(1)} + \hat{Z}_{(1)}^{-1}\widetilde{Y}_{(2)}Q^{\mathrm{T}} + Q\hat{Z}_{(4)}^{-1}\widetilde{Y}_{(3)} + Q\hat{Z}_{(4)}^{-1}\widetilde{Y}_{(4)}Q^{\mathrm{T}} \right| \\ = & \left| \begin{bmatrix} I & Q \end{bmatrix} \begin{bmatrix} \hat{Z}_{(1)}^{-1} & \\ & \hat{Z}_{(4)}^{-1} \end{bmatrix} \begin{bmatrix} \widetilde{Y}_{(1)} & \widetilde{Y}_{(2)} \\ \widetilde{Y}_{(3)} & \widetilde{Y}_{(4)} \end{bmatrix} \begin{bmatrix} I \\ Q^{\mathrm{T}} \end{bmatrix} \right| \\ = & \left| \begin{bmatrix} A_t & A_c \end{bmatrix} \begin{bmatrix} \hat{Z}_{(1)}^{-1} & \\ & \hat{Z}_{(4)}^{-1} \end{bmatrix} \begin{bmatrix} \widetilde{Y}_{(1)} & \widetilde{Y}_{(2)} \\ \widetilde{Y}_{(3)} & \widetilde{Y}_{(4)} \end{bmatrix} \begin{bmatrix} A_t^{\mathrm{T}} \\ A_c^{\mathrm{T}} \end{bmatrix} \right| \end{aligned} \tag{14}$$

where in the last step of (14) we used the fact that $|A_t||A_t| = 1$. We stress that equation (14) is crucial for mapping the terms in the expansion of the determinant $\det|M|$ to certain trees or tree-pairs found from the original network topology.

Since $|A_t||B_c| = \pm 1$, the identity of $\det|M| = 0$ is equivalent to

$$\det|\hat{Z}_{(1)}| \det|\hat{Z}_{(4)}| \det|M_1| = 0.$$

Let

$$\hat{Z} = \hat{Z}(\varepsilon) := \begin{bmatrix} \hat{Z}_{(1)}(\varepsilon) & \\ & \hat{Z}_{(4)}(\varepsilon) \end{bmatrix}$$

and

$$S = \left|\hat{Z}(\varepsilon)\right| \left|A\hat{Z}^{-1}(\varepsilon)\widetilde{Y}(\varepsilon)A^{\mathrm{T}}\right|. \tag{15}$$

It is clear that $\det|M| = 0$ is equivalent to $S = 0$.

By Binet-Cauchy Theorem [17] and noticing that $\hat{Z}$ is diagonal, we have

$$\begin{aligned} S &= \left|\hat{Z}\right| \sum_{j_1,\cdots,j_n} \left[A\hat{Z}^{-1}\right] \begin{pmatrix} 1 & \cdots & n \\ j_1 & \cdots & j_n \end{pmatrix} \left[\widetilde{Y}A^{\mathrm{T}}\right] \begin{pmatrix} j_1 & \cdots & j_n \\ 1 & \cdots & n \end{pmatrix} \\ &= \left|\hat{Z}\right| \sum_{\substack{j_1,\cdots,j_n \\ k_1,\cdots,k_n}} \left(\prod_{i=1}^{n} \hat{Z}_{j_i}^{-1}\right) A \begin{pmatrix} 1 & \cdots & n \\ j_1 & \cdots & j_n \end{pmatrix} \widetilde{Y} \begin{pmatrix} j_1 & \cdots & j_n \\ k_1 & \cdots & k_n \end{pmatrix} A^{\mathrm{T}} \begin{pmatrix} k_1 & \cdots & k_n \\ 1 & \cdots & n \end{pmatrix} \end{aligned} \tag{16}$$

where n is the number of nodes in the network (excluding the ground node) and the notation $A\begin{pmatrix} 1 & \cdots & n \\ j_1 & \cdots & j_n \end{pmatrix}$ denotes the major of matrix A formed by the rows $\{1, \cdots, n\}$ and the columns $\{j_1, \cdots, j_n\}$. By default the index set $\{j_1, \cdots, j_n\}$ for summation means the summation over all distinct indices satisfying $j_1 < j_2 < \cdots < j_n$.

We observe from (16) that a nonzero term in the summand results from the condition that all the three factors

$$A\begin{pmatrix} 1 & \cdots & n \\ j_1 & \cdots & j_n \end{pmatrix},$$
$$\widetilde{Y}\begin{pmatrix} j_1 & \cdots & j_n \\ k_1 & \cdots & k_n \end{pmatrix},$$
$$A^{\mathrm{T}}\begin{pmatrix} k_1 & \cdots & k_n \\ 1 & \cdots & n \end{pmatrix}$$

are nonzero. It is a well-known fact that a nonzero major of the reduced incidence matrix corresponds to a tree. Since A is a reduced incidence matrix, if the index sets $\{j_1, \cdots, j_n\}$ and $\{k_1, \cdots, k_n\}$ coincide, then the term is determined by a tree with its edges identified by the index set, while if the index sets are not identical, then the term is determined by a tree-pair. Note that in the case of a tree-pair, the row indices of $\widetilde{Y}$ associated with the term are the edge numbers of the left-tree (L-tree), while the column indices of $\widetilde{Y}$ associated with the term are the edge numbers of the right-tree (R-tree).

However, given any tree or tree-pair, the corresponding term can still be vanishing if the determinant of $\widetilde{Y}\begin{pmatrix} j_1 & \cdots & j_n \\ k_1 & \cdots & k_n \end{pmatrix}$ vanishes. To prevent from enumerating such trees or tree-pairs leading to vanishing terms, we shall do a careful analysis of the structure of the matrices $\widetilde{Y}$ and $\hat{Z}$. Such an analysis leads to a set of rules that only enumerate those trees and tree-pairs contributing to nonvanishing product terms in the expansion of (16). Since all such trees or tree-pairs are distinct, the fact of *cancellation-free* is self-evident.

The enumeration rules can now be derived from the expansion form in (16). It is important to note that the three factors, $|\hat{Z}|$, $\left(\prod_{i=1}^{n}\hat{Z}_{j_i}^{-1}\right)$ and the minors of $\widetilde{Y}$, all could involve ε. When a product term is formed, the limit $\varepsilon \to 0$ must be taken if any ε is involved. Thus we shall show that the rules automatically drop all vanishing terms. It is a simple matter to note that no term tends to infinity according to the form (16). Since all ε's introduced were for the purpose of substituting those diagonal zeros of the Z matrix, the determinant $|\hat{Z}|$ has the product of all ε's as its factor. Being a common factor, all ε's of $|\hat{Z}|$ should be cancelled by the ε's from the product of $\left(\prod_{i=1}^{n}\hat{Z}_{j_i}^{-1}\right)$ and a minor of $\widetilde{Y}$ to yield a nonvanishing term.

We proceed to prove Rule 1. In the case of tree, only the diagonal elements of $\widetilde{Y}$ are involved, which do not have any ε symbol according to the stamps. Since the $\hat{Z}_{j_i}$'s in the product $\left(\prod_{i=1}^{n}\hat{Z}_{j_i}^{-1}\right)$ are just part of the diagonals of $\hat{Z}$, all diagonal ε of $\hat{Z}$ should be included in every product of $\left(\prod_{i=1}^{n}\hat{Z}_{j_i}^{-1}\right)$, because one missing ε in the product will result in an extra ε in $|\hat{Z}|$, making the term vanishing. Also zero diagonal minors of $\widetilde{Y}$ lead to vanishing terms, hence no zero diagonal of $\widetilde{Y}$ should be selected. These two requirements pose a condition on a term tree; namely, all ε in the diagonal of $\hat{Z}$ must associate with a nonzero diagonal of $\widetilde{Y}$ at the same location. However, the stamp of a nullor does not meet this condition. Therefore, if at least one nullor exists in a circuit, no term tree should be enumerated. If no nullor is in presence, then all CC and VS edges (if any) meet the condition we just stated. Hence all CC and VS edges if present in the network must be included in all term trees.

Rule 2 for the product term given by a term tree is straightforward. A tree is only associated with the nonzero diagonals of $\widetilde{Y}$, which are $-Y_j$, $-Z_i^{-1}$, or -1, and the factor of $|\hat{Z}|$ in (16) after the cancellation of all ε's is just 1. Hence we have Rule 2. It suffices to make a note on the sign. Since all nonzero diagonal elements of $\widetilde{Y}$ are negative, the sign of all terms thus obtained is $(-1)^n$. But a common factor has no effect on the homogeneous equation $\det|M| = 0$, therefore it is safe to drop all the minus signs of the nonzeros diagonals of $\widetilde{Y}$.

It remains to prove the rules on the term tree-pairs. In the two trees of a tree-pair some edges are common and some edges are distinct but in pairs. All the common edges are associated with the nonzero diagonals of matrix $\widetilde{Y}$ and all the paired edges are associated with the nonzero off-diagonals of $\widetilde{Y}$. We first derive from the stamps in (6) and (7) that some edges (if present in the graph) *must* be included in all term tree-pairs.

If any nullors appear in the circuit, all the nullor edges must show up in all term tree-pairs pairwise. As a matter of fact, the nullor stamp has an ε on the Z diagonal, which implies (as we argued before for the tree case) that the NO (norator) edge must be included. In the meanwhile note that the corresponding row of $\widetilde{Y}$ has a diagonal 0 and an off-diagonal 1, which implies that only the NU-edge can be paired with the NO-edge. In other words, all nullator pairs must be included in all term tree-pairs.

The other edges *must be included* are CC and VS edges, but they are allowed to appear either in common or in pair. To see this, we consider the relevant stamps in four cases one by one.

For the VCVS stamp, there is an ε at the Z diagonal associated with the VS edge. Hence the VS edge must be on the L-tree. However, there are two nonzeros in the matrix $\widetilde{Y}$, -1 and $E_{i,j}$, at the same row with ε. If -1 is selected, it means that the VS edge is simply a common edge for both trees. If $E_{i,j}$ is selected, then it means that VS is paired with its dependent VC on the tree-pair.

For the CCCS stamp in (7), since the diagonal ε of the Z-stamp is associated with a diagonal -1 in the Y-stamp, CC must appear and can be a common edge. On the other hand, the diagonal 1 in the Z-stamp is associated with the off-diagonal $-\varepsilon^{-1}F_{i,j}$ of the Y-stamp, this means that, alternately, (CS,CC) can appear as a pair. In this case, CC cannot be a common edge anymore, because the ε^{-1} factor in the term $-\varepsilon^{-1}F_{i,j}$ must cancel an extra ε from $|\hat{Z}|$ if the diagonal ε in the Z-stamp is not used.

The situation with the CCVS stamp in (7) is analogous. According to the stamp, we have two mutually exclusive cases possible, CC and VS both are common edges, i.e. (CC,CC) and (VS, VS), or CC and VS as a pair, i.e. (VS, CC). The explanation is omitted.

The case of VCCS is somehow special, because both diagonals of the Z-stamp of VCCS are 1. This implies that the VC and CS edges in this case need not appear in all term tree. However, since the diagonals of the Y-stamp are all zero, whenever a VC or CS edge of a VCCS appears, it must be accompanied by its dependent edge. This is item *(iv)* stated in Rule 3.

Following the exposition above, Rule 4 for the product term determined by a term tree-pair is rather straightforward. To see the sign rule for the gains in Rule 5, it suffices to note that in the stamps only the VCVS gain $E_{i,j}$ and the CCVS gain $\varepsilon^{-1}H_{i,j}$ are plus-signed while all other nonzeros of $\widetilde{Y}$ are minus-signed. Hence the reason for adding minus signs to $E_{i,j}$ and $H_{i,j}$ is clear. Furthermore, according to the nullor stamp, a nullor pair always contributes a weight one to each term determined by a term tree-pair.

The term sign rule stated in Rule 6 follows directly from the Binet-Cauchy expansion in (16). However a few words are worthwhile for the column alignment stated in the rule. Recall that by numbering the controlling edges larger than the controlled edges, all dependent source gains appear in the upper triangular part of $\widetilde{Y}$. This means that the submatrix of $\widetilde{Y}$ corresponding to a tree-pair in the Binet-Cauchy expansion (16) is also upper triangular. Note that exchanging two columns of a minor of $\widetilde{Y}$ and the associated two rows of an A_R^{T} will not change the sign of a product term in the Binet-Cauchy expansion. Therefore, if an off-diagonal gain symbol in $\widetilde{Y}$ is exchanged to the diagonal position at the same row while the associated rows in A_R^{T} are exchanged accordingly, the columns of the two majors corresponding to the L-tree and R-tree are then aligned, as stated in Rule 6. Such column alignment of the matrices A_L and A_R also justifies the sign rule for gains in Rule 5, because after a gain symbol is permutated to the diagonal, it can be treated similarly to the other elements on the diagonal of $\widetilde{Y}$.

The proof of the rules is now complete.

IV. Algorithms for Term Enumeration

A parser can be used to parse a standard Spice netlist into a graph representation following the Graph Construction Rules stated in Section II. There are many different ways to spanning tree enumeration. Typical ways include: (i) to select edges sequentially and then check whether a tree is formed [18], and (ii) to form a spanning tree first and then obtain other trees by exchanging one edge on the tree with another edge not on the tree [19]. Minty's algorithm belongs to the former and can easily be modified to enumerate all trees and tree-pairs that obey the rules presented in Section II, from which all nonvanishing product terms are obtained. The key task of modified Minty's algorithm is to select edges that satisfy the rules and add them as tree edges meanwhile check whether a tree or a tree-pair has been formed. Tree-checking is implemented efficiently by edge coloring, record-keeping of connected components, and loop-checking [1]. The details are described below.

Let E and N be the numbers of edges and nodes, respectively. Suppose the nodes are numbered continuously 0, 1, 2, $\cdots, N-1$, with the ground numbered 0. The following data structures are used in the implementation.

1) An array of edge structures $edge[E]$, each containing information of an edge name, two end node numbers, a type, and a link to another edge in pair if a dependent edge.

2) Three working arrays: $compL[N]$, $compR[N]$, and $mark[E]$. The arrays $compL[N]$ and $compR[N]$ store colored components for checking whether a loop is formed in the L-tree or the R-tree. The array $mark[E]$ is used for marking those edges processed. Five marks (colors) are used: 1 for an unprocessed edge, 0 for a removed edge, -1 for a common edge (Y/Z/CC/VS) on a tree or tree-pair, -2 for a controlling edge (CC/VC) or a nullor (NU) on the R-tree, and -3 for a controlled edge (CS/VS) or a norator (NO) on the L-tree.

3) Three 2D stacks: $stackL[E][N]$, $stackR[E][N]$, and $stackE[E][E]$ for storing the component information of L-tree, R-tree, and the edge marks, respectively.

A nice property of Minty's algorithm is that the depth of the stacks never exceeds E, i.e. optimal in memory.

The basic idea of Minty's algorithm is binary decision: pick up an unprocessed edge in sequence and then either process it or delete it. When an edge is processed, its type is checked and appropriate actions are taken. In the original Minty's algorithm [18] all edges have the same type, hence are processed in the same way. In the current context we have different types of edges and they have to be processed in different ways according to the rules. The modified Minty's algorithm is listed below.

Modified Minty's Algorithm:

Step 1 Initialization: Mark all edges 1 (unprocessed) and set both arrays $compL[N]$ and $compR[N]$ to zero (no components yet). Sort the edge list so that all NU and NO edges precede all CC and VS edges, which in turn precede all the rest edges. Use the initial graph as the working graph.

Step 2 Select an unprocessed edge from the working graph and check its type.

```
Case 1: A Y or Z edge.
    Remove it and check the possibility of tree.
    If tree possible, push stacks.
    Mark this edge $-1$.
    Case 2: A CC edge.
        Mark it $-1$ (common edge), check loop.
        If no loop, check its pairing edge type.
            If VS, mark it $-1$ and check loop.
                If no loop, push stacks.
            If CS, remove it.
        Mark this edge $-2$ and its pairing edge $-3$.
    Case 3: A VS edge.
        Mark it $-1$ (common edge), check loop.
        If no loop, check its pairing edge type.
            If CC, mark it $-1$ and check loop.
                If no loop, push stacks.
            If VC, remove it.
        Mark this edge $-3$ and its pairing edge $-2$.
    Case 4: A VC edge of VCCS.
        Remove pair and check the possibility of tree.
        If tree possible, push stacks.
        Mark this edge $-2$ and its pairing edge $-3$.
    Case 5: A CS edge of VCCS.
        Remove the pair and check the possibility of tree.
        If tree possible, push stacks.
        Mark this edge $-3$ and its pairing edge $-2$.
    Case 6: A NU or NO edge.
        Mark the (NO,NU) pair $(-3, -2)$.
Step 3 Check loop.
    If yes, discard the graph and goto Step 6.
Step 4 Check tree or tree-pair.
    If yes, process the term and its sign. Goto Step 6.
Step 5 More edges unprocessed?
    If yes, goto Step 2.
Step 6 Stacks nonempty?
    If yes, pop the stacks and goto Step 2.
    Else, quit the enumeration.
```

Some explanations are in order. First, to check whether a potential tree is possible after an edge or a pair of edges is removed, there are several ways to do so. Currently we have implemented the following two:

One is to check the connectivity of the remaining graph; if disconnected, then no tree will be possible. The other is to count the number of the remaining edges; if the number is less than $(N-1)$, then no tree will be possible. Second, when pushing stacks, all information related to the working graph up to the moment has to be preserved. This includes the edge marks and the colored components of both L-tree and R-tree.

The correctness of this algorithm is justified by checking the order of operations with the rules stated in Section II. Before ending this section, we discuss the implementation of several functions used in the Modified Minty's Algorithm.

The loop-checking function is implemented by means of connected components. The details are similar to the Chan's modified algorithm described in [1] page 112, with slight modifications. Specifically, in an R-tree all connected edges marked -1 or -2 form a component, while in a L-tree all connected edges marked -1 or -3 form a component. When a newly processed edge has both end nodes from the same component, a loop is detected. The same technique can be used to check whether a graph is disconnected when some edges are removed. This is used for checking the availability of trees.

The tree-checking function is implemented by simply checking whether all nodes have been connected to one single component without a loop. In the case of tree-pair, we have to check whether both L-tree and R-tree are formed at the same time. This is easily done by checking the two component arrays $compL[N]$ and $compR[N]$.

The implementation of the term sign rule stated in Rule 6 is nontrivial. The sign problem has been addressed in many early research papers that deal with tree-pair methods, e.g. [20, 10, 4]. The procedure developed by Talbot in [10] uses the elementary tree transformation idea, which determines the sign by successive edge substitution. The rules developed in those early papers were targeted for manual manipulation, not immediately suited for computer implementation.

For completeness, we present here an algorithm that is easy and efficient for computer implementation. The algorithm essentially is based on Gaussian elimination, but taking great advantage of the simple structure of an incidence matrix, i.e. each column has only two nonzero entries, $+1$ and -1, making it possible to implement Gaussian elimination by a process of relabelling the edge nodes. Specifically, as we process the tree edges in sequence, we relabel the end nodes of the remaining edges depending on the edge just removed (or collapsed), meanwhile we keep the record of two permutation arrays that also contribute to the sign. This algorithm does not require an incidence matrix to be physically stored in the memory and the procedure is rather systematic. The details of this algorithm are given below.

Suppose every edge (n_1, n_2) is directed from node n_1 to n_2. Let $permL[N]$ and $permR[N]$ be two integer arrays that keep the permutation of the rows of the reduced incidence matrices corresponding to the L-tree and the R-tree, respectively. Let $eL[N-1]$ and $eR[N-1]$ be two edge arrays containing the edges of L-tree and R-tree, respectively. The following algorithm processes both L-tree and R-tree simultaneously. Note that this sign algorithm takes care of the gain sign as stated in Rule 5 in the mean time.

Term Sign Algorithm:

Step 0 Initialize: $sign := 1$ and $k := 1$.

Step 1 Get the kth edges of L-tree and R-tree, $eL[k]$ and $eR[k]$. If the two edges are paired and involve a VS, $sign := sign * (-1)$. Get the end nodes of both edges.

Step 2 If n_1 of the edge $eL[k]$ is ground, $sign := sign * (-1)$ and exchange n_1 and n_2 of edge $eL[k]$. Meanwhile if n_1 of the edge $eR[k]$ is ground, $sign := sign * (-1)$ and exchange n_1 and n_2 of edge $eR[k]$.

Step 3 Save n_1 of edge $eL[k]$ to $permL[k]$ and $n1$ of edge $eR[k]$ to $permR[k]$, respectively.

Step 4 Relabel all end node n_1 of the remaining edges in L-tree to n_2. Do the same to R-tree.

Step 5 If $k < N - 1$, goto Step 1.

Step 6 Finalize the sign:

$$sign := sign * \text{SignOf}(permL) * \text{SignOf}(permR).$$

In Step 5, the sign of a permutation is positive if the permutation is even, and negative if the permutation is odd. This can be evaluated by a linear algorithm.

Proof of the Term Sign Algorithm : The proof is based on Gaussian elimination. We shall focus on one tree, say, the L-tree. Let A_L be the reduced incidence matrix of the L-tree. The rows of A_L correspond to the nodes of the tree excluding the ground node, and the columns of A_L correspond to the edges of the tree. Note that each column of A_L belongs to one of the following three cases: (a) all zeros but a -1 and a 1, (b) all zeros but a 1, and (c) all zeros but a -1. Since we assume an edge (n_1, n_2) is directed from node n_1 to node n_2, the column corresponding to this edge in the matrix A_L has the nonzero 1 at row n_1 and the nonzero -1 at row n_2. If the edge is connected to the ground, then either $n_1 = 0$ or $n_2 = 0$ and the corresponding row is not in the matrix A_L.

Since the edges of a tree are scanned in sequence, if an edge is not connected to the ground, then both nodes n_1 and n_2 of the edge are nonzero, i.e. case (a) above. Then we can add row n_1 of matrix A_L to row n_2, eliminating -1 in the column corresponding to this edge. Now all other nonzeros ± 1 at the same row as n_1 are carried to row n_2 (because of no parallel edges in a tree). According to the Laplace expansion of a determinant, the row n_1 and the column marked by this edge can now be removed from the matrix A_L, reducing one order of the determinant. It is easy to see that carrying all the nonzeros from row n_1 to row n_2 is equivalent to relabelling node n_1 to node n_2 for all the unscanned edges in the L-tree. In other words, the edge (n_1, n_2) is collapsed by merging the two end nodes, and the merged node is relabelled to n_2. In implementation, this is done merely by relabelling all the remaining edges in the tree connected to n_1 with n_1 replaced by n_2. This process is called *relabelling.*

For case (b), i.e. the edge being processed is $(n_1, 0)$ with $n_1 \neq 0$, there is only a nonzero 1 at row n_1 in the column marked by this edge. Then row n_1 can be directly removed in the Laplace expansion. But all the rest edges connected to node n_1 have to be relabelled with n_1 replaced by 0. In effect, this is also equivalent to collapsing the edge and relabel the merged node by 0. The reason to relabel node n_1 to 0 is that the reduced tree after all the previous collapsing and relabelling has the same reduced incidence matrix as the remaining submatrix of A_L after all the previous row and column eliminations.

Case (c) is analogous to case (b) except that now there is only one nonzero -1 at the column marked by this edge. Hence the variable *sign* has to be updated by -1, i.e. $sign := sign * (-1)$. Since this edge is labelled $(0, n_2)$ and node n_2 is to be relabelled to 0, to be consistent with the other cases, the two nodes 0 and n_2 are swapped so that we can always replace the first node by the second node, as stated in Step 4 of the algorithm.

According to the definition of Laplace expansion of a determinant, the row permutation should be taken account of for the sign. This is done by Step 6. The sign change for an edge pair involving a VS is obvious according to Rule 5. ∎

V. Symbol Decision Diagram

The Modified Minty's Algorithm enumerates all cancellation-free product terms, from which a transfer function in symbolic form can be formed. For large networks, printing out all symbolic terms of a transfer function is usually not the goal of symbolic simulation. Rather, we would like to extract useful information from the available terms for analysis and optimization. Therefore, finding an efficient way to store those symbolic product terms in the computer memory is one of the key tasks in the construction of a symbolic simulator based on term enumeration.

There are many different ways to store all enumerated terms in memory. One may choose a *compact* output for spanning tree enumeration [21], but the time complexity is in the order of the number of terms. Because the number of terms grows exponentially in the circuit size, the storage scheme must consider manipulation efficiency in addition to compactness. Typical manipulations include the evaluation of a frequency response when numerical values of the symbols are given. If the terms are stored in a compact data structure, the numerical evaluation time could be reduced greatly, even in a time complexity linear to the size of a circuit. A good data structure suitable for this purpose is *Binary Decision Diagram* (BDD) [22]. BDD was originally used to represent binary terms in digital systems, and has been extended to represent subset systems [23]. Since the product terms in symbolic analysis can be viewed as subsets of a whole set containing all symbols, a decision diagram can be applied to symbolic analysis of analog circuits. Shi and Tan uses decision diagrams to store all cofactors of a nodal admittance matrix in its Laplace expansion [15]. In most practical applications, if a good ordering of the symbols is chosen, the size of a decision diagram can be made almost linearly proportional to the circuit size. The sharing of subterms in all product terms is the main contributor to the compactness of a decision diagram. Thus the decision diagram representation provides an efficient medium for storage and numerical manipulation in term-based symbolic analysis.

A decision diagram used for representing all signed product terms is called a *Symbol Decision Diagram* (SDD). We use an example to demonstrate the construction of a SDD. More details of the related concepts can be found in [23, 15]. Consider the circuit shown in Fig. 4. The following six signed terms are enumerated

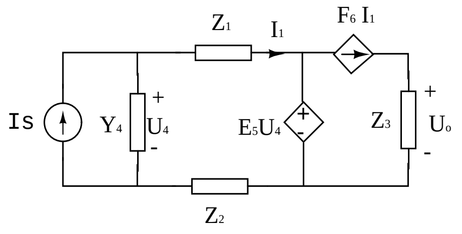


Fig. 4. A circuit example.

which sum up to zero.

$$X \cdot E_5 Z_2^{-1} Z_1^{-1} F_6 - X \cdot Z_2^{-1} Z_1^{-1} F_6 + Z_2^{-1} Z_3^{-1} Y_4$$
$$+ Z_2^{-1} Z_3^{-1} Z_1^{-1} + Z_3^{-1} Y_4 Z_1^{-1} - E_5 Z_2^{-1} Z_3^{-1} Z_1^{-1} = 0 \tag{17}$$

where X is the unknown. Note that all impedances are inverted. The transfer function from I_s to U_o is

$$T(s) = \frac{1}{X} =$$
$$-\frac{E_5 Z_2^{-1} Z_1^{-1} F_6 - Z_2^{-1} Z_1^{-1} F_6}{Z_2^{-1} Z_3^{-1} Y_4 + Z_2^{-1} Z_3^{-1} Z_1^{-1} + Z_3^{-1} Y_4 Z_1^{-1} - E_5 Z_2^{-1} Z_3^{-1} Z_1^{-1}}. \tag{18}$$

Note that symbolically all Z_i^{-1} can be treated as a symbol Z_i. Only in numerical evaluation must all values for symbol Z_i's be inverted according to the rules.

We treat the sign of each term as a symbol as well. Hence there are 9 symbols in total for this example. The way these six terms are stored in a SDD is explained in light of the decision diagram in Fig. 5. First we choose an order of the 9 symbols: $X > + > - > E_5 > Z_2 > Z_3 > Y_4 > Z_1 > F_6$. To facilitate numerical evaluation of frequency response, we have specifically indexed the three special symbols with the highest order in $X > + > - > \cdots$. The order of the rest of symbols are arbitrary, but it affects the size of a SDD. The optimal order is usually determined heuristically because it is an NP-hard problem. The symbols are indexed by integers with the index 9 for X, 8 for '+' and so on. The SDD nodes are identified by the indexes, which are shown beside each node in Fig. 5. Each node of the SDD branches two edges pointing to two descendant nodes, except the two nodes at the bottom marked 1 and 0 called *terminal nodes* in squares. Every nonterminal node is indexed larger than all its descendant nodes. The (solid) left edge is called a 1-edge, while the (dashed) right edge is called a 0-edge. A signed product term is represented by a path from the root node to the 1-terminal, called *1-path*, and all the symbols originating a 1-edge along the 1-path make up the term. One can easily check that the SDD in Fig. 5 represents the six product terms in (17). The number of nodes in the SDD determines the SDD size, denoted $|SDD|$.

The numerical value of each product term is the product of all symbols substituted by their (complex) numerical values (the numerical value for symbol Z_i is Z_i^{-1}). For numerical evaluation, the '+' symbol is assigned a value 1 and the '$-$' symbol a value -1. Each node in SDD can be interpreted as the representation of a group of subterms that form a subdiagram rooted at this node. Such *nested* subterms are used recursively to evaluate the algebraic sum of all product terms.

There are two groups of product terms in a SDD; one includes those terms containing the unknown symbol X which form the numerator of the transfer function in (18) with X excluded, and the other includes those

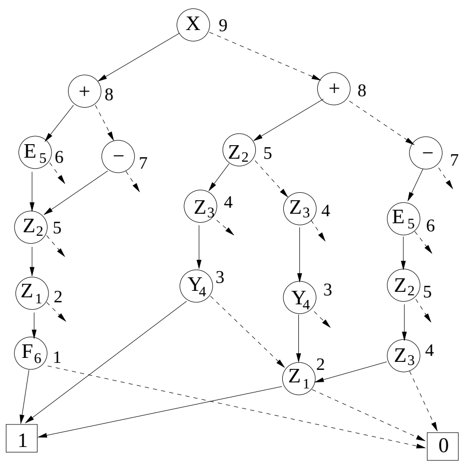


Fig. 5. A symbol decision diagram (SDD). The unterminated 0-edges are all connected to the 0-terminal. The Z_i nodes represent Z_i^{-1} in the numerical sense.

terms not involving the X symbol which form the denominator of the transfer function. From the structure of SDD we see that the numerator is evaluated from the '+' node pointed by the 1-edge of node X, while the denominator is from the value of the '+' node pointed by the 0-edge of the root node X. The transfer function evaluated at one frequency point is then the quotient of the two values pointed by the root node multiplied by -1.

Hashing and cache are commonly used in standard BDD implementation for better efficiency of manipulation [24]. Our prototype simulator also implemented the same techniques which greatly improves the efficiency for frequency response evaluation.

VI. Examples

We have implemented a symbolic simulator called CFDD (Cancellation-Free Decision Diagram) that integrats all the components discussed so far. The CFDD simulator reads in a standard netlist compatible to Spice3, creates an internal symbolic representation of the network function in a SDD, and prints out or plots the numerical frequency response based on the nominal element values provided in the netlist. Further analysis can be done by modifying the symbol values without repeating the enumeration process. The simulator was written in C++ and run on Intel Pentium 1.3GHz processor with 256 MB memory and 1G paging memory. To reduce the number of terms, all parallel Y/Z edges are lumped and treated as one symbol. In this section we report preliminary test results of the CFDD simulator on two circuit examples. Extensive experiments and full implementation of other functionalities, such as sensitivity analysis, approximate analysis, and hierarchical analysis, etc. will be reported in forthcoming papers.

The first circuit example shown in Fig. 6 is a bandpass filter with 13 ideal opamps, which is a standard benchmark circuit used by many authors [25, 2]. An ideal opamp is modeled by a nullor, i.e. a pair of nullator and norator. This example is used to test the enumeration rules related to nullors. Although this circuit has a relatively large number of nodes and edges, the number of expanded terms is small comparing to transistor circuits with a similar number of nodes and edges. It took the simulator only 0.8 seconds to

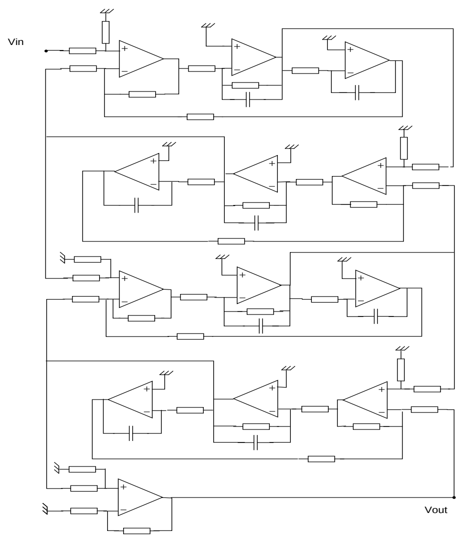


Fig. 6. A bandpass filter.

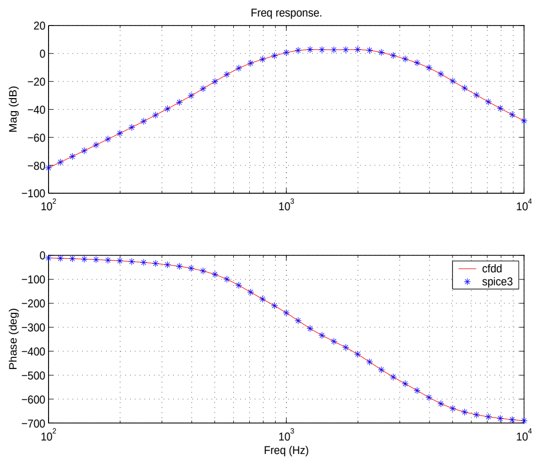


Fig. 7. Frequency response compared with Spice3.

construct the transfer function and evaluate the frequency response at 21 points. Some data of interest are listed in Table I. The correctness of the frequency response was verified by Spice3 (see Fig. 7). For Spice simulation, each opamp was modeled as a VCVS with a gain of 10^{10}.

The second circuit example shown in Fig. 8 is a gain-stage used in μA741 op-amp with 9 transistors. After converted to a graph, it has 54 edges and 9 nodes. Edge-lumping gives rise to 37 edges and 30 symbols. It took the simulator 10.2 seconds to enumerate 44,112 terms and evaluate the frequency response at 21 points. Other data related to this example are again summarized in Table I. The frequency response of this example obtained from CFDD was also checked by Spice3 and is plotted in Figure 9.

VII. Concluding remarks

A new set of topological rules has been developed in this paper. These rules can be used for deriving symbolic network functions directly from a network, without converting any elements. These rules apply to

TABLE I
Performance of the CFDD simulator

Circuit	bandpsss	gain-stage
Transistor	–	9
Edge	72	54
Edge (lumped)	68	37
Node	33	9
Symbol	43	30
Term	1,770	44,112
$\lvert SDD \rvert$	493	1,514
Freq Points	21	21
Time	0.8 sec	10.2 sec
Memory	165 MB	216 MB

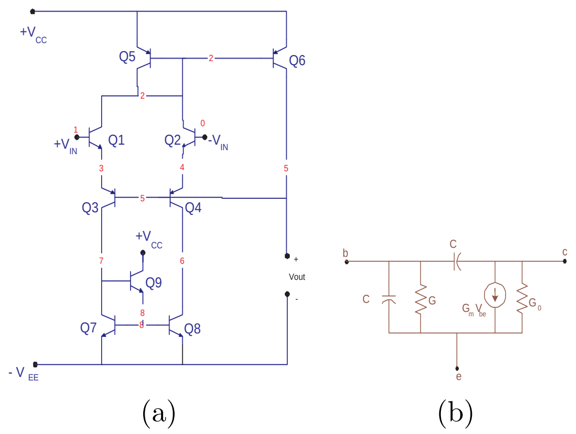


Fig. 8. (a) A gain stage. (b) BJT small signal model.

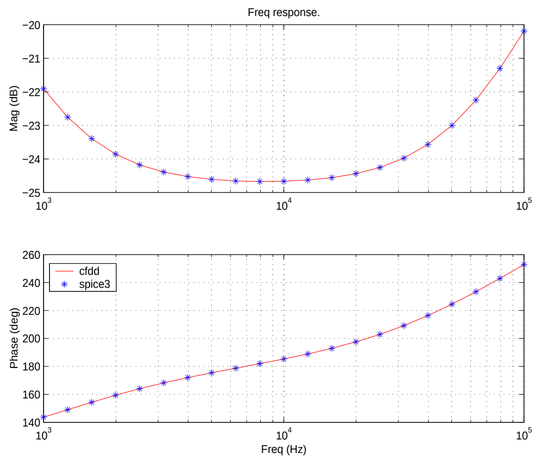


Fig. 9. Frequency response compared with Spice3.

all dependent sources, nullors, and other common circuit elements. The terms enumerated according to the rules do not contain any vanishing terms and are free of cancellation. We have provided an algebraic proof that shows the correctness of the enumeration rules and implemented the rules in a symbolic simulator. The correctness of the rules are further tested by circuit examples. Furthermore, we have demonstrated that a decision diagram can be used as an efficient data structure for storage and numerical manipulation.

It was once believed that tree-enumeration methods had difficulties in handling all types of controlled sources [2, 26]. This paper has shown that this is actually not true. However, as commonly recognized in the literature, direct application of topological methods permits the analysis of transistor networks with about 10 to 20 nodes [27]. As the circuit size becomes larger, application of topological methods for transistor networks is much more time consuming than other numerical methods or methods based on nodal formulation. Our experiment also has confirmed this observation. The limitation is mainly due to the astronomically large number of terms in topological approaches to determination of a network function [17]. In general, an active filter network with a number of ideal opamps has a relatively smaller number of terms because of the constraints posed by the ideal opamps, while for a transistor network with a comparable size the number of terms forming a transfer function could be many orders of magnitude larger. For this reason, it is not feasible to directly apply the enumeration algorithm developed in this paper to large transistor networks such as a μa741 opamp in a fully expanded form. However, it is possible to overcome this obstacle by applying other methods proposed in the literature for large scale networks.

References

[1] P. Lin, *Symbolic Network Analysis.* New York: Elsevier, 1991.

[2] G. Gielen, P. Wambacq, and W. Sansen, "Symbolic analysis methods and applications for analog circuits: A tutorial overview," *Proceedings of the IEEE*, vol. 82, pp. 287–303, Feburary 1994.

[3] W. Mayeda and S. Seshu, "Topological formulas for network functions." Engineering Experimentation Station Bullitin 446, University of Illinois, Urbana, 1959.

[4] W. Mayeda, *Graph Theory.* New York: Wiley-Interscience, 1972.

[5] Q. Yu and C. Sechen, "A unified approach to the approximate symbolic analysis of large analog integrated circuits," *IEEE Trans. on Circuits and Systems – I: Fundamental Theory and Applications*, vol. 43, no. 8, pp. 656–669, 1996.

[6] C. Coates, "General topological formulas for linear network functions," *IRE Trans. on Circuit Theory*, vol. CT-5, no. 1, pp. 42–54, Mar 1958.

[7] D. Brown, "New topological formulas for linear networks," *IEEE Trans. Circuit Theory*, vol. CT-12, pp. 358–365, Sept. 1965.

[8] W. Chen, "Topological analysis for active networks," *IEEE Trans. on Circuit Theory*, vol. CT-12, pp. 85–91, 1965.

[9] S.-D. Shieu and S.-P. Chan, "Topological formulation of symbolic network functions and sensitivity analysis of active networks," *IEEE Trans. on Circuits and Systems*, vol. CAS-21, no. 1, pp. 39–45, 1974.

[10] A. Talbot, "Topological analysis of general linear networks," *IEEE Trans. on Circuit Theory*, vol. CT-12, no. 2, pp. 170–180, June 1965.

[11] J. Numata and M. Iri, "Mixed-type topological formulas for general linear networks," *IEEE Trans. on Circuit Theory*, vol. CT-20, no. 5, pp. 488–494, 1973.

[12] P. Wambacq, G. Gielen, and W. Sansen, "A cancellation-free algorithm for the symbolic simulation of large analog circuits," in *Proc. Int'l Symposium on Circuits and Systems*, pp. 1157–1160, 1992.

[13] Z. Yin, "Symbolic network analysis with the valid trees and the valid tree-pairs," in *IEEE Int'l Symposium on Circuit and Systems*, (Sydney, Australia), pp. 335–338, 2001.

[14] X. Li and Z. Yin, "The algorithm and program scheme to find out all valid trees and valid tree-pairs," in *Proc. 5th Int'l Conference on ASIC*, (Beijing, China), pp. 298–301, 2003.

[15] C.-J. Shi and X.-D. Tan, "Canonical symbolic analysis of large analog circuits with determinant decision diagrams," *IEEE Trans. on Computer-Aided Design*, vol. 19, no. 1, pp. 1–18, January 2000.

[16] M. Hassoun and P. Lin, "A hierarchical network approach to symbolic analysis of large-scale networks," *IEEE Trans. on Circuits and Systems – I: Fundamental Theory and Applications*, vol. 42, no. 2, pp. 201–211, 1995.

[17] W. Chen, *Applied Graph Theory – Graphs and Electrical Networks.* Amsterdam: North-Holland, 1976.

[18] G. Minty, "A simple algorithm for listing all the trees of a graph," *IEEE Trans. on Circuit Theory*, vol. CT-12, p. 120, 1965.

[19] H. Gabow and E. Myers, "Finding all spanning trees of directed and undirected graphs," *SIAM J. Computing*, vol. 7, pp. 280–287, 1978.

[20] A. Ali, "On the sign of a tree pair," *IEEE Trans. on Circuit Theory*, vol. CT-11, no. 2, pp. 294–296, June 1964.

[21] A. Shioura, A. Tamura, and T. Uno, "An optimal algorithm for scanning all spanning trees of undirected graphs," *SIAM J. Computing*, vol. 26, no. 3, pp. 678–692, 1997.

[22] R. Bryant, "Graph-based algorithms for boolean function manipulation," *IEEE Trans. on Computers*, vol. C-35, no. 8, pp. 677–691, 1986.

[23] S. Minato, "Zero-suppressed BDD's for set manipulation in combinatorial problems," in *Proc. 30th IEEE/ACM Design Automation Conf.*, (Dallas, TX), pp. 272–277, 1993.

[24] K. Brace, R. Rudell, and R. Bryant, "Efficient implementation of a BDD package," in *Proc. 27th IEEE/ACM Design Automation Conference*, pp. 40–45, June 1990.

[25] J. Starzyk and A. Konczykowska, “Flowgraph analysis of large electronic networks,” *IEEE Trans. on Circuits and Systems*, vol. CAS-33, no. 3, pp. 302–315, 1986.

[26] M. Sharif-Bakhtiar and M. A. Ahmad, “Symbolic analysis of electronic circuits based on a tree enumeration technique,” *Proceedings IEE, pt. G*, vol. 140, pp. 68–74, Feb. 1993.

[27] L. Chua and P. Lin, *Computer Aided Analysis of Electronic Circuits – Algorithms and Computational Techniques.* Englewood Cliffs, NJ: Prentice-Hall, 1975.

A New Topological Approach to Symbolic Network Analysis

Guoyong Shi and C.-J. Richard Shi

Department of Electrical Engineering
University of Washington
Seattle, WA 98195, U.S.A.
E-mail: {gshi,cjshi}@ee.washington.edu

Manuscript received _________________________.

The authors are with the Department of Electrical Engineering, University of Washington, Seattle, WA 98195 USA (e-mail: gshi@ee.washington.edu; cjshi@ee.washington.edu).

This work was supported by DARPA NeoCAD Program under Grant No. N66001-01-8920.

List of Figures

Symbolic Model Order Reduction*

Bo P. Hu, Guoyong Shi, and C.-J. Richard Shi
Mixed-Signal CAD Research Laboratory
Department of Electrical Engineering
University of Washington, Seattle, WA 98195
{hubo, gshi, cjshi}@ee.washington.edu

Abstract – Symbolic Model Order Reduction (SMOR) is the problem of reducing a large circuit that contains symbolic circuit parameters to smaller low order models at its ports. Several methods, including symbol isolation, single frequency point reduction, and multiple frequency point reduction, are described and compared. Test circuits with simulation results are presented to demonstrate the accuracy and efficiency of SMOR.

I. Introduction

Model Order Reduction (MOR) is an efficient technique for fast and accurate simulation of RLC circuits. Various types of algorithm such as AWE [1], PVL [3], and PRIMA [4] have been developed to numerically reduce a large RLC network to smaller models, which can be used to replace the original large network in the subsequent simulation.

The drawback of those numeric MOR algorithms is the lack of flexibility. Whenever some element or parameter values in the large circuit change, the reduction has to be repeated.

Progress has been made to construct parameterized reduced model for interconnect [7], and to include variation analysis in RLC interconnect modeling [9]. Each method is efficient for some special cases.

In this paper, we present Symbolic Model Order Reduction (SMOR) as an attempt to overcome the limit of numeric model order reduction methods; several approaches are developed to handle symbolic elements inside a large circuit. Ideally, with SMOR, we will obtain a small model with symbols inside, and those symbols represent circuit elements (such as R, L, or C) or/and design parameters of interest (such as width, length of interconnect). The advantage of SMOR is its flexibility: when those symbolic elements or parameters change values, we can quickly update the symbolically reduced model, without performing the reduction again.

In this paper, first the PRIMA algorithm, on which SMOR is build, is briefly reviewed, followed by symbol isolation method, single frequency point reduction and multiple frequency point reduction method. Examples are given after the description of each method. Finally, the difficulties of SMOR are discussed.

* This research was partly supported by DARPA NeoCAD Program under Grant No. 66001-01-1-8920 and in part by SRC under Contract No. 2001-TJ-921.

II. Review on PRIMA

Let us first examine the PRIMA algorithm briefly. Consider a SISO system that can be described by the following equation:

$$C\frac{dx}{dt} = -Gx + bu \qquad (1)$$
$$y = \ell x$$

where u is the external stimulus to the system, b is the input vector, ℓ is the output vector, and y is the output of the model under the stimulus u. C and G are the system matrices which describe the dynamic behavior.

The key issue in model order reduction is to find a transformation matrix V. There are many ways to compute V, and in PRIMA it is given by:

$$V = K_q\left(G^{-1}C, G^{-1}b\right) \qquad (2)$$

The notation $K_q(A,b)$ denotes the Krylov subspace spanned by the vectors $[b, Ab, \ldots, A^{q-1}b]$. The original system can be reduced by congruence transformation [2] as follows:

$$C_r = V^{\mathbf{T}}CV,\ \ G_r = V^{\mathbf{T}}GV, b_r = V^{\mathbf{T}}b,\ \ell_r = \ell V \qquad (3)$$

And after the transformation, we have a smaller model, which is described as follows:

$$C_r\frac{dz}{dt} = -G_r z + b_r\mathrm{u} \qquad (4)$$
$$y = \ell_r z$$

The sizes of the system matrices C_r and G_r are much smaller than those of C and G. This reduced model can be used to achieve faster simulation speed, and in the same time reserve the required accuracy of the original system at its input and output ports [2] [4].

III. SMOR by Symbol Isolation

In the simplest but quite frequently happening scenario, the sweeping analysis of some circuit elements is desired, and it will be convenient to construct a compact model with those *symbolized* elements retained. The symbol isolation method is for such a purpose.

The basic idea is to isolate and replace each element of interest with a symbol. For each symbol, we model its interaction with the circuit as a port. For a typical two terminal element, its terminal voltage (or current flowing through it) would be added as the input stimulus to the circuit, and the current flowing through (or voltage across) the element would be the output under such a stimulus. After adding such extra inputs and outputs to the circuit, we carry out reduction on the circuit (which is purely numeric after isolating those symbolic elements from it), and then combine those symbolic elements to construct a symbolic model. The symbolic model can be reused whenever the value of any symbolic element changes.

The isolation method is discussed for the case of R, L or C element separately. In the case of multiple symbolic elements, there will be an extra step to combine them together into one compact symbolic model.

For simplicity, we will consider the SISO dynamic system as discussed in Section II. In the first case, we consider that a capacitor in this SISO system between nodes *i* and *j* is going to be isolated and symbolized. As illustrated in the following diagram, first, we isolate the capacitor and add one port to the large network, then reduce the large network, and finally put the capacitor back to make a compact symbolic model.

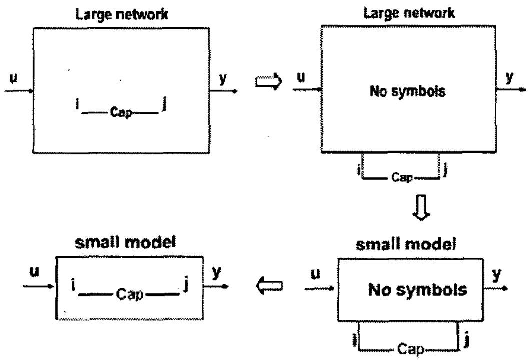


Fig. 1. Flow graph of isolation method when a capacitor is treated as a symbolic element.

Let $\hat{c}$ be the capacitance of the isolated capacitor, which is treated as a symbol. Let $\hat{I}$ and $\hat{v}$ be the current through (from *i* to *j*) and voltage across the capacitor respectively. We can describe this new system as follows:

$$\begin{bmatrix} \mathbf{C} & 0 \\ 0 & 0 \end{bmatrix} \frac{d}{dt} \begin{bmatrix} x \\ \hat{I} \end{bmatrix} = -\begin{bmatrix} G & E_{ij} \\ -E_{ij}^T & 0 \end{bmatrix} \begin{bmatrix} x \\ \hat{I} \end{bmatrix} + \begin{bmatrix} 0 \\ -1 \end{bmatrix} \hat{v} + \begin{bmatrix} b \\ 0 \end{bmatrix} u \quad (5)$$

$$\hat{c}\frac{d\hat{v}}{dt} = \hat{I}$$

$$\begin{bmatrix} y \\ \hat{I} \end{bmatrix} = \begin{bmatrix} \ell & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ \hat{I} \end{bmatrix}$$

where

$$E_{ij} = [\dots 1 \dots -1 \dots]^T$$

is a column vector containing all zeros but 1 and –1 at the locations corresponding to v_i and v_j in x, respectively.

After the isolation of the capacitor, the original SISO system becomes a MIMO system (there are two inputs and two outputs). If we apply the congruence transformation [2], we have

$$\begin{bmatrix} x \\ \hat{I} \end{bmatrix} = Vz \qquad \text{(V is the transformation matrix)}$$

$$b_v = V^T \begin{bmatrix} 0 \\ -1 \end{bmatrix}, \qquad b_r = V^T \begin{bmatrix} b \\ 0 \end{bmatrix}$$

$$\ell_r = [\ell \quad 0]V, \qquad \ell_v = [0 \ 1]V$$

$$C_r = V^T \begin{bmatrix} C & 0 \\ 0 & 0 \end{bmatrix} V, \qquad G_r = V^T \begin{bmatrix} G & E_{ij} \\ -E_{ij}^T & 0 \end{bmatrix} V$$

Then the MIMO system is reduced to:

$$C_r \frac{d\mathbf{z}}{dt} = -G_r\mathbf{z} + b_r u + b_v \hat{\mathrm{u}} \qquad (6)$$

$$\begin{bmatrix} y \\ \hat{I} \end{bmatrix} = \begin{bmatrix} \ell_r \\ \ell_v \end{bmatrix} z$$

$$\hat{c}\frac{d\hat{v}}{dt} = \hat{I}$$

If we introduce $\hat{v}$ as a state variable, the MIMO system becomes an SISO system again with $\hat{c}$ as a symbol.

$$\begin{bmatrix} C_r & 0 \\ 0 & \hat{c} \end{bmatrix} \frac{d}{dt} \begin{bmatrix} z \\ \hat{v} \end{bmatrix} = -\begin{bmatrix} G_r & -b_v \\ -\ell_v & 0 \end{bmatrix} \begin{bmatrix} z \\ \hat{v} \end{bmatrix} + b_r u \qquad (7)$$

$$y = [\ell_r \quad 0] \begin{bmatrix} z \\ \hat{v} \end{bmatrix}$$

After the combination step, we obtain a compact symbolic model, which is an SISO system as the original circuit, but has the extra flexibility to quickly update the symbolic capacitance.

For the resistor case, we follow the same procedure as for capacitor, and the system could be reduced to:

$$\begin{bmatrix} C_r & 0 \\ 0 & 0 \end{bmatrix} \frac{d}{dt} \begin{bmatrix} z \\ \hat{v} \end{bmatrix} = -\begin{bmatrix} G & -b_v \\ -\ell_v & 1/\hat{r} \end{bmatrix} \begin{bmatrix} z \\ \hat{v} \end{bmatrix} + b_r u \qquad (8)$$

$$y = [\ell_r \quad 0] \begin{bmatrix} z \\ \hat{v} \end{bmatrix}$$

For inductor, we introduce $\hat{I}$ as additional variable and the system could be reduced to:

$$\begin{bmatrix} C_r & 0 \\ 0 & \hat{L} \end{bmatrix} \frac{d}{dt} \begin{bmatrix} z \\ \hat{I} \end{bmatrix} = - \begin{bmatrix} G_r & -b_v \\ -\ell_r & 0 \end{bmatrix} \begin{bmatrix} z \\ \hat{I} \end{bmatrix} + b_r u \quad (9)$$

$$y = \begin{bmatrix} \ell_r & 0 \end{bmatrix} \begin{bmatrix} \mathbf{z} \\ \hat{I} \end{bmatrix}$$

In the case of multiple symbolic elements, we first need to identify those symbolic elements as ports in the circuit, introduce auxiliary variables as needed, reduce the remaining numerical model, then combine the symbolic ports with the numerically reduced model. The main processing overhead involved in this process is in the isolation and combination part.

We use the circuit shown in Fig. 2 to illustrate the effectiveness of the symbol isolation method. The circuit is an RLC ladder network with 301 elements (101 nodes). The size of the system matrices describing the original network is 103 by 103, and the size of the symbolic model is 12 by 12. The resistor, capacitor, or inductor placed between nodes 11 and 12 is treated as a symbolic element.

Simulation results are shown in Figs. 3 – 5. The results demonstrate that the model is accurate compared to the full order system, and has the flexibility to handle symbolic elements.

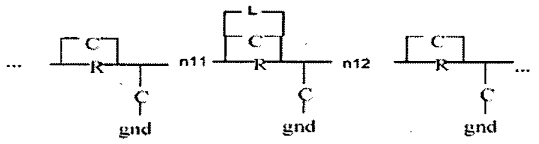


Fig. 2. A ladder circuit (301 elements).

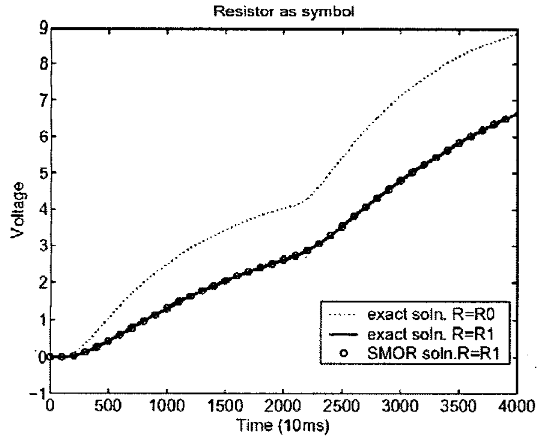


Fig. 3. The resistor between node 11 and node12 is a symbolic element. The time domain response to an external voltage stimulus is compared to the result given by the unreduced system, at both the nominal value and changed value of the resistor. The reduced symbolic model size is 12; the original network's size is 103.

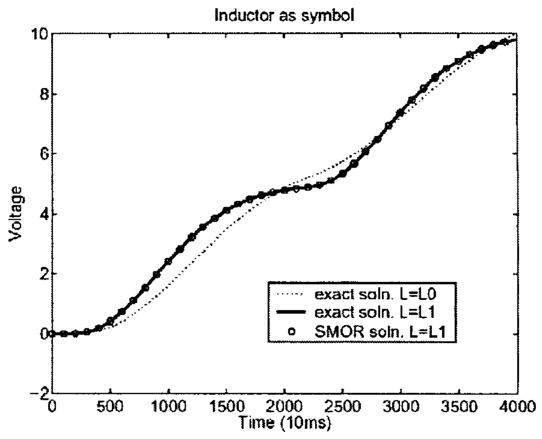


Fig. 4. Inductor between node 11 and node 12 is the symbol.

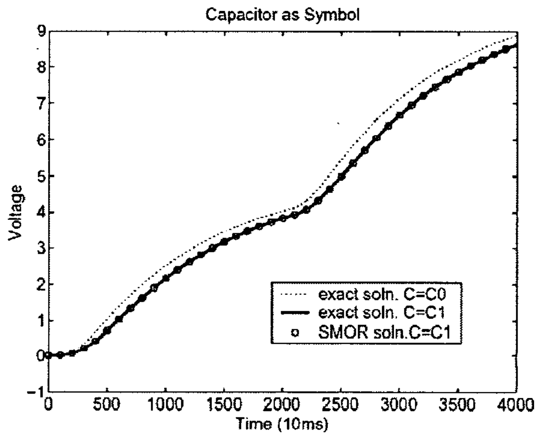


Fig. 5. Capacitor between node 11 and node 12 is the symbol.

In summary, the isolation method described above is useful in fast sweep analysis on some variables of a large circuit. However, this method is only applicable to a few symbolic elements. New approaches become necessary for more general problems.

IV. SMOR at Single Frequency Point

In the general case, it is not practical to isolate symbols one by one as in the previous example. Further the number of ports can grow rapidly, and thus decreases the value of model order reduction. In this section we propose a single frequency point based method for symbolic model order reduction.

Let $X(s) = T(s)U(s)$, where $T(s) = (Cs + G)^{-1}b$. The frequency point method arises from the expansion of $T(s)$, i.e.

$$T(s) = (Cs + G)^{-1}b$$
$$= [C(s-\sigma) + (C\sigma + G)]^{-1}b$$
$$= \sum_{i=0}^{\infty}[-(C\sigma + G)^{-1}C]^{i}(C\sigma + G)^{-1}b(s-\sigma)^{i}$$

σ is called a frequency point. In this section we consider $\sigma = 0$. In the next section, we consider multiple σ's, so called multiple frequency points.

We will restrict our discussion to the SISO network that can be described by the equation (1), where C and G contain symbols. The single frequency point method constructs transformation matrix V, from the Krylov subspace $K_q\left(G^{-1}C, G^{-1}b\right)$, then performs congruence transformation symbolically. The following algorithm constructs a symbolic transformation matrix V:

Algorithm 1

```
1). Symbolically inverse $G$ (expensive operation)
        $G^{-1}$ = inverse( $G$ )
2). Perform matrix-vector multiplication
        $v_1 = G^{-1}b$
3). For $k=2$ to $q$ ($q$ is the size of reduced model)
        $v_k = G^{-1}Cv_{k-1}$
4). For $k=1$ to $q$, construct transformation matrix $V$
        $V(:,k) = v_k$
```

V is an N by q matrix, N is the original size of the network, and q is the size of reduced model.

After V is constructed, the second step is to perform congruence transformation as described in (3), except that the matrix multiplication involves symbolic computation.

From our implementation with Maple [10], the two steps described above are quite complex, and the complexity grows exponentially with the sizes of N, q and the number of symbols inside the network to be reduced.

Note that in Algorithm 1, V is generated without orthonormalization. This reduces the symbolic computation complexity, but causes numerical instability problem. Since V is not orthonormalized, it could be extremely ill conditioned once every symbol is substituted by a numerical value. If we use such a V to do the congruence transformation, the reduced model may also become ill conditioned and likely useless.

To ease the condition number problem, we introduce a pseudo orthonormalization method.

The pseudo orthonormalization method is based on the assumption that the value of each symbol in matrices C and G will only change slightly from its nominal value. If this assumption holds, the transformation matrix V will not change much from its nominal value. The idea of pseudo orthonormalization method is to construct another numerical transformation matrix V_n (labeled as V_n to distinguish from V, which is a symbolic matrix; and if every symbol inside V takes its nominal value, then V_n equals V) along with the construction of the symbolic transformation matrix V; and use V_n as a tool to reduce the condition number of V.

The algorithm is described in pseudo code as follows:

(Notation: *vn(i)* is the *i*th column vector of the numerical transformation matrix V_n, and *v(i)* is the *i*th column vector of the symbolic transformation matrix V.)

Algorithm 2.

```
1). Generate vn(1), normalize vn(1) by coefficient $c11$
        $vn(1) \Leftarrow vn(1) \times c11$
    Generate v(1), and pseudo normalize v(1) with $c11$
        $v(1) \Leftarrow v(1) \times c11$
2). Generate vn(2), orthonormalize it with respect to vn(1)
        $vn(2) \Leftarrow vn(2) \times c22 + vn(1) \times c21$
    Generate v(2), pseudo orthonormalize it with respect to v(1),
    using c22 and c21
        $v(2) \Leftarrow v(2) \times c22 + v(1) \times c21$
3). Continue until we get all vectors pseudo orthonormalized.
```

However, during the implementation we observed that the coefficients *cii* are very sensitive to the variation of each symbol, e.g. even when a symbol just changes very little (far less than 1%) from its nominal value, with every other symbol remaining unchanged at all, the corresponding *cii* will change a lot and cause the pseudo orthonormalization fail to produce a better conditioned transformation matrix V for most of the cases.

As a result, the reduced model must have low order (about 10) by using the single frequency point method for symbolic model order reduction. For some systems, low order models are sufficiently accurate, especially in overly damped systems, where oscillation is weak. However, in general the low order approximation is not sufficient, especially when the inductance is in presence.

Figure 7 shows an example of the quick increase of the condition number of matrix V . The comparison is made between the Algorithm 1 (no orthonormalization) and Algorithm 2 (pseudo orthonormalization method). The example is based on the circuit presented in Section III, and the resistor between nodes 10 and 11 is changed from its nominal value by 1%.

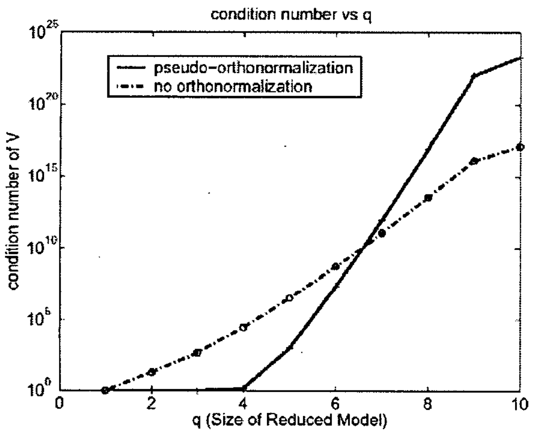


Fig 7. Condition number of V versus the size of reduced model q. It shows that the pseudo orthonormalization method is effective only when q is small.

V. SMOR by Multiple Frequency Point Mapping

To improve accuracy without greatly increasing the condition number of transformation matrix V , in this section we propose a multiple frequency point based reduction method. Using multiple frequency points (imaginary and/or real) in numerical model order reduction often generates more compact and accurate models [5] [6] [3], and here we use it for symbolic model order reduction.

The basic idea of multiple frequency point based method is to use the Krylov subspace:

$$K_q\left((C\sigma + G)^{-1}C, (C\sigma + G)^{-1}b\right)$$

For each choice of σ , we generate a few vectors in the Krylov subspace, and put them together to form the transformation matrix V. Specifically, we have the following algorithm.

Algorithm 3.

```
1). Symbolically inverse $(G+\sigma C)$
        $(G+\sigma C)^{-1}$ = inverse$(G+\sigma C)$
2). Perform matrix-vector multiplication
        $v_1 = (G+\sigma C)^{-1}b$
3). For $k=2$ to $q$ ($q$ is the size of reduced model)
        $v_k = (G+\sigma C)^{-1}Cv_{k-1}$
4). For $k=1$ to $q$, construct transformation matrix $V_\sigma$
        $V_\sigma(:,k) = v_k$
```

The procedure is similar to that described in Section IV, except that we replace G with $(G+\sigma C)$. The transformation matrix is denoted as V_σ correspondingly.

For the general case, when we expand the system at multiple frequency points $\{\sigma_1, \sigma_2, \cdots, \sigma_n\}$ the final transformation matrix V is formed by grouping each subspace together as follows:

$$V = colsp\{V_{\sigma_1}, V_{\sigma_2}, \ldots, V_{\sigma_n}\} \qquad (10)$$

If we use the above V to transform the original system using Equation (3), the resulting model will be a good approximation to the original system in a broader frequency range than single frequency point reduction.

It is straightforward to construct V if all the frequency points chosen are real; if σ_i is complex frequency point, an approach to obtaining a real V_{σ_i} was developed in [6][8]. Putting in a simple way, if one frequency point σ_i , is complex, the corresponding V_{σ_i} will be the combination of the real part and the imaginary part, rather than a complex subspace.

The important problem in multiple frequency point reduction is the selection of frequency points set $\{\sigma_1, \sigma_2, \cdots, \sigma_n\}$. Two factors are important in this selection. First, the frequency points should cover the interested frequency range to satisfy certain accuracy requirement; second, the frequency points should be carefully chosen to minimize the condition number of the union transformation matrix V (which is the union of the transformation matrix at each frequency point).

One approach in numerical multiple frequency point reduction is to choose evenly spaced frequency points [6], and use orthonormalization procedure to make sure the union transformation matrix V is well conditioned. However, in symbolic model order reduction, this approach is not feasible, which means we have to rely on the careful selection of the set of frequency points to lower the condition number of V .

To minimize the condition number of union transformation matrix V, the set of frequency points $\{\sigma_1, \sigma_2, \cdots, \sigma_n\}$should be well separated from each other, so that their corresponding transformation matrices do not overlap. If they overlap, the union transformation matrix V will have many vectors that are almost dependent on each other, and become ill conditioned.

Our first selection scheme is a very simple one: First, choose 0 as the first frequency point; then choose $\sigma_{\max}$ as the highest frequency point; and then choose σ_{mid} as the middle frequency point, based on the estimation of the dominant pole of the original circuit. Using the 3-point scheme, the resulting model would often be fairly good.

The second method is more computationally expensive, but compared with the saving of SMOR, such a cost is still justifiable. This selection scheme is based on the evaluation of the frequency domain behavior of the original system at each symbol's nominal value, using numerically reduced model by any reduction method. First, we perform numerical reduction (PRIMA, PVL etc) on the original system, and quickly evaluate the frequency response on the reduced model, then choose frequency points based on this evaluation.

One fundamental assumption in symbolic model order reduction is that the symbol values will not change drastically from their nominal values, hence the set of frequency points should still work, even if the symbols take different values from their nominal values.

There is a tradeoff between the number of frequency points and the condition number of the transformation matrix V. More frequency points will produce better accuracy in the reduced system, at the cost of high condition number of V. The general requirement for the condition number of V is by the order of 10^{10}, beyond that, the possibility of numeric breakdown will be very high. And in the practice of SMOR, we choose as many frequency points as possible under the constraint of the condition number of V.

To test the idea of multiple frequency point reduction, we consider the circuit shown below. This circuit is composed of 300 similar blocks of RLC elements, and the value of each element are different from block to block, however within a certain range (*R: 10-100 Ohm, L: 1-10 nH, C: 1-10 pF*).

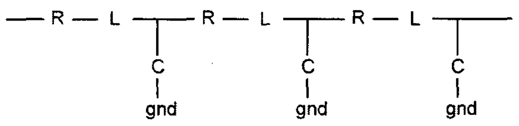
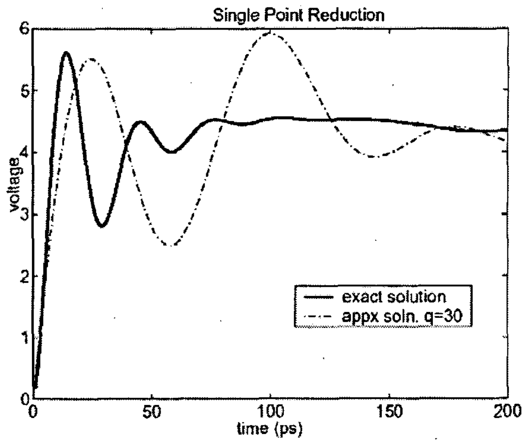
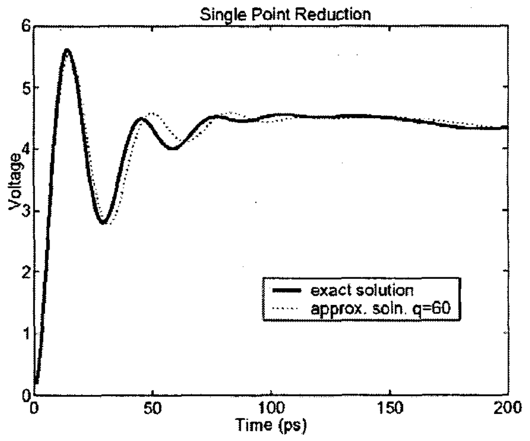


Fig. 6. The diagram of the test circuit (3 out of 300 blocks are plotted here)

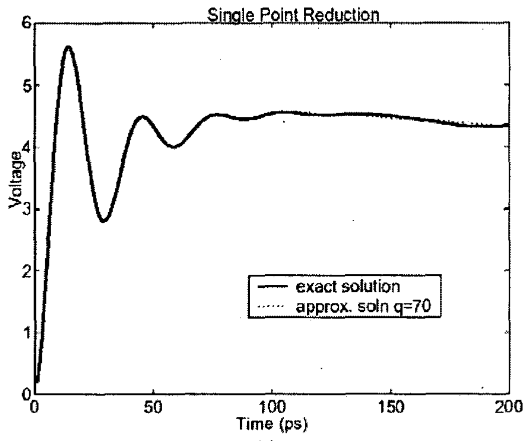
First, the efficiency of multiple frequency point reduction is demonstrated compared with single point reduction. Here the stimulus is a step voltage input at the left most node (node 1); the output is the 3rd node's voltage. The size of the original network is 902, and the size of the reduced model is increased from q=30 up to q=70 for the PRIMA algorithm; and for the multiple frequency point symbolic model, the size is just 9.



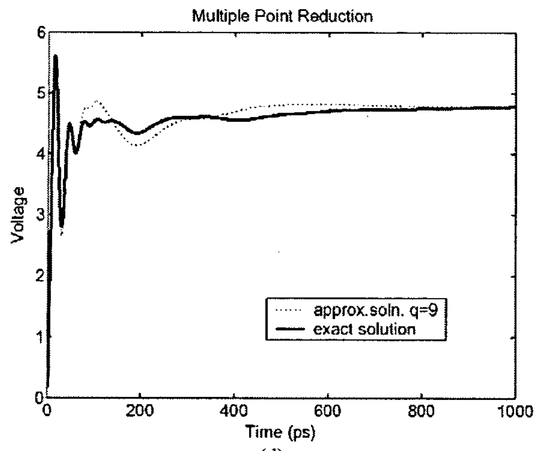
(a)



(b)



(c)



(d)

Fig 8. a),b),c): The step input response of reduced model (at single point $\sigma = 0$), with different model size: $q = 30$, $q = 60$, $q = 70$. d): step input response of 3-point reduced model: $q = 9$.

The test results show that the accuracy of single point reduction is improved by increasing the model size. And to obtain a reasonably good accuracy, the model size should be sufficiently large. However, using multiple frequency point reduction, we can achieve good accuracy with a small model size as shown above, in which case the model size is just $q = 9$, much smaller than the $q \geq 60$ in single point reduction.

From the comparison in the example above we conclude that the multiple frequency point based method is likely most effective for symbolic MOR, because it avoids computing high order Krylov subspace and in turn reduces the ill conditioning of the transformation matrix.

VI. Conclusion

In this paper, the problem of symbolic model order reduction (SMOR) is described. Some algorithms based on PRIMA are proposed and tested. Unlike numeric MOR, which is quite mature for linear dynamic system, symbolic MOR is still not well studied. The methods and algorithms proposed in this paper are only some preliminary results. Further research is needed for practical applications of symbolic model reduction techniques.

References

[1] L. T. Pillage and R. A. Rohrer, "Asymptotic waveform evaluation for timing analysis", *IEEE Trans. Computer-Aided Design,* vol. 9, pp.352-366, April 1990.

[2] K. J. Kerns, I. Wemple, and A. Yang, "Stable and efficient reduction of substrate model networks using congruence transforms," in *Proc. IEEE Int. Conf. Computer-Aided Design (ICCAD),* 1995, pp.207-214.

[3] P. Feldman and R. W. Freund "Efficient linear circuit analysis by Pade approximation via the Lanczos process," *IEEE Trans. Computer-Aided Design*, vol. 14, no. 5, pp. 639-649, May 1995.

[4] A. Odabasioglu, M. Celik and L. T. Pillage, "PRIMA: Passive reduced order interconnect macromodeling algorithm", *IEEE Trans. Computer-Aided Design*, vol. 17, no. 8, August 1998.

[5] E. Chiprout and M. Nakhla, "Analysis of interconnect networks using complex frequency hopping (CFH)," in *IEEE Trans. Computer-Aided Design*, vol. 14, Feb. 1995, pp.186-200.

[6] Eric Grimme, *Krylov Projection Methods for Model Reduction*, Ph.D Thesis, Univ. of Illinois, 1997.

[7] Luca Daniel, C.S. Ong, S. C. Low, K. H. Lee, and J. White, "Geometrically parameterized interconnect performance models for interconnect synthesis", *Proc. Int'l Symposium on Physical Design,* 2002, pp. 202-207.

[8] A. Ruhe, "The rational Krylov algorithm for nonsymmetric eigenvalue problems III: Complex shifts for real matrices," *BIT*, vol.34, pp.165-176,1994.

[9] Y. Liu, L.T. Pileggi, and A.J. Strojwas, "Model order-reduction of RC(L) interconnect including variational analysis". in *Proc. 36th Design Automation Conference*, New Orleans, LA, 1999, pp. 201-206.

[10] Waterloo Maple Inc, *Maple 7,* 2001.

Parametric Reduced Order Modeling for Interconnect Analysis [*]

Guoyong Shi and C.-J. Richard Shi

Electrical Engineering Department
University of Washington
Seattle, WA 98195
{gshi,cjshi}@ee.washington.edu

Abstract— VLSI circuit models are subject to parameter variations due to temperature, geometry, process, and operating conditions. Parameter model order reduction is motivated by such practical problems. The purpose is to obtain a parametric reduced order model so that repeated reduction can be avoided. In this paper we propose two techniques: a nominal projection technique and an interpolation technique. The nominal projection technique is effective for small parameter perturbation by using a robust projection. The interpolation technique takes the advantage of simple matrix structure resulting from the PVL algorithm. A new moment matching concept in the discrete-time domain is also introduced, which is intended for a better performance in waveform matching and stability. Interconnect examples are used to test the effectiveness of the proposed methods.

I. Introduction

Gigahertz frequency operation is already a common practice in the current VLSI technology. Accurate interconnect modeling and analysis have gained increasing importance in state-of-the-art System-on-Chip (SoC) designs. The recent tutorial paper by Achar and Nakhla [1] presents a comprehensive review in this area.

Accurate interconnect models usually end up with high order. For fast analysis, model order reduction techniques have emerged as a valuable tool for such models. Also in many applications, models are likely parametric; for example, the geometric layout parameters, frequency dependent RLC values, and others. Parametric models facilitate synthesis and optimization. If a parametric model is large, it is favorable to have a reduced order model also parametric for efficient synthesis. Traditionally, parametric models are analyzed by repeated simulations and statistical experiments [3]. Interval analysis is another useful tool [13]. If a model is of high order, these methods are less efficient.

Although a number of model reduction techniques have been proposed in the literature (see a survey in [2]), almost all of them are numerical [10, 12, 7, 11]. They must be adapted to treat parametric models. Only a few research results reported in the literature deal with parametric models. Weile *et al.* treated two-parameter linear model reduction problem [15], where the parametric model matrices take the linear combination form as $p_1 M_1 + p_2 M_2$ with p_1 and p_2 being the parameters and M_1 and M_2 known matrices. The reason of choosing this special form is that the parameters are easily maintained after reducing the model order by a projective transformation. The projection is constructed from the Taylor expansion of the transfer function with respect to multiple parameters, from which moments are computed and matched. However, the number of moments in the multivariate expansion increases exponentially as the order increases, leading to an exponentially increasing computation for matching the high order moments. The idea of [15] is applied to multiple-line bus synthesis where the model parameters are wire spacing and wire width [6]. The work in [9] treats model involving parameters due to manufacturing variation. It applies the matrix perturbation theory of singular value decomposition and uses a parameter identification technique by assuming second order polynomial expressions for the variations of the dominant eigenvalue/eigenvectors and the congruence transformation. Because of the high complexity of the computation involved in this method, its applicability is very limited.

This paper is organized as follows. In Section II we briefly review the formulation of linear model order reduction and the projection method. Then we pose the general parametric model reduction problem and propose a nominal projection idea applicable to small range parameter perturbation. In Section III we introduce a new concept of moment matching in the discrete-time domain and argue that this new approach could possibly improve the robustness of nominal projection and the stability of the reduced model. An interpolation technique is introduced in Section IV, which takes the advantage of the simple matrix structure resulting from the PVL algorithm. Examples are presented in Section V. Finally, this paper is concluded in Section VI.

[*]This research was supported in part by DARPA NeoCAD Program under Grant No. 66001-01-1-8920 and by SRC under Contract No. 2001-TJ-921.

II. Model reduction by projection

We consider circuit models that are modeled by differential-algebraic equations

$$\begin{aligned} C\frac{dx}{dt} + Gx &= Fu \\ y &= Lx \end{aligned} \tag{1}$$

where $x \in \mathbb{R}^n$ is the state vector, $u \in \mathbb{R}^m$ is the input (source) vector, and $y \in R^\ell$ is the output (measurement) vector. The transfer function of model (1) is

$$H(s) = L(Cs + G)^{-1}F. \tag{2}$$

The model order reduction problem is to find a reduced order model

$$\begin{aligned} \hat{C}\frac{d\xi}{dt} + \hat{G}\xi &= \hat{F}u \\ y &= \hat{L}\xi \end{aligned} \tag{3}$$

where $\xi \in \mathbb{R}^q$ is the reduced state vector and $q \ll n$ is the reduced model order. The transfer function of the reduced model becomes

$$\hat{H}(s) = \hat{L}(\hat{C}s + \hat{G})^{-1}\hat{F}. \tag{4}$$

A practical requirement is that model (3) be a good approximation of the full order model (1) in the frequency domain.

Moment matching by projection is an efficient model order reduction technique [14]. Recent progress in numerical computation has made this approach widely accepted in practice [7].

Let W and V be two real matrices in $\mathbb{R}^{n\times q}$ so that $W^{\mathrm{T}}V$ is invertible. By restricting the state x in the subspace spanned by the columns of V, we can substitute x in (1) by $V\xi$ and pre-multiply the first equation by W^{T} to obtain a reduced order model (3) with

$$\hat{C} = W^{\mathrm{T}}CV, \;\; \hat{G} = W^{\mathrm{T}}GV, \;\; \hat{F} = W^{\mathrm{T}}F, \;\; \hat{L} = LV.$$

The two matrices W and V used in this reduction process are called projection matrices. They are constructed from standard algorithms for computing basis vectors of Krylov subspaces [7, 11]. If we choose $W = V$, then the transform is called the *congruence transform.*

Parametric linear time-invariant models can be described by

$$\begin{aligned} C(\beta)\frac{dx}{dt} + G(\beta)x &= F(\beta)u \\ y &= L(\beta)x \end{aligned} \tag{5}$$

where $C(\beta)$, $G(\beta)$, $F(\beta)$, and $L(\beta)$ are model matrices depending on the parameter vector β containing a number of parameters. Ideally we would like to have a linear reduced order model that retains the same parameters. However, directly approaching this problem by symbolic linear algebra is clearly not feasible. In this paper we are interested in computationally feasible approaches to the parametric model reduction problem.

In many applications, the model parameters only perturb around some nominal values. For such cases, the parametric reduction can be formulated as a robust reduction problem. That is, we construct *nominal projections* from a set of nominal parameters and use them for reducing models with perturbed parameters. Since the foundation of projection-based reduction is subspace, we believe that the nominal subspace would possess certain degree of robustness if constructed appropriately. This idea will be tested in the experiment section.

The traditional projection algorithms are solely for matching moments in the frequency domain. Since here we are interested in a robust subspace in the time-domain, seeking a robust subspace construction method is one of the goals of this paper. In the next section a new moment matching concept is introduced for this purpose.

III. Moment matching in the discrete-time domain

There are many variants of Krylov subspace. One example of Krylov subspace is the one used in Complex Frequency Hopping (CFH) [4]. It is obtained by expanding the transfer function at some point in the complex plane

$$\begin{aligned} H(s) = L(Cs+G)^{-1}F &= L\left[C(s-s_0) + (Cs_0+G)\right]^{-1}F \\ &= \sum_{i=0}^{\infty} LA^iB(s-s_0)^i \end{aligned} \tag{6}$$

where $A = -(Cs_0 + G)^{-1}C$ and $B = (Cs_0 + G)^{-1}F$ and the matrix $(Cs_0 + G)$ is assumed to be invertible. If we construct a reduced order model which matches the leading moments (coefficients) of the above expansion, the reduced model approximates the original model at least in certain frequency range centering around $s_0 = j\omega$.

The Krylov subspaces used for matching the moments are generated by the triple [7, 11]

$$\left(L,\; (Cs_0+G)^{-1}C,\; (Cs_0+G)^{-1}F\right). \tag{7}$$

Standard Lanczos or Arnoldi algorithm can be used for this purpose.

Here we present a new approach to moment matching, which is from the time-domain perspective. If we discretize a continuous-time system using some discretization method, we obtain a discrete-time model. Then we can expand the transfer function in the z-domain and match the coefficients of those z^k terms, called z-moments.

For simplicity, we use the backward Euler formula to discretize the continuous-time model (1). The uniform time-step backward Euler formula is

$$\dot{x}_{k+1} = \frac{x_{k+1} - x_k}{h}$$

where $h > 0$ is the time-step length and $x_k = x(kh)$. After substitution, the continuous-time model (1) is discretized to

$$\begin{aligned} (\gamma C + G)x_{k+1} &= \gamma C x_k + F u_{k+1} \\ y_{k+1} &= L x_{k+1} \end{aligned} \tag{8}$$

where $\gamma = 1/h$. Suppose $(\gamma C + G)$ is invertible, the state impulse response of (8) consists of the vectors

$$\left\{\Phi F,\ \Phi(\gamma C)\Phi F,\ [\Phi(\gamma C)]^2 \Phi F,\ \cdots\right\}$$

where $\Phi = (\gamma C + G)^{-1}$. These recursive vectors are clearly related to a Krylov subspace formed by the pair

$$\left((\gamma C + G)^{-1}C,\ (\gamma C + G)^{-1}F\right). \tag{9}$$

The basis vectors of this Krylov subspace can be obtained by Arnoldi algorithm [11]. On the other hand, from the input-output point of view, one can use the triple

$$\left(L,\ (\gamma C + G)^{-1}C,\ (\gamma C + G)^{-1}F\right) \tag{10}$$

to obtain a pair of bi-orthonormalized dual Krylov subspaces by Lanczos algorithm [7].

The Krylov subspace in (9) takes exactly the same form of rational Krylov subspace studied in Grimme's thesis [8]. However, Grimme came up with the same type of Krylov subspace from the shifted system of linear equations, from which the exact meaning of the real parameter γ is not clear. Furthermore, as we shall demonstrate below, moment matching in the z domain has the effect of waveform matching, which is also not observed in the Grimme's formulation. To avoid confusion we keep calling the Krylov subspace in (9) *rational Krylov subspace*.

Coincidentally, the rational Krylov subspace is also related to the moments by expanding the transfer function $H(s)$ at a positive real point γ by choosing $s_0 = \gamma$ in (6). Because of this simple connection, the rational Krylov subspace has already been used in many works, such as [7, 2], with good experimental results but without much justification. Grimme made some effort in his thesis ([8], Chapter 6), but did not reach a conclusive result.

The discrete-time moment matching can formally be described as follows. Let the columns of matrix V be the orthonormal basis vectors of the Krylov subspace in (9) obtained from the Arnoldi algorithm. Then the following identities hold:

$$\left[(\gamma C + G)^{-1}C\right]^i (\gamma C + G)^{-1}F = V\left[(\gamma\hat{C} + \hat{G})^{-1}\hat{C}\right]^i (\gamma\hat{C} + G)^{-1}\hat{F} \tag{11}$$

for $i = 0, 1, \cdots, q-1$, where

$$\hat{C} = V^{\mathrm{T}}CV,\quad \hat{G} = V^{\mathrm{T}}GV,\quad \hat{F} = V^{\mathrm{T}}F.$$

This result is established in [8]. If we use V for projection and let

$$\hat{Y}(z) = \hat{L}\left[(\gamma\hat{C} + \hat{G})z - \gamma\hat{C}\right]^{-1}\hat{F}zU(z)$$

be the reduced order transfer function in the z-domain where $\hat{L} = LV$, then it follows that the leading q moments of $\hat{Y}(z)$ match those of $Y(z)$, which implies that

$$\hat{y}_k = y_k, \qquad \text{for } k = 0, 1, \cdots, q.$$

This means that, starting from the same zero initial condition with the same input, the discrete-time responses of the full and reduced order models match at least in the first q steps. Note that matching a set of discretized points can be viewed as a constraint on the waveforms of the two models. Furthermore, due to the shifting property of discrete-time systems, matching the discretized points at the initial period may have a global effect, which implies global waveform matching in the time-domain. It would be interesting to derive an error bound in the time-domain based on this observation.

It is worth noting that a discretization by using the trapezoidal rule results in a Krylov subspaces same as in (9). This indicates from another angle that matching in a subspace is a much more general constraint than just matching several discrete-time points.

Remark 1 *The optimal choice of γ is not discussed here; in fact it is a further research topic. Since γ is the inverse of the time-step taken in discretization, it should be chosen so that the sampled state vectors have sufficient information for characterizing a model.*

IV. Interpolation method

The interpolation method is motivated by the simple matrix structure resulting from the PVL algorithm for model order reduction [7]. The input to the PVL algorithm is the matrix triple

$$\left(L,\ (\gamma C + G)^{-1}C,\ (\gamma C + G)^{-1}F\right).$$

We illustrate the interpolation principle by using a single-input-single-output (SISO) circuit model (MIMO cases can be treated similarly.) In the SISO case, we use the row vector ℓ^{T} in place of L and a column vector b in place of B.

The q step Lanczos algorithm (assuming no breakdowns) applied to the preceding triple (with $L = \ell^{\mathrm{T}}$ and $B = b$) yields a reduced order model in the state space

$$\begin{aligned} T_q\frac{d\xi}{dt} + (I - \gamma T_q)\,\xi &= e_1 u \\ y &= (\ell^{\mathrm{T}}b)e_1^{\mathrm{T}}\xi \end{aligned} \tag{12}$$

with the transfer function

$$\hat{H}(s) = (\ell^{\mathrm{T}}b)e_1^{\mathrm{T}}\left[T_q(s-\gamma) + I\right]^{-1}e_1, \tag{13}$$

where e_1 is the first column of the $q \times q$ identity matrix, $(\ell^{\mathrm{T}}b)$ is a scalar, and the matrix T_q is a $q \times q$ tridiagonal matrix. Thus, from the interpolation perspective, the free parameters that determine a reduced order model include the scalar $(\ell^T b)$ and the $3q-2$ possibly nonzero elements of the tridiagonal matrix T_q. Consequently, we can represent each reduced order model by a $(3q-1)$-dimensional vector.

The basic steps involved in the interpolation method are outlined here. Let $\mathcal{M}_i,\ i = 1, 2, \cdots, N$, be the N models sampled at the grid points of the parameters. Let $v_i \in \mathbb{R}^{3q-1}$ be the vector representing the models reduced from $\mathcal{M}_i$ by PVL. Reduced order models for new parameter values are obtained by interpolating the vectors $v_i,\ i = 1, 2, \cdots, N$.

There are many different ways to do interpolation. Certainly we should choose numerically simple but effective methods in that the computation should be many orders faster than running a whole reduction algorithm. For a single parameter, the Lagrange interpolation is a good candidate. Let $p_i,\ i = 1, 2, \cdots N$, be the grid points of a parameter p, which varies in the interval $[p_1, p_N]$. The basis polynomial for Lagrange interpolation is defined as (see [5], page 285)

$$\delta_i(p) = \prod_{\substack{j=1\\ j\neq i}}^{N}(p - p_j) \Big/ \prod_{\substack{j=1\\ j\neq i}}^{N}(p_i - p_j). \tag{14}$$

It is readily verified that $\delta_i(p_j) = \delta_{ij}$, where δ_{ij} is the Kronecker delta. Then for any $p \in [p_1, p_N]$, the model corresponding to p can be interpolated by

$$v(p) = \sum_{i=1}^{N} \delta_i(p) v_i.$$

For multiple parameters, there is a straightforward extension from the one-parameter interpolation by constructing multivariate basis polynomials from the direct product of single-parameter Lagrange basis functions as defined in (14). However, this approach is numerically not practical. Hence, in the multiple parameter case we use linear interpolation of the surrounding sample points, which is implemented as follows. Suppose we have K parameters, denoted by $\vec{p} = (p_1,\ p_2\ ,\cdots,\ p_K)$. Each parameter is gridded separately and all sampled models are reduced by PVL. Then, given a new set of parameters, first the surrounding sample points are identified. Let $[p_k^\ell, p_k^r]$ be the smallest interval containing the kth parameter value p_k. Let $\mathcal{P}$ denote the set of parameter tuples $\{(p_1^t, \cdots, p_K^t) : t = \ell \text{ or } r\}$. Let $v_{(p_1^t,\cdots,p_K^t)}$ be the sampled model vector obtained at one of the surrounding sample points. Then the new reduced model at the parameter vector $\vec{p}$ can be written as

$$v(\vec{p}) = \sum_{(p_1^t,\cdots,p_K^t)\in\mathcal{P}} \left[\prod_{j=1}^{K} \delta\left([p_j^\ell, p_j^r], p_j^t, p_j\right)\right] v_{(p_1^t,\cdots,p_K^t)} \tag{15}$$

where the function $\delta([a, b], p, x)$ is defined as one of the linear basis functions for interpolation over the interval $[a, b]$, i.e.

$$\delta([a, b], a, x) = \frac{x - b}{a - b} \qquad \text{and} \qquad \delta([a, b], b, x) = \frac{x - a}{b - a}.$$

There are two aspects of complexity involved in the interpolation method: one is the computation complexity and the other is the memory requirement. Depending on the operating frequency and the physical properties of interconnect, different reduced orders are needed to achieve acceptable analysis accuracy. Resistive interconnects can normally be analyzed by using very low order models with sufficient accuracy. However, inductive interconnects usually require higher order models to characterized the resonance effect at high frequencies; hence, the reduced model order should be chosen relatively higher. Obviously, the computation complexity of interpolation method depends on the order q, the number of parameters, and the number of sample models. For the reduction of each sample model, PVL is an extremely efficient algorithm (one LU factorization plus some matrix-vector multiplications).

For the purpose of interpolation, a number of reference models must be created first. The principle is similar to a look-up table. The reference models are created from the models sampled at the grid points of the parameters. Because of the possibility of exponentially increasing computation in the multi-parameter case, the number of parameters cannot be too large for applying the interpolation method. One can take the advantage of those insensitive parameters by using fewer number of grids for such parameters. For interpolation purpose, the reduced order models at the sampled parameter points must be stored in memory. The memory requirement is proportional to the product of the number of samples N and the reduced model order q.

Finally we mention that the stability of the interpolated model is a property related to the sample models. By continuity, the stability of all models for interpolation would likely imply that the interpolated model is stable as well.

V. Examples

The circuit shown in Fig. 1 is discretized from a one-dimensional interconnect or transmission line. Inductors are included in order to consider the inductive effect explicitly. For demonstration purpose, we assume that the model parameters characterizing different physical properties have been converted to the RLC values as parameters. The state space model is formulated by modified nodal analysis (MNA) with the nodal voltages and the currents passing the inductors as the state variables.

The nominal projection method is tested first. Note that higher order moments are needed for inductive interconnect analysis. In this test a 320th order model is reduced to 50th order with the voltage source as the input

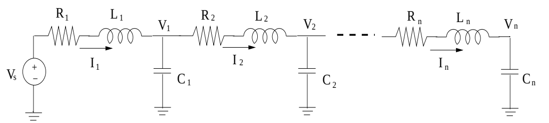


Fig. 1. An RLC line.

and V_1 (the voltage at node 1) as the output. The nominal RLC values are taken to be uniform with $R = 0.2\,\Omega$, $L = 1.0\,nH$, and $C = 0.5\,pF$. We choose a $\gamma = 10^9$ for discrete moment matching. Figure 2 shows the frequency responses of both full and reduced order models together with the error plot. The discrete moment matching yields a good approximation over a wide frequency band.

Then the RLC parameters are perturbed up to $\pm 50\%$ to test whether the reduced model maintains certain accuracy by using the nominal projection. Shown in Fig. 3 is the frequency response result and the error plot. The frequency response of the nominal full order model is also plotted (the dotted curve) to indicate the perturbation effect. Clearly, the frequency response of the reduced order model still captures the frequency response of the full order very well, but with a little sacrifice of the accuracy. An important observation from this test is that when a model is perturbed, the Krylov subspace associated the perturbed model actually is not perturbed much. Hence, a new model reduced by the nominal projection still captures the poles and zeros of the perturbed model. A theoretical justification of this experimental is under development.

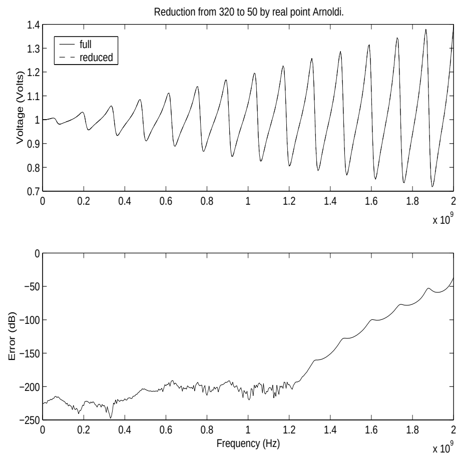


Fig. 2. Nominal reduction.

Next the same circuit in Fig. 1 is used to test the interpolation method. We assume that all R_i, L_i, and C_i take respectively the uniform values with $R_i = R_0$, $L_i = L_0$, and $C_i = C_0$, for all i and R_0, L_0, and C_0 are treated as three parameters of this model.

To study the sensitivity of R_0, we assume that R_0 varies in the interval $[0.1,\,1.0]\Omega$, and $L_0 = 1.0\,nH$ and $C_0 = 0.1\,pF$ are fixed. The voltage V_1 remains as the

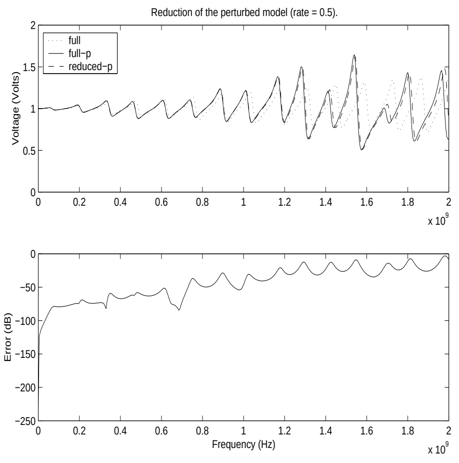


Fig. 3. Reduction of a perturbed model by nominal reduction. The dotted frequency plot indicates the nominal full order model for reference.

output. This model has only one parameter. The interval [0.1, 1.0] is sampled by 11 equally spaced points and each of the 11 sampled models is reduced by PVL to 20th order. Then all other reduced order models are created by interpolation. Three test results are shown in Fig. 4 for $R_0 = 0.407,\ 0.694,\ 0.839\,\Omega$. It is clear that the resonance modes at high frequency are quite sensitive to the minor change of R_0 and the reduced models obtained by interpolation approximate the full-order model very well (as indicated by the error curves).

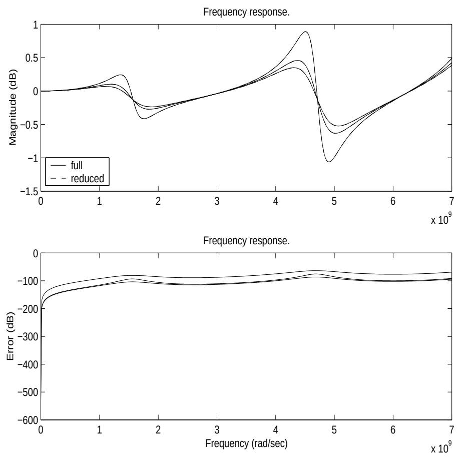


Fig. 4. Test results for one RLC line (three cases).

The interpolation method is also tested on another example with two coupled transmission lines as shown in Fig. 5. The input is $u = V_s$ and the voltage V_{11} is chosen as the output. The test case assumes that $R_{ij} = R_0$, $L_{ij} = L_0$, $C_{ij} = C_0$ for all i, j and $CC_i = CC_0$ for all i. Thus there are four parameters in this case. The interconnects are divided into 50 stages, resulting in a 200th order model. Each model is reduced to 20th order. For the plots in Fig. 6 we chose $R_0 \in [0.1,\,0.2]\,\Omega$ with 4 grids, $L_0 \in [0.1, 0.5]\, nH$ with 3 grids, $C_0 \in [0.1, 0.5]\, pF$ with 3 grids, and $CC_0 \in [0.1, 0.2]\, pF$ with 3 grids. Thus in total we need to collect 108 sample models and reduce them by PVL. Show in Fig. 6 are the reduction results for the three randomly generated models and their reductions by linear interpolation over the surrounding points. We see that the frequency responses change drastically despite that the parameters only change mildly. Hence the resonance modes are very sensitive to the interconnect parameter variation. Regardless of the frequency response variation, the reduced models obtained by interpolation all approximate their full order models very well (see the error plots).

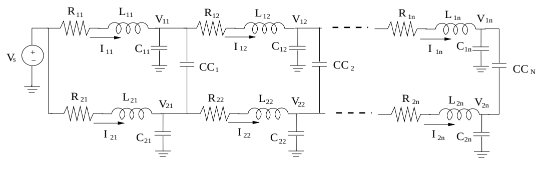


Fig. 5. Two coupled RLC lines.

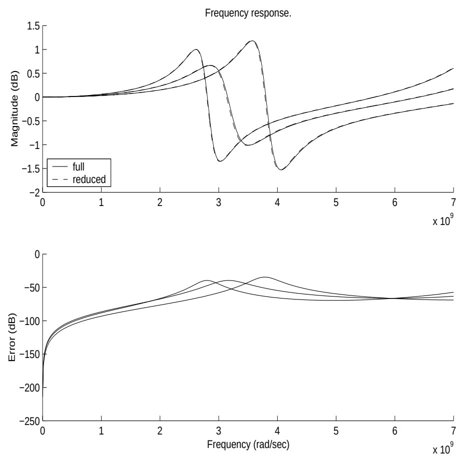


Fig. 6. Test results for the coupled interconnect (three cases).

VI. Conclusion

Techniques for linear model order reduction have reached maturity. However, new problems still bring challenges to them. Parametric model order reduction is one of the practical problems, to which the traditional methods do not apply directly. Interconnect analysis is one of sources for such problems. Inductive effects of interconnect is being recognized as important for accurate delay measurement and design. This paper proposes two ideas for solving parametric model reduction under the assumption that the parameters have variations in a limited range. The application to interconnect analysis has been demonstrated by examples. Continuing research effort is needed for more general and effective methods.

References

[1] R. Achar and M.S. Nakhla. Simulation of high-speed interconnects. *Procedings of the IEEE*, 89(5):693–728, 2001.

[2] Z. Bai. Krylov subspace techniques for reduced-order modeling of large-scale dynamical systems. *Applied Numerical Mathematics*, 43:9–44, 2002.

[3] N. Chang, V. Kanevsky, B. Queen, S. Nakagawa, and S.-Y. Oh. 3-sigma worst-case calculation of delay and crosstalk for critical net. In *Proc. ACM/IEEE International Workshop on Timing Issues in the Specification and Synthesis of Digital Systems*, 1997.

[4] E. Chiprout and M.S. Nakhla. Analysis of interconnect networks using complex frequency hopping (CFH). *IEEE Trans. on Computer-Aided Design*, 14(2):186–200, February 1995.

[5] G. Dahlquist and A. Björck. *Numerical Methods.* Prentice Hall, Inc., Englewood Cliffs, NJ, 1974.

[6] L. Daniel, C. S. Ong, S. C. Low, K. H. Lee, and J. White. Geometrically parameterized interconnect performance models for interconnect synthesis. In *Proc. Int'l Symosium on Physical Design*, pages 202–207, San Diego, CA, 2002.

[7] P. Feldmann and R.W. Freund. Efficient linear circuit analysis by Padé approximation via the Lanczos process. *IEEE Trans. on Computer-Aided Design*, 14(5):639–649, May 1995.

[8] E.J. Grimme. *Krylov Projection Methods for Model Reduction.* PhD thesis, ECE Dept., University of Illinoise at Urbana-Champaign, 1997.

[9] Y. Liu, L.T. Pileggi, and A.J. Strojwas. Model order-reduction of RC(L) interconnect including variational analysis. In *Proc. 36th Design Automation Conference*, pages 201–206, New Orleans, LA, 1999.

[10] B.C. Moore. Principal component analysis in linear systems: controllability, observability, and model reduction. *IEEE Trans. on Automatic Control*, AC-26(1):17–32, February 1981.

[11] A. Odabasioglu, M. Celik, and L.T. Pileggi. PRIMA: Passive reduced-order interconnect macromodeling algorithm. *IEEE Trans. on Computer-Aided Design*, 17(8):645–654, August 1998.

[12] L.T. Pillage and R.A. Rohrer. Asymptotic waveform evaluation for timing analysis. *IEEE Trans. on Computer-Aided Design*, 9:352–366, April 1990.

[13] C.-J.R. Shi and M.W. Tian. Simulation and sensitivity of linear analog circuits under parameter variations by robust interval analysis. *ACM Trans. on Design Automation of Electronic Systems*, 4(3):280–312, July 1999.

[14] C.D. Villemagne and R.E. Skelton. Model reduction using a projection formulation. *Int. J. Control*, 46:2141–2169, 1987.

[15] D.S. Weile, E. Michielssen, E. Grimme, and K. Gallivan. A method for generating rational interpolant reduced order models of two-parameter linear systems. *Applied Mathematics Letters*, 12(5):93–102, 1999.

Model Order Reduction by Dominant Subspace Projection: Error Bound, Subspace Computation and Circuit Applications

Guoyong Shi, *Member, IEEE* and C.-J. Richard Shi, *Senior Member, IEEE*

***Abstract*— Balanced truncation is a well-known technique for model order reduction with a known uniform reduction error bound. However, its practical application to large-scale problems is hampered by its cubic computational complexity. While model order reduction by projection to approximate dominant subspaces without balancing has produced encouraging experimental results, the approximation error bound has not been fully analyzed. In this paper, a square-integral reduction error bound is derived for unbalanced dominant subspace projection by using a frequency-domain solution of the Lyapunov equation. Such an error bound is valid in both the frequency and time domains. Then a dominant subspace computation scheme together with three Krylov subspace options is introduced. It is analytically justified that the Krylov subspace for moment matching at low frequencies is able to provide a better dominant subspace approximation than the Krylov subspace at high frequencies, while a rational Krylov subspace with a proper real shift parameter is capable of achieving superior approximation than the Krylov subspace at low frequency. A heuristic method of choosing a real shift parameter is also introduced based on its connection to the discretization of a continuous-time model. Such a connection has not been recognized elsewhere in the literature. The computation algorithm and theoretical analysis are then examined by several numerical examples to demonstrate the effectiveness. Finally the dominant subspace computation scheme is applied to the model order reduction of two large-scale interconnect circuit examples.**

***Index Terms*— Circuit simulation, dominant subspace, error bound, Krylov subspace, model order reduction, moment matching.**

I. Introduction

Model order reduction is emerging as an effective technique for the modeling and simulation of very large scale integrated circuits (VLSIs) and structures. As the integration level increases and the transistor feature size shrinks, many circuit parasitics can no longer be ignored. Incorporating these parasitics commonly leads to large-scale linear or nonlinear models that are computationally prohibitive even for modern computing resources. Therefore, reducing models before simulation is now becoming a common practice. Large-scale full-order models commonly have a high degree of redundancy. Also in many applications an accurate model at a limited frequency range is of interest. In these cases model order reduction is capable of reducing the model redundancy and providing compact models for efficient simulation. Numerous model reduction techniques have been developed in the past decades, mostly in the control literature. Comprehensive reviews can be found in [1], [2] with an emphasis on large-scale models. Several popular algorithms are compared in [3].

Two representative reduction techniques widely used in circuit simulation are balanced truncation [4], [5], [6] and moment matching [7], [8], [9]. Balanced truncation yields stable reduced order models with a proven uniform error bound. However, due to its cubic computational complexity, balanced truncation is not directly applicable to large-scale model reduction. On the other hand, moment matching has a relatively lower computational complexity and can take the advantage of sparsity in circuit models, and has been widely used for integrated circuit modeling and analysis [8]. Moment matching for model order reduction has been further popularized by the development of numerically stable computation methods based on Krylov subspace [9], [10]. However, since moment matching only matches moments at some local frequency points, the resulting reduced order model may have fairly large errors at some other frequency band. Moreover, a small error bound in the frequency domain does not necessarily imply an accurate waveform matching in the time domain. Maintaining the stability of a reduced order model is also a critical issue. Many authors have made efforts on extending the Krylov subspace methods in a variety of directions for better results, see [11], [12], [13], [14] and the references therein.

Knowing the limitations of both techniques, some researchers attempt to use modified balanced truncation schemes to improve the global approximation accuracy while keeping a low computational cost [1], [15], [16], [17]. The underlying idea is to use approximately computed dominant subspaces and then to project the state space of a full-order model to the dominant subspaces. In this approach efficient computation of the approximate dominant subspace becomes an important task. Unlike the balanced truncation method where the exact Gramians are used for balancing transformation, approximate dominant subspaces are not sufficient for balancing. A natural question in this regard is, if one uses a dominant subspace for model reduction without balancing, what is the error bound? Such an error bound can help us estimate the model reduction accuracy when *approximate* dominant subspaces are used in

Manuscript received ______________; revised ______________. This work was supported in part by the US Defense Advanced Research Projects Agency (DARPA) NeoCAD Program under Grant No. N66001-01-8920 and in part by the National Science Foundation (NSF) CAREER Award under Grant No. 9985507. This paper was recommended by Associate Editor ______________.

G. Shi and C.-J. R. Shi are with the Department of Electrical Engineering, University of Washington, Seattle, WA 98195 USA (e-mails: gshi@ee.washington.edu; cjshi@ee.washington.edu).

practice. Some good experimental results on circuit simulation problems by using the approximate dominant subspaces for model order reduction publications have been reported [16], [17], but an error bound of using unbalanced dominant subspace projection has not been fully analyzed yet. The first contribution of this paper is the establishment of such an error bound. In Section III we follow a frequency domain approach and derive a new $\mathcal{L}_2$ error bound which holds in both the frequency and time domains.

Dominant subspace is not only useful for linear model reduction, but also finds applications in nonlinear model order reduction [18]. However, direct computation of the exact dominant subspace for large-scale models, linear or nonlinear, is in general not feasible in practice. For linear time-invariant models, the exact dominant subspace can only be obtained from solving the exact Gramian solution of a Lyapunov equation, which is of the cubic time complexity. The high computational complexity has motivated many researchers to study approximate solutions of a large-scale Lyapunov equation [15], [19], [20], [21], [22], [23], [24]. It has been observed that frequently the Gramian solved from a Lyapunov equation is of low numerical rank [25], which implies that the state space of the full-order model is dominated by a low-dimensional subspace, see [3], [23] for some examples. In fact originally the balanced truncation principle and dominant subspace principle introduced for model order reduction were motivated by this low-rank phenomenon. Also because of this phenomenon, the computation of low-dimensional approximate dominant subspace becomes important for practical application.

The key idea used for a low-rank approximation of a Gramian is to apply an iterative computation technique in the framework of Krylov subspace [26]. Most low-rank approximation techniques proposed in the literature utilized the idea of forming a Krylov subspace with the matrix pair $(A,\, B)$ (see eqn. (2)) [20], [21], [22]. Only recently have two papers addressed the low-rank approximation using rational Krylov subspace as an extension of the ADI (Alternate Direction Implicit) algorithm [23], [24]. In particular, numerical examples were presented in [24] to demonstrate that different types of Krylov subspaces could give rise to approximate dominant subspaces with different accuracy. But no analysis on the phenomenon was given there. We also found in our experiments similar effects by using different types of Krylov subspaces. In agreement to [24], we found that the Krylov subspace formed by the pair $(A,\, B)$ always yielded the worst results.

In the second part of this paper we carry out a careful study on the approximation accuracy of using three different types of Krylov subspaces. In Section IV we first introduce a general approximate dominant subspace computation scheme based on Krylov subspace and low-order Lyapunov equation solving. Then we justify analytically that the Krylov subspace formed by the pair $(A^{-1},\, A^{-1}B)$ has a better approximation performance than that by the pair $(A,\, B)$, and furthermore a rational Krylov subspace with an appropriately chosen real shift parameter can produce superior approximation results than that of the Krylov subspace $(A^{-1},\, A^{-1}B)$. We also introduce a heuristic for choosing a real shift parameter by building a connection between a real rational Krylov subspace and the discretization of a continuous-time model. We show that model reduction in a *real* rational Krylov subspace bears the physical meaning of waveform matching in the discrete-time domain, in contrast to the implication of local approximation in the frequency domain for rational Krylov subspace with purely imaginary shift parameters.

The computation scheme and theoretical analysis are then examined by numerical examples in Section V. First we use an interconnect circuit example with different orders and element values to demonstrate that the three types of Krylov subspaces do provide different approximation accuracies in the dominant subspace computation, as predicted by the analysis in Section IV. For the evaluation purpose, three measures are introduced for comparing the approximation accuracy. The effectiveness of applying approximate dominant subspace to large-scale model order reduction is further demonstrated by using two interconnect circuits.

The terminologies and notations used in this paper are fairly standard. A matrix is called a Hurwitz matrix if it is asymptotically stable. Vectors without transpose are in column convention. Since the column dimension of a matrix is of special interest in this paper, we specifically use a subscript to indicate the column size of a matrix in many places. The subspace spanned by the columns of a matrix $V_m \in \mathbb{R}^{n\times m}$ is denoted by $\operatorname{span} V_m$. The subspace perpendicular to a subspace $\mathcal{S}$ is denoted by $\mathcal{S}^\perp$. When we say that a matrix V_m spans a subspace, we mean that the columns of matrix V span the subspace, and we call the matrix V_m the basis matrix. The qth order Krylov subspace generated by two matrices A and B is denoted by

$$\mathcal{K}_q(A,B) := \operatorname{span}\begin{bmatrix} B, & AB, & \cdots, & A^{q-1}B \end{bmatrix}. \qquad (1)$$

In some places we shall use the standard matrix manipulation notation used in MATLAB[1]. For example, the notation $M(:,1{:}q)$ means the matrix formed by taking the first q columns from matrix M. The standard basis vectors, i.e., the columns of the identity matrix I, are denoted by e_i whose dimension should be clear from the context if not specified. The transpose of matrix A is denoted by A^{T} and if A is complex, the conjugate transpose of A is denoted by A^{H}. The ith eigenvalue of matrix A is denoted by $\lambda_i(A)$ and the maximal eigenvalue of A is denoted by $\lambda_{\max}(A)$ if A is symmetric. The norm of a vector $\|v\|$ is the conventional Euclidean 2-norm, i.e. $\|v\| := (v^{\mathrm{T}}v)^{1/2}$. The norm of a matrix A is defined by $\|A\| := \lambda_{\max}^{1/2}(AA^{\mathrm{T}})$. The Frobenius norm of A is defined by $\|A\|_F := [\operatorname{tr}(AA^{\mathrm{T}})]^{1/2}$. The maximum singular value of matrix A is denoted by $\sigma_{\max}(A) := \lambda_{\max}^{1/2}(AA^{\mathrm{T}})$.

II. Preliminary

We consider circuits that can be modeled by linear time-invariant (LTI) systems

$$\begin{aligned} \frac{dx}{dt} &= Ax + Bu \\ y &= Cx + Du \end{aligned} \qquad (2)$$

[1]MATLAB is a trademark of The Mathworks, Inc. http://www.mathworks.com.

where $x \in \mathbb{R}^n$ is the state vector, $u \in \mathbb{R}^r$ is the input (source) vector, and $y \in R^p$ is the output (measurement) vector. The transfer function of model (2) is

$$H(s) = C(sI - A)^{-1}B + D. \tag{3}$$

Sometimes it is convenient to use packed notation to represent a linear system and its transfer function

$$H(s) = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] := C(sI - A)^{-1}B + D. \tag{4}$$

The model order reduction problem is to find a reduced order model

$$\begin{aligned} \frac{d\xi}{dt} &= \hat{A}\xi + \hat{B}u \\ y &= \hat{C}\xi + Du \end{aligned} \tag{5}$$

where $\xi \in \mathbb{R}^q$ is the state vector with a reduced order q satisfying $\max\{r, p\} \le q < n$, so that model (5) is a good approximation of the full order model (2). The reduced order model can be written in packed notation as well

$$\hat{H}(s) = \left[\begin{array}{c|c} \hat{A} & \hat{B} \\ \hline \hat{C} & D \end{array}\right] = \hat{C}(sI - \hat{A})^{-1}\hat{B} + D. \tag{6}$$

Since D does not play a role in projection-based model reduction, we simply assume $D = 0$ throughout the paper.

A widely accepted model order reduction formulation is by projection. Let W_q and V_q be two real matrices in $\mathbb{R}^{n\times q}$ satisfying the biorthogonality condition

$$W_q^{\mathrm{T}} V_q = I_q. \tag{7}$$

If we consider the restriction of the state x to $\operatorname{span} V_q$, we can replace x by $V_q\xi$ and premultiply the first equation in (2) by W_q^{T}. The resulting model (5) is of a reduced order with

$$\hat{H}(s) = \left[\begin{array}{c|c} \hat{A} & \hat{B} \\ \hline \hat{C} & 0 \end{array}\right] = \left[\begin{array}{c|c} W_q^{\mathrm{T}} A V_q & W_q^{\mathrm{T}} B \\ \hline C V_q & 0 \end{array}\right]. \tag{8}$$

The quality of a reduced order model obtained by projection can be measured by several different criteria. Typical measures are: (a) the number of moments matched at some frequency points [7], [8], [9], [10] and (b) the uniform frequency domain error bound established for balanced truncation [5].

Given the LTI system in (2) with the system matrix A Hurwitz, two Gramians are important in the context of model order reduction. The controllability Gramian is the unique solution P of the Lyapunov equation

$$AP + PA^{\mathrm{T}} + BB^{\mathrm{T}} = 0, \tag{9}$$

and the observability Gramian is the unique solution Q of the dual Lyapunov equation [27]

$$A^{\mathrm{T}}Q + QA + C^{\mathrm{T}}C = 0. \tag{10}$$

If A is Hurwitz, then both Lyapunov equations (9) and (10) have unique solutions and can be expressed, respectively, in integrals

$$P = \int_0^\infty e^{At}BB^{\mathrm{T}}e^{A^{\mathrm{T}}t}dt \tag{11}$$

and

$$Q = \int_0^\infty e^{A^{\mathrm{T}}t}C^{\mathrm{T}}Ce^{At}dt. \tag{12}$$

Clearly, both Gramians are symmetric and positive semi-definite.

An alternate solution of Lyapunov equation (9) is an integral expression in the frequency domain, which is essentially a result of the matrix form Parseval identity. This alternate expression turns out to be useful for deriving an $\mathcal{L}_2$ error bound for unbalanced dominant projection in the next section. Meanwhile it provides an analytical justification that a Krylov subspace computed at the low frequency can provide a better approximate dominant subspace to be discussed in Section IV. Since the authors have not seen this alternate expression in the literature, a formal proof is provided for completeness.

Lemma 1: Assume that A is asymptotically stable. The controllability Gramian P can be expressed by an integral in the frequency domain, i.e.

$$P = \frac{1}{2\pi}\int_{-\infty}^{\infty} (j\omega I - A)^{-1}BB^{\mathrm{T}}(-j\omega I - A^{\mathrm{T}})^{-1}d\omega. \tag{13}$$

Proof: The Lyapunov equation (9) can be rewritten as

$$(j\omega I - A)P + P(-j\omega I - A^{\mathrm{T}}) = BB^{\mathrm{T}}$$

which is equivalent to

$$\begin{aligned}(j\omega I - A)^{-1}P + P(-j\omega I - A^{\mathrm{T}})^{-1} = \\ (j\omega I - A)^{-1}BB^{\mathrm{T}}(-j\omega I - A^{\mathrm{T}})^{-1}.\end{aligned}$$

Taking integral from $\omega = -\infty$ to ∞ yields

$$\begin{aligned}\int_{-\infty}^{\infty}(j\omega I - A)^{-1}d\omega P + P\int_{-\infty}^{\infty}(-j\omega I - A^{\mathrm{T}})^{-1}d\omega = \\ \int_{-\infty}^{\infty}(j\omega I - A)^{-1}BB^{\mathrm{T}}(-j\omega I - A^{\mathrm{T}})^{-1}d\omega.\end{aligned} \tag{14}$$

Since $(sI - A)^{-1}$ vanishes at $s = \infty$, the integration above can be replaced by a contour integral $\oint_{C_R}$ where C_R is the closed path going from $-jR$ to jR along the $j\omega$ axis and following the left semi-circle $Re^{j\theta}$ for $\theta \in \left[\frac{\pi}{2}, \frac{3\pi}{2}\right]$. For sufficiently large R, this loop encircles all eigenvalues of A but not those of $-A$. By Cauchy Theorem we obtain for R sufficiently large that

$$\frac{1}{j2\pi}\oint_{C_R}(sI - A)^{-1}ds = I, \quad \frac{1}{j2\pi}\oint_{C_R}(sI + A^{\mathrm{T}})^{-1}ds = 0.$$

Then the identity (13) follows immediately by substituting $s = j\omega$ and letting $R \to \infty$. ■

III. Error bound

Suppose that W_q and V_q are two projection matrices in $\mathbb{R}^{n\times q}$ satisfying $W_q^{\mathrm{T}}V_q = I_q$ and the reduced order model (8) is obtained from these two projection matrices. The error of model order reduction is defined to be the difference between the full order model and a reduced order model in the frequency domain, i.e.,

$$E(s) = H(s) - \hat{H}(s). \tag{15}$$

Several error bounds are available in the literature. A well-known error bound was established in [5] for balanced truncation. Its practical use is, however, limited to small-scale models since the computational complexity of solving Lyapunov equations and singular value decomposition is of $\mathcal{O}(n^3)$. Another error bound was derived for moment matching in [28]. However, since moment matching is based on the Taylor expansion in the local sense, an error bound for the truncated terms does not provide much information on the quality of a reduced order model in the wide-band sense.

Since the theoretically solid error bound for balanced truncation cannot be used if the exact Gramians are not available, it is of interest to consider the error bound if only one exact Gramian is used without performing the balancing transformation. Such an error bound would be useful in practice because, as will be discussed later in this paper, we are able to compute an approximate dominant subspace with high accuracy using specially designed Krylov subspace methods. If an error bound is available for a reduced model obtained from an exactly computed dominant subspace, then this error bound together with the subspace approximation error can be used for an estimate of the model reduction error. Moreover, the better an approximate dominant subspace is computed, the more trustful is the error bound. For large-scale models, the exact error bound is rarely computed because of the computational complexity, nevertheless it still serves as a theoretical measure to justify that an accurately computed approximate dominant subspace is trustfully in model order reduction.

Before establishing the new error bound mentioned above, we first derive a general characterization of the error function defined in (15).

Lemma 2: Suppose that the reduced order model (8) is obtained from the two projection matrices W_q and V_q in $\mathbb{R}^{n\times q}$ satisfying $W_q^{\mathrm{T}}V_q = I_q$. The following identity holds for the error $E(s)$ defined in (15),

$$E(s) = L(s)\left(I - V_qW_q^{\mathrm{T}}\right)F(s) \tag{16}$$

where

$$L(s) = \hat{C}(sI-\hat{A})^{-1}W_q^{\mathrm{T}}A + C, \quad F(s) = (sI-A)^{-1}B.$$

By duality, an alternate error expression is

$$E(s) = F_d(s)\left(I - V_qW_q^{\mathrm{T}}\right)L_d(s) \tag{17}$$

where

$$F_d(s) = C(sI-A)^{-1}, \quad L_d(s) = AV_q(sI-\hat{A})^{-1}\hat{B} + B.$$

Proof: In packed notation we can write

$$E(s) = \left[\begin{array}{cc|c} W_q^{\mathrm{T}}AV_q & 0 & W_q^{\mathrm{T}}B \\ 0 & A & B \\ \hline -CV_q & C & 0 \end{array}\right].$$

A state transformation $\begin{bmatrix} I & W_q^{\mathrm{T}} \\ 0 & I \end{bmatrix}$ of this error system leads to

$$E(s) = \left[\begin{array}{cc|c} W_q^{\mathrm{T}}AV_q & -W_q^{\mathrm{T}}A(I-V_qW_q^{\mathrm{T}}) & 0 \\ 0 & A & B \\ \hline -CV_q & C(I-V_qW_q^{\mathrm{T}}) & 0 \end{array}\right].$$

This is equivalent to a system in the augmented state space

$$\begin{aligned} \dot{\eta} &= \hat{A}\eta - W_q^{\mathrm{T}}A(I-V_qW_q^{\mathrm{T}})x \\ \dot{x} &= Ax + Bu \\ e &= -\hat{C}\eta + C(I-V_qW_q^{\mathrm{T}})x \end{aligned}$$

where $\eta = \xi - W_q^{\mathrm{T}}x$, with zero initial conditions. Identity (16) then follows directly from this system.

The dual identity (17) is proven by considering $E^{\mathrm{T}}(s)$ and replacing the triple (A, B, C) by its dual $(A^{\mathrm{T}}, C^{\mathrm{T}}, B^{\mathrm{T}})$. ■

The importance of Lemma 2 lies in the factor $(I - V_qW_q^{\mathrm{T}})$. Since $W_q^{\mathrm{T}}V_q = I_q$, this factor is an oblique projector to the subspace $(\operatorname{span} W_q)^{\perp}$ along $\operatorname{span} V_q$. The dual factor $(I - W_qV_q^{\mathrm{T}})$ is another oblique projector to the subspace $(\operatorname{span} V_q)^{\perp}$ along $\operatorname{span} W_q$.

Lemma 2 has two immediate applications. First, it can be used to check the accuracy of moment matching if V_q and W_q are computed from the moment matching algorithms [7], [9]. For example, if W_q and V_q are generated by the q-step Lanczos process as in PVL [9], then the coefficients of s^{-i} in $E(s)$ up to order $2q$ vanish, i.e., $2q$ moments of $H(s)$ and $\hat{H}(s)$ are matched. Second, Lemma 2 is instrumental for deriving an error bound for $E(s)$ if either W_q or V_q spans respectively the dominant observable or controllable subspace, which is the remaining task of this section.

The standard $\mathcal{L}_2$ norm of the error $E(s)$ is defined as

$$\|E\|_2 := \left\{\frac{1}{2\pi}\int_{-\infty}^{\infty} \operatorname{tr}\left[E(j\omega)E^{\mathrm{H}}(j\omega)\right]d\omega\right\}^{1/2} \tag{18}$$

where $E^{\mathrm{H}}(j\omega) = E^{\mathrm{T}}(-j\omega)$, and the standard $\mathcal{L}_\infty$ norm of $L(s)$ is defined as [27]

$$\|L\|_\infty := \sup_{\omega\in(-\infty,\,\infty)} \sigma_{\max}[L(j\omega)]. \tag{19}$$

The next theorem establishes a bound on $\|E\|_2$, where we only consider the controllability Gramian and assume that the exact dominant subspace corresponding to the dominant singular values is available.

Theorem 1: Let P be the controllability Gramian and $P = U\Sigma U^{\mathrm{T}}$ be the SVD of P, where U is an orthonormal matrix and $\Sigma = \operatorname{Diag}[\sigma_1, \cdots, \sigma_n]$ is a diagonal matrix containing the singular values of P in the descending order. Let $1 \le q < n$. If $V_q = U(:, 1{:}q)$ and $W_q \in \mathbb{R}^{n\times q}$ satisfying $W_q^{\mathrm{T}}V_q = I_q$ are used for reduction projection and assume that the reduced matrix $\hat{A} = W_q^{\mathrm{T}}AV_q$ is asymptotically stable. Then we have

$$\|E\|_2 \le \left(\sum_{i=q+1}^{n}\sigma_i\right)^{1/2} \|L\|_\infty \|I - W_qV_q^{\mathrm{T}}\| \tag{20}$$

where $L = L(s)$ is defined in Lemma 2. If $W_q = V_q$, then (20) reduces to

$$\|E\|_2 \le \left(\sum_{i=q+1}^{n}\sigma_i\right)^{1/2} \|L\|_\infty. \tag{21}$$

Proof: Since $\hat{A}$ is asymptotically stable, the $\mathcal{L}_\infty$ norm $\|L\|_\infty$ is finite. Let $V_q^c = U(:, q+1{:}n)$ be the columns of U

complementary to V_q and $\Sigma_q^c = \mathrm{Diag}[\sigma_{q+1}, \cdots, \sigma_n]$. Using the error expression (16) in Lemma 2 and the integral solution P in Lemma 1, we can bound the $\mathcal{L}_2$ norm $\|E\|_2$ defined in (18) as follows

$$\begin{aligned}\|E\|_2^2 &\leq \mathrm{tr}\{(I - V_q W_q^{\mathrm{T}})P(I - W_q V_q^{\mathrm{T}})\}\|L\|_\infty^2 \\ &= \mathrm{tr}\{(I - V_q W_q^{\mathrm{T}})V_q^c \Sigma_q^c V_q^{c\mathrm{T}}(I - W_q V_q^{\mathrm{T}})\}\|L\|_\infty^2 \\ &\leq \|L\|_\infty^2 \left(\sum_{i=q+1}^{n} \sigma_i\right) \|I - W_q V_q^{\mathrm{T}}\|^2\end{aligned}$$

where we have used the fact that $(I - V_q W_q^{\mathrm{T}})V_q = 0$. Hence the error bound (20) follows. Bound (21) is due to the fact that $\|I_q - V_q V_q^{\mathrm{T}}\| = 1$ for $q < n$. ■

Remark 1: By Parseval's theorem [27], the $\mathcal{L}_2$ norm of error $E(s)$ in the frequency domain is the same as that that in the time domain. Therefore, the reduction error bounds in Theorem 1 are valid in both the time and frequency domains. Consequently, a good waveform approximation is guaranteed in the time domain as well if a good frequency domain approximation is established. Note that the $\mathcal{L}_\infty$ error bound of balanced truncation does not have such a property.

An immediate consequence of Theorem 1 is that if the trailing singular values are all zero, then the projection based reduction does not lose any accuracy.

Corollary 1: Under the same conditions as in Theorem 1, if $\sigma_i = 0$ for $i = q+1, \cdots, n$, then $\hat{H}(s) = H(s)$, i.e. the reduced order model is equivalent to the full order model.

Remark 2: In circuit simulation, passivity is an important property. Since the congruence transformation preserves passivity for circuit models with port formulation, choosing $W_q = V_q$ is preferred if the columns of V_q are orthonormalized [10]. In Theorem 1 we assumed that $\hat{A} = W_q^{\mathrm{T}} A V_q$ is asymptotically stable. However, the stability of $\hat{A} = W_q^{\mathrm{T}} A V_q$ is in general not guaranteed, unless W_q and V_q are obtained from balanced truncation. In practice sometimes $\hat{A}$ might contain a few unstable poles. If this happens, certain postprocessing is needed, such as dropping the unstable poles by extracting the stable subspace. Since $\hat{A}$ is usually a low-dimensional matrix, the stability check and stable subspace extraction are computationally feasible.

Remark 3: Note that Theorem 1 is stated for the controllability Gramian only. A dual result holds for the observability Gramian as well, which follows directly from Lemma 2.

IV. Computation of approximate dominant subspaces

In the preceding section we established an $\mathcal{L}_2$ error bound in model reduction by unbalanced dominant subspace projection. However, since the exact Gramians are not easily computable for large-scale models, we have to resort to approximation in practice for dominant subspaces computation. As long as an approximate dominant subspace is computed with sufficient accuracy, the model reduction accuracy can be estimated by the error bound derived above.

Approximate computation of dominant subspaces for large-scale models has been studied by many researchers in the literature. Most of the results stem from the idea of low-rank approximate solution of a high order Lyapunov equation. The low-rank solution of Lyapunov equation was first addressed by Hodel and Poolla [19], where several heuristic algorithms were proposed. Saad [20] specifically analyzed the low-rank approximation by using the Krylov subspace $\mathcal{K}_m(A, B)$, where the Galerkin condition on the residual was considered. Jaimoukha and Kasenally [21] extended Saad's idea on the single-input-single-output case to the multiple-input-multiple-output case and proposed a GMRES-like solution scheme by deriving an explicit expression for the residual. In all of these papers, the Krylov subspace was formed by the pair (A, B) because of their appearance in the Lyapunov equation (9). However, satisfaction of the Galerkin condition does not necessarily imply that the Krylov subspace formed by the pair (A, B) is optimal for effective dominant subspace computation. Other algorithms along the same line are proposed in [22], [23], [24], where in [23], [24] rational Krylov subspace was used for low-rank approximate solution in the framework of ADI algorithms.

The key point we would like to make in this section is that the traditionally used Krylov subspace $\mathcal{K}_m(A, B)$ in the literature is in fact not the best choice for effective computation of dominant subspace, especially for large-scale models. Some other Krylov subspace options with a commensurate computation complexity can potentially provide better results. In Subsection IV-A we introduce a general dominant subspace computation scheme, which is a different algorithm than the CF-ADI algorithm in [24]. The specific choice of Krylov subspace is not specified in the computation scheme. In Subsection IV-B we review the properties of three types of Krylov subspaces that are optional for the computation scheme. These properties are used in Subsection IV-C for an analytical comparison of their performance in dominant subspace approximation. Finally in Subsection IV-D we establish a new connection between a real rational Krylov subspace and the discretization of a continuous-time model and propose a heuristic way of choosing an appropriate real shift parameter for a real rational Krylov subspace.

A. A dominant subspace computation scheme

We start with the outline of a general iterative dominant subspace computation scheme. Algorithms similar to this scheme have been used in some works [20], [21], but not from the point of view of comparing the performance of different Krylov subspaces. We formulate the scheme in a generic way without specifying the matrices used for the Krylov subspace computation. Later on this generic algorithm is specialized to several different Krylov subspaces and their performances in dominant subspace approximation are compared analytically. The computation scheme is general enough for multiple-input models.

Dominant Subspace Computation Scheme:

Input: Two matrices $\Phi \in \mathbb{R}^{n \times n}$ and $\Theta \in \mathbb{R}^{n \times r}$, the dimension q of an approximate dominant subspace, and an intermediate integer m satisfying $r \leq q \leq m < n$.

Step 1: Run the (block) Arnoldi algorithm [26] to generate the basis vectors for the νth order Krylov subspace $\mathcal{K}_\nu(\Phi, \Theta)$. Let $V_m \in \mathbb{R}^{n\times m}$ $(m \le \nu)$ be the basis matrix, i.e., $\operatorname{span} V_m = \mathcal{K}_\nu(\Phi, \Theta)$.

Step 2: Form the reduced order Lyapunov equation

$$A_m P_m + P_m A_m^{\mathrm{T}} + B_m B_m^{\mathrm{T}} = 0 \tag{22}$$

where

$$A_m = V_m^{\mathrm{T}} A V_m, \qquad B_m = V_m^{\mathrm{T}} B.$$

Step 3: Solve P_m from the mth order Lyapunov equation (22) and find the SVD of P_m, i.e.

$$P_m = U_m \Sigma_m U_m^{\mathrm{T}}$$

where $U_m \in \mathbb{R}^{m\times m}$ is an orthonormal matrix and the singular values of P_m are arranged in the descending order in Σ_m.

Step 4: Extract the leading q column vectors of U_m, denoted by $\widetilde{U}_q = U_m(:, 1 : q)$, and define $\widetilde{V}_q = V_m \widetilde{U}_q$.

Output: Matrix $\widetilde{V}_q$ whose columns span a q-dimensional approximate dominant subspace.

Since the algorithm requires solving a Lyapunov equation of dimension m, we recommend that the intermediate dimension m should not be too large (normally no greater than 100) so that solving the Lyapunov equation (22) is kept at a low cost.

One could choose to return V_m after Step 1 for an approximate subspace as done in Section 8 of [24]. However, V_m directly generated by the Arnoldi algorithm without the correction in the other steps might not capture the dominance very well. Steps 2 to 4 in the computation scheme are merely for a better low-dimensional approximation of dominance subspace with some modest additional computation cost. We refer to the reduction procedure from m basis vectors to q dominant basis vectors as *compaction*.

It is easily verified that if $\widetilde{V}_q$ is returned from the computation scheme, then $\widetilde{A}_q = \widetilde{V}_q^{\mathrm{T}} A \widetilde{V}_q$ and $\widetilde{B}_q = \widetilde{V}_q^{\mathrm{T}} B$ satisfy

$$\widetilde{A}_q \Sigma_q + \Sigma_q \widetilde{A}_q^{\mathrm{T}} + \widetilde{B}_q \widetilde{B}_q^{\mathrm{T}} = 0 \tag{23}$$

where Σ_q is a diagonal matrix containing the leading q singular values of Σ_m. The computation scheme outlined above also yields a low-rank approximation of the Gramian

$$\widetilde{P} = V_m P_m V_m^{\mathrm{T}} = (V_m U_m)\Sigma_m (U_m^{\mathrm{T}} V_m^{\mathrm{T}}). \tag{24}$$

This approximate solution has the property that the Galerkin condition is satisfied [20], i.e.,

$$V_m^{\mathrm{T}} R(\widetilde{P}) V_m = 0, \tag{25}$$

where $R(\cdot)$ is the residual matrix of the Lyapunov equation defined by

$$R(X) = AX + XA^{\mathrm{T}} + BB^{\mathrm{T}}. \tag{26}$$

Remark 4: The dominant subspace computation scheme is formulated for the computation of a dominant controllable subspace. It can also be used for the computation of a dominant *observable* subspace if the dual matrices $(A^{\mathrm{T}}, C^{\mathrm{T}})$ are used in place of (A, B).

B. Three Krylov subspaces for dominance approximation

Note that we did not specify the choice of the matrices Φ and Θ for the Krylov subspace $\mathcal{K}_m(\Phi, \Theta)$ in the computation scheme described above. There are three typical Krylov subspaces optional for the computation scheme. In this subsection we briefly review the construction and basic properties of the Krylov subspaces. These properties will be used to examine the approximation performance of the three types of Krylov subspaces.

A traditional option for Φ and Θ is to choose $\Phi = A$ and $\Theta = B$; that is, the Krylov subspace $\mathcal{K}_m(A, B)$ is used for the dominant controllable subspace computation. The choice of this Krylov subspace used in many earlier works is a direct consequence of the Lyapunov equation in the standard form of (9) and its solution in the conventional form of (11).

The second choice of Krylov subspace comes from the following reformulation. If A is asymptotically stable (hence invertible), the Lyapunov equation (9) can equivalently be written as

$$A^{-1}P + PA^{-T} + A^{-1}BB^{\mathrm{T}}A^{-T} = 0, \tag{27}$$

which remains to be a Lyapunov equation. This operation leads to another way of computing the dominant controllable subspace by using the pair $(\Phi, \Theta) = \left(A^{-1}, A^{-1}B\right)$. We note that in circuit applications obtaining A or A^{-1} involves almost the equal amount of computation, depending on whether inverting the susceptance matrix or the conductance matrix.

The third option for choosing the pair (Φ, Θ) is by using a rational Krylov subspace [29] with a real shift parameter $\gamma > 0$

$$\mathcal{K}_m\left((\gamma I - A)^{-1}, (\gamma I - A)^{-1}B\right). \tag{28}$$

Note that by choosing $\gamma = 0$ this Krylov subspace reduces to the second option $\mathcal{K}_m\left(A^{-1}, A^{-1}B\right)$. It should be noted that there are many possible ways of choosing the shift parameter γ, which in general could be complex and distinct in the iteration, for instance, $\left[(\gamma_1 I - A)^{-1}B, (\gamma_2 I - A)^{-2}B, \cdots\right]$ as typically used in the ADI-type algorithms [23], [24], [30]. Clearly, if a set of distinct shift parameters is used, additional linear solves for $(\gamma_i I - A)^{-1}B$ are required. In this paper we focus ourselves on rational Krylov subspaces with only one real shift parameter γ, and compare its performance to the first two options in dominant subspace approximation. An algorithm using distinct shift parameters can be found in [24].

We are mainly interested in one simple question: which one among the three Krylov subspaces could provide the best results for dominant subspace approximation if the Dominant Subspace Computation Scheme is used with low orders m and q? We learned from our numerical experiments that in general the traditionally used Krylov subspace $\mathcal{K}_m(A, B)$ almost always yielded the worst approximation results, the Krylov subspace $\mathcal{K}_m\left(A^{-1}, A^{-1}B\right)$ could provide a relatively better approximation result, and the rational Krylov subspace (28) with an appropriately chosen positive γ frequently gave rise to the most superior approximation result. This observation is also consistent with the numerical results reported by Li and White [24]; but no analytical justification were given there. In the next subsection we shall attempt to provide an analytical justification for the different approximation performances of the three Krylov subspaces. For this purpose, the following three properties associated with moment matching in the three Krylov subspaces will be useful.

The moment matching property of a Krylov subspace is now well-known. Let $X(s) = F(s)U(s)$ be the Laplace transform of the state x of model (2), where $F(s) = (sI - A)^{-1}B$ is the transfer function from input to state. The Taylor expansion of $F(s)$ can take the following three forms

$$F(s) = \sum_{i=0}^{\infty} A^i B s^{-(i+1)} \tag{29a}$$
$$= -\sum_{i=0}^{\infty} A^{-(i+1)} B s^i \tag{29b}$$
$$= \sum_{i=0}^{\infty} (-1)^i (\gamma I - A)^{-(i+1)} B (s-\gamma)^i \tag{29c}$$

where the first expansion is at $s = \infty$, the second at $s = 0$, and the third at $s = \gamma$. The leading coefficient matrices (vectors) of the terms $s^{-(i+1)}$, s^i, and $(s-\gamma)^i$ in the above three expansions are, respectively, the column vectors forming the three Krylov subspaces we mentioned above.

An important property of (rational) Krylov subspace is that if we reduce the matrices A and B by projection to a low-dimensional Krylov subspace characterized by an orthonormal basis matrix V_q, then the leading q moments of the following two input-to-state transfer functions are matched

$$F(s) = (sI - A)^{-1}B \tag{30a}$$
$$\widetilde{F}(s) = V_q(sI - \hat{A})^{-1}\hat{B} \tag{30b}$$

where
$$\hat{A} = V_q^{\mathrm{T}} A V_q \qquad \text{and} \qquad \hat{B} = V_q^{\mathrm{T}} B. \tag{31}$$

are the two reduced matrices.

The moment matching properties of the three Krylov subspaces are listed below for the general multiple-input case. They will be used in the next subsection for dominance approximation analysis. By an orthonormal basis matrix V_q of a Krylov subspace $\mathcal{K}_m$ we mean that V_q has orthonormal columns (i.e. $V_q^{\mathrm{T}}V_q = I_q$) and $\mathcal{K}_m = \operatorname{span} V_q$.

Property 1: If V_q, $q \le m$, is the orthonormal basis matrix of the Krylov subspace $\mathcal{K}_m(A,\, B)$, then we have

$$A^i B = V_q \hat{A}^i \hat{B} \tag{32}$$

for $i = 0, \cdots, m-1$, where $\hat{A}$ and $\hat{B}$ are defined in (31).

Property 2: If V_q, $q \le m$, is the orthonormal basis matrix of the Krylov subspace $\mathcal{K}_m\left(A^{-1}, A^{-1}B\right)$ and $\hat{A}$ is invertible, then we have

$$A^{-i} B = V_q \hat{A}^{-i} \hat{B}. \tag{33}$$

for $i = 1, \cdots, m$, where $\hat{A}$ and $\hat{B}$ are defined in (31).

Property 3: If V_q, $q \le m$, is the orthonormal basis matrix of the rational Krylov subspace $\mathcal{K}_m\left((\gamma I - A)^{-1},\, (\gamma I - A)^{-1}B\right)$ with $\gamma > 0$, then we have

$$(\gamma I - A)^{-i} B = V_q (\gamma I - \hat{A})^{-i} \hat{B}. \tag{34}$$

for $i = 1, \cdots, m$, where $\hat{A}$ and $\hat{B}$ are defined in (31), and γ is chosen such that the matrix inversions exist.

The first property follows directly from the Arnoldi algorithm. Its proof is straightforward and can be found in, for example, [7], [31]. The proof of Property 3 can be found in ([29], Section 3.1). The proof of Property 2 is slightly different from the other two, because the reduced matrix $V_q^{\mathrm{T}} A V_q$ rather than $V_q^{\mathrm{T}} A^{-1} V_q$ is formed. For completeness a proof is provided in Appendix .

To be specific, we shall use the term *moment matching at high frequency* to indicate the first type of Krylov subspace which is obtained by the Taylor expansion at $s = \infty$, the term *moment matching at low frequency* to indicate the second type of Krylov subspace which is obtained by the Taylor expansion at $s = 0$, and the term *moment matching at the real* γ for the third type Krylov subspace which is obtained by the Taylor expansion at $s = \gamma$.

C. Comparison of performances in dominance approximation

Our experiment and the numerical examples reported in [24] all show that the three Krylov subspaces examined in the preceding subsection have different performances for the approximation of dominant subspaces, especially for large-scale models. The goal of this subsection is to justify using the tools developed so far that in the computation of approximate dominant subspace for large-scale models, the following two observations are in general true.

(a) The Krylov subspace $\mathcal{K}_m(A^{-1}, A^{-1}B)$ has a better performance than that of $\mathcal{K}_m(A,\, B)$.
(b) The Krylov subspace $\mathcal{K}_m\left((\gamma I - A)^{-1},\, (\gamma I - A)^{-1}B\right)$ with an appropriately chosen $\gamma > 0$ has a superior performance than that of $\mathcal{K}_m(A^{-1}, A^{-1}B)$.

We first justify the first observation by using Properties 1 and 2 and the the integral expression of P in the frequency domain (see (13)). Let $V^{(\infty)}$ and $V^{(0)}$ be the basis matrices of the Krylov subspaces $\mathcal{K}_m(A,\, B)$ and $\mathcal{K}_m\left(A^{-1}, A^{-1}B\right)$, respectively. Let

$$\hat{A}^{(t)} = {V^{(t)}}^{\mathrm{T}} A V^{(t)}, \qquad \hat{B}^{(t)} = {V^{(t)}}^{\mathrm{T}} B,$$

and $F^{(t)}(s) = V^{(t)}\hat{F}^{(t)}(s)$ where $\hat{F}^{(t)}(s) = (sI - \hat{A}^{(t)})^{-1}\hat{B}^{(t)}$ for $t = 0,\, \infty$.

Let $\hat{P}^{(t)}$ $(t = 0,\, \infty)$ be the solution of the reduced order Lyapunov equation

$$\hat{A}^{(t)}\hat{P}^{(t)} + \hat{P}^{(t)}\hat{A}^{(t)\mathrm{T}} + \hat{B}^{(t)}\hat{B}^{(t)\mathrm{T}} = 0.$$

Then according to the Dominant Subspace Computation Scheme, the two matrices $\widetilde{P}^{(t)} := V^{(t)}\hat{P}^{(t)}{V^{(t)}}^{\mathrm{T}}$ for $t = 0, \infty$ are two approximations to the the controllability Gramian P represented as (see Lemma 1)

$$P = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(j\omega)F^{\mathrm{H}}(j\omega)d\omega. \tag{35}$$

Then we have

$$\begin{aligned}\widetilde{P}^{(t)} &= V^{(t)}\hat{P}^{(t)}V^{(t)^{\mathrm{T}}} \\ &= \frac{1}{2\pi}\int_{-\infty}^{\infty} V^{(t)}\hat{F}^{(t)}(j\omega)\hat{F}^{(t)^{\mathrm{H}}}(j\omega)V^{(t)^{\mathrm{T}}}d\omega \\ &= \frac{1}{2\pi}\int_{-\infty}^{\infty} F^{(t)}(j\omega)F^{(t)^{\mathrm{H}}}(j\omega)d\omega \\ &\approx \frac{1}{2\pi}\int_{-\infty}^{\infty} F(j\omega)F^{\mathrm{H}}(j\omega)d\omega.\end{aligned}$$

The accuracy of the approximation in the last step can now be evaluated by the moment matching properties of the two associated Krylov subspaces.

Properties 1 and 2 imply that the leading m moments of $F^{(\infty)}(s)$ match those of $F(s) = (sI - A)^{-1}B$ at $s = \infty$, while the leading m moments of $F^{(0)}(s)$ match those of $F(s)$ at $s = 0$. Since $(j\omega I - A)^{-1}$ rolls off as $\omega \to \infty$, the magnitude of the integral of P in (35) mainly comes from the contribution of $F(j\omega) = (j\omega I - A)^{-1}B$ at the low frequency part rather than from the high frequency part. Hence in the sense of approximation, an integral with its integrand matching the moments of $F(s)$ at the low frequency and rolling off at the high frequency must have a better approximation to the Gramian P than the one that matches only the moments of $F(s)$ at the high frequency which is the rolling off (non-significant) part. Observation (a) is thus justified. Fortunately this result is not against the practice where the circuit operation frequency band is typically below the gigahertz level, never going toward the infinitely high frequencies.

Observation (b) is justified by a different approach that uses another representation of the controllability Gramian P in infinite series. Note that the Lyapunov equation (9) can equivalently be written as

$$(\gamma I - A)P(\gamma I - A)^{\mathrm{T}} = (\gamma I + A)P(\gamma I + A)^{\mathrm{T}} + 2\gamma BB^{\mathrm{T}}.$$

Assuming $(\gamma I - A)$ invertible, we get

$$P = \Phi(\gamma)P\Phi^{\mathrm{T}}(\gamma) + (2\gamma)(\gamma I - A)^{-1}BB^{\mathrm{T}}(\gamma I - A)^{-\mathrm{T}} \quad (36)$$

where $\Phi(\gamma) := (\gamma I - A)^{-1}(\gamma I + A)$. This is another type of Lyapunov equation arising from discrete-time systems; its solution can be obtained by the following iteration

$$P_{k+1} = \Phi(\gamma)P_k\Phi(\gamma)^{\mathrm{T}} + (2\gamma)(\gamma I - A)^{-1}BB^{\mathrm{T}}(\gamma I - A)^{-\mathrm{T}}. \quad (37)$$

The convergence of this iteration is guaranteed if A is Hurwitz, since in this case $\Phi(\gamma)$ has spectral radius less than one for any $\gamma > 0$. This algorithm is known as the Smith algorithm [32]. With zero initial condition $P_0 = 0$, the iteration (37) converges to

$$P = \sum_{i=0}^{\infty}\left[\Phi^i(\gamma)\right]M\left[\Phi^i(\gamma)\right]^{\mathrm{T}} \quad (38)$$

where

$$M = (2\gamma)(\gamma I - A)^{-1}BB^{\mathrm{T}}(\gamma I - A)^{-\mathrm{T}}. \quad (39)$$

Note that the Smith algorithm is rarely used in practical computation due to its slow convergence especially for stiff matrix A, i.e., A has both fast and slow modes, which is typical for many circuit models. Variants of the Smith algorithm lead to the *alternate direction implicit* (ADI) algorithm [30] and the CF-ADI algorithm [24]. Here we use the series representation of P in (38) and Property 3 to justify Observation (b).

Using the identity

$$\Phi(\gamma) = (\gamma I - A)^{-1}(\gamma I + A) = 2\gamma(\gamma I - A)^{-1} - I. \quad (40)$$

one can easily verify that

$$\mathcal{K}_m\left(\Phi(\gamma),\, (\gamma I - A)^{-1}B\right) = \mathcal{K}_m\left((\gamma I - A)^{-1},\, (\gamma I - A)^{-1}B\right).$$

Let V_q $(q \le m)$ be the orthonormal basis matrix of the rational Krylov subspace $\mathcal{K}_m\left((\gamma I - A)^{-1},\, (\gamma I - A)^{-1}B\right)$ and define

$$\hat{\Phi}(\gamma) := (\gamma I - \hat{A})^{-1}(\gamma I + \hat{A}),$$

where $\hat{A}$ is as defined in (31). Then we have the following lemma.

Lemma 3: Given the notations above, the following identities hold

$$\Phi^i(\gamma)(\gamma I - A)^{-1}B = V_q\hat{\Phi}^i(\gamma)(\gamma I - \hat{A})^{-1}\hat{B}. \quad (41)$$

for all $i = 0, \cdots, m-1$, where $\gamma > 0$ is chosen such that the two matrix inversions exist and $\hat{A}$ and $\hat{B}$ are defined in (31).

Proof: The identities (41) follow from the moment matching property 3 and the identity (40). ■

It follows directly from Lemma 3 that the following partial-sum identity holds

$$\sum_{i=0}^{m-1}\left[\Phi^i(\gamma)\right]M\left[\Phi^i(\gamma)\right]^{\mathrm{T}} = V_q\sum_{i=0}^{m-1}\left[\hat{\Phi}^i(\gamma)\right]\hat{M}\left[\hat{\Phi}^i(\gamma)\right]^{\mathrm{T}}V_q^{\mathrm{T}} \quad (42)$$

where M is define in (39) and $\hat{M}$ is defined by

$$\hat{M} = (2\gamma)(\gamma I - \hat{A})^{-1}\hat{B}\hat{B}^{\mathrm{T}}(\gamma I - \hat{A})^{-\mathrm{T}}. \quad (43)$$

Now let $\hat{P}$ be the solution of the reduced order Lyapunov equation

$$\hat{A}\hat{P} + \hat{P}\hat{A}^{\mathrm{T}} + \hat{B}\hat{B}^{\mathrm{T}} = 0.$$

Then according to the Smith algorithm the solution $\hat{P}$ can also be expressed as a series if $\hat{A}$ remains Hurwitz, i.e.

$$\hat{P} = \sum_{i=0}^{\infty}\left[\hat{\Phi}^i(\gamma)\right]\hat{M}\left[\hat{\Phi}^i(\gamma)\right]^{\mathrm{T}}. \quad (44)$$

Going back to our Dominant Subspace Computation Scheme, we can use $V_q\hat{P}V_q^{\mathrm{T}}$ as an approximation to the controllability Gramian P.

The identity (42) is now used to compare the approximation accuracy of $V_q\hat{P}V_q^{\mathrm{T}} \approx P$ by using different γ. If an appropriate γ is chosen so that the convergence of (38) is optimal, then the approximation by using $V_q\hat{P}V_q^{\mathrm{T}}$ for P should be superior if V_q is the basis matrix of the rational Krylov subspace $\mathcal{K}_m\left((\gamma I - A)^{-1},\, (\gamma I - A)^{-1}B\right)$ comparing to other rational Krylov subspaces with other γ. This is because for an optimum γ such that the convergence of (38) is optimal, the partial-sum

$$\sum_{i=0}^{m-1}\left[\Phi^i(\gamma)\right]M\left[\Phi^i(\gamma)\right]^{\mathrm{T}} \quad (45)$$

better dominates the total sum. The partial-sum identity (42) then implies that the subspace dominance is also better captured by the basis matrix V_q if the rational Krylov subspace at an optimum γ is selected in computation. On the other hand, observe that the second type of Krylov subspace $\mathcal{K}_m(A^{-1}, A^{-1}B)$ is the limiting case of a rational Krylov subspace by choosing γ sufficiently small, which is however associated with the *slow* convergence of (38); namely, the dominance captured by the partial-sum (45) for $\gamma \to 0$ is not as good as that by using an optimum γ. Hence this partial-sum matching approach has justified that Observation (b) is in general true.

D. The choice of γ

In the last subsection we argued that an optimum γ would result in an optimal approximation of the dominant subspace by using a real rational Krylov subspace. However, in general it is not easy to find an optimum γ for such a purpose. The similar issue of choosing optimum convergence parameters in ADI-type algorithms has been addressed in many places, see [24] and the references therein. The main technique is to solve a rational minimax problem which requires the information of the eigenvalues of the matrix A. This problem is not completely solved when some eigenvalues of A are complex. Also, when A is high dimensional, finding the eigenvalues or a compact containing region of the eigenvalues of A is also expensive in general. To have a guideline for choosing an appropriate parameter γ, we provide a heuristic method here by following a discretization approach.

Rational Krylov subspace has been widely used in model order reduction. Since moment matching is traditionally formulated as local approximation in the frequency domain, a pure complex $s = j\omega$ or several such points can be used as the shift parameters to improve the approximation accuracy at certain frequency ranges of interest [33]. Also real numbers can be used for shift parameters, see for example [9], [2], which also give good simulation results. However, since moment matching at a real point does not have a direct connection to the local approximation of a frequency response, which is normally along the $j\omega$ axis, the physical meaning of a rational Krylov subspace with *real* shift parameters is not well understood. Grimme made some argument in his thesis for the real shift parameters ([29], Section 6.2.2), but the interpretation is not as clear as that of pure imaginary shift points. It turns out that interpreting a real shift parameter in terms of discretization bears a better physical meaning for practical application.

One way to discretize a continuous-time model

$$\dot{x} = Ax + Bu \tag{46}$$

is to replace the derivative operator (or s operator) approximately by

$$s \approx \frac{z-1}{hz} = \frac{\gamma(z-1)}{z} \tag{47}$$

where $z = e^{sh}$, h is a small time step, and $\gamma = 1/h$ is the sampling frequency. After the discretization, the continuous-time model (46) becomes

$$(\gamma I - A)x_{k+1} = \gamma x_k + Bu_{k+1}, \tag{48}$$

which is the same as the backward Euler integration formula.

Taking z-transform, we obtain $X(z) = F(z)U(z)$, where

$$F(z) = \left[(\gamma I - A)z - \gamma I\right]^{-1} BzU(z). \tag{49}$$

If $(\gamma I - A)$ is invertible, $F(z)$ can be expanded in terms of z^{-1}

$$F(z) = B_\gamma + \gamma A_\gamma B_\gamma z^{-1} + \gamma^2 A_\gamma^2 B_\gamma z^{-2} + \cdots$$

where $A_\gamma = (\gamma I - A)^{-1}$ and $B_\gamma = A_\gamma B$. It is clear that the leading m coefficients in the expansion of $F(z)$ are the matrices that form the rational Krylov subspace $\mathcal{K}_m\left((\gamma I - A)^{-1}, (\gamma I - A)^{-1}B\right)$. If a sequence of different γ_i's are used in the iteration, corresponding to using varying time steps, then a rational Krylov subspace with a set of real shift parameters is formed. Some properties of a rational Krylov subspace with distinct shift parameters are discussed in [24], where the rational Krylov subspaces are obtained by factorizing and reducing the ADI algorithm.

It is interesting to observe that if the transfer functions of two discrete-time models, $Y(z)$ and $\widetilde{Y}(z)$, have moments (i.e. coefficients of z^{-i}) matched for $i = 0, \cdots, m-1$, then by the definition of z-transform we have $y_k = \widetilde{y}_k$, for $k = 0, \cdots, m-1$, in the discrete-time domain. This implies that moment matching using a real rational Krylov subspace bears the meaning of waveform matching in the time-domain, rather than the traditional moment matching in the frequency domain. Strictly speaking, a discretized model is only an approximation of the continuous-time model. Nevertheless, by matching the waveform of an approximate dynamic model, the dominant subspace characteristics is still captured to certain degree. This new interpretation of a real shift parameter γ is expected to provide a heuristic way of choosing an appropriate γ for a reasonably good performance in the dominant subspace approximation. We emphasized that in the current interpretation the parameter γ is the inverse of the time step used for integration or discretization.

It is also interesting to note that the iteration formula for P_k in (37) can be viewed as a consequence of discretization of the differential Lyapunov equation

$$\frac{dP(t)}{dt} = AP + PA^{\mathrm{T}} + BB^{\mathrm{T}}$$

whose steady state solution $P(\infty)$ is the solution of the Lyapunov equation (9). In the same spirit it makes sense to interpret γ as the sampling rate in the discretization.

Although the new interpretation does not give us an optimal choice of γ, it does provide an empirical guideline for choosing an appropriate γ. We recommend to choose a γ several magnitudes smaller than the magnitude of the fastest mode of the full order model. In our experiment we observed that an overly small γ usually resulted in a performance similar to choosing $\gamma = 0$, i.e. moment matching at $s = 0$, while an exceedingly large γ ended up with very bad approximation because clustered sampling could not capture well the dynamic behavior of the waveform. Such observations are quite consistent to our theoretical analysis.

Remark 5: Another discretization is to approximate the s operator by

$$s \approx \frac{2(z-1)}{h(z+1)} = \frac{2\gamma(z-1)}{(z+1)} \tag{50}$$

which is equivalent to the Trapezoidal Rule in numerical integration. It can be verified that the Krylov subspace resulting from this discretization is the same as that from the backward Euler discretization.

V. Numerical experiments

In this section we report comprehensive numerical results. We first introduce three typical measures in Subsection V-A for a complete experimental comparison in the subsequent two subsections. In Subsection V-B we demonstrate several numerical results that are consistent with the theoretical predictions made in Subsection IV-C for the approximation performance of the three types of Krylov subspaces. Then in Subsection V-C numerical examples further demonstrate that the Dominant Subspace Computation Scheme with appropriately chosen Krylov subspaces is very effective for large-scale circuit models. In particular, we demonstrate the convergence effect when a sequence of different dimensions is chosen for intermediate subspaces. Finally, applications of approximately computed dominant subspaces to model order reduction are reported in Subsection V-D. All the computations in this section were carried out using MATLAB 6 for demonstration purpose. When MATLAB becomes inefficient for practical large-scale problems, the free software library SLICOT is recommended for large-scale numerical computation [34].

A. Three measures for comparison

We use three measures to compare how accurately the dominant subspace is approximated. The first measure is defined to compare the approximate singular values with the exact ones. This requires the computation of the exact Gramian and its singular values, thus is for demonstration purpose. In practice we can check the convergence of the computed singular values to determine whether or not the approximate dominant subspace is sufficiently accurate. Let σ_i and $\widetilde{\sigma}_i$ be respectively the exact and approximate singular values for $i = 1, \cdots, m$. The total relative error of singular value approximation is defined by

$$\sigma_{err} = \sum_{i=1}^{m} \frac{|\sigma_i - \widetilde{\sigma}_i|}{\sigma_i} \tag{51}$$

where we assume $\sigma_i > 0$ for $i = 1, \cdots, m$.

The second measure is defined to compute the distance between the approximate subspace basis matrix $\widetilde{U} \in \mathbb{R}^{n\times q}$ and an exact dominant subspace basis matrix $U \in \mathbb{R}^{n\times q}$ obtained from the SVD of P. The distance is defined by

$$\operatorname{dist}(\widetilde{U}, U) := \left\| \widetilde{U} - U(U^{\mathrm{T}}\widetilde{U}) \right\|_F \tag{52}$$

which measures the mismatch between $\widetilde{U}$ and U. The choice of Frobenius norm is to better differentiate the distance for different basis matrices. Note that this measure also requires an exact Gramian, thus is also for demonstration purpose. In practice, one can check the convergence of $\widetilde{U}$ by comparing the distances computed between two consecutive basis matrices.

The third measure is defined to be the residual of the Lyapunov equation, i.e. $R(X)$ defined in (26), by substituting the approximate Gramian formed as in (24). To reduce the computational complexity, we adopt an idea presented in [23] for residual computation. Let $U_q \in \mathbb{R}^{n\times q}$ span the dominant subspace and $\widetilde{P} = U_q P_q U_q^{\mathrm{T}}$ be the approximate solution, where P_q is the solution of the reduced order Lyapunov equation as in (22). Then the residual can be expressed as

$$\begin{aligned} R(\widetilde{P}) &= AU_qP_qU_q^{\mathrm{T}} + U_qP_qU_q^{\mathrm{T}}A^{\mathrm{T}} + BB^{\mathrm{T}} \\ &= \begin{bmatrix} AU_q & U_q & B \end{bmatrix} \begin{bmatrix} 0 & P_q & 0 \\ P_q & 0 & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} U_q^{\mathrm{T}}A^{\mathrm{T}} \\ U_q^{\mathrm{T}} \\ B^{\mathrm{T}} \end{bmatrix}. \end{aligned}$$

Let

$$QR = \begin{bmatrix} AU_q & U_q & B \end{bmatrix}$$

be the QR factorization of the matrix on the right hand side, where R is square and upper-triangular. Then

$$\left\| R(\widetilde{P}) \right\| = \|R\Delta R^{\mathrm{T}}\|$$

where

$$\Delta = \begin{bmatrix} 0 & P_q & 0 \\ P_q & 0 & 0 \\ 0 & 0 & I \end{bmatrix}.$$

Since R is a low-dimensional matrix, this method reduces the computation of residual evaluation significantly. The relative residual is defined to be

$$\varepsilon_{res} = \frac{\left\| R(\widetilde{M}) \right\|}{\|B\|^2}. \tag{53}$$

B. Comparison of approximation without compaction

The first test case is an RLC line with N segments shown in Fig. 1, which could be a discretized transmission line model or an interconnect model. This example will be used to test the approximation effects of the three Krylov subspaces, then the effectiveness of the Dominant Subspace Computation Scheme, and finally model order reduction by using computed dominant subspaces. By the modified nodal analysis (MNA) formulation [35], we use $x = (V_1, \cdots, V_N, I_1, \cdots, I_N)^{\mathrm{T}} \in \mathbb{R}^{2N}$ for the state vector and choose $u = V_s$ and $y = V_N$ to be the input and output, respectively.

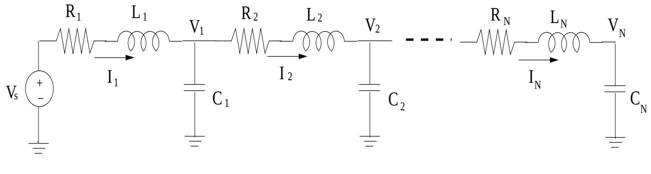


Fig. 1. An RLC line.

This test circuit is first used to demonstrate different approximation performances of the three Krylov subspaces $\mathcal{K}_m\left((\gamma I - A)^{-1}, (\gamma I - A)^{-1}B\right)$, $\mathcal{K}_m\left(A^{-1}, A^{-1}B\right)$, and $\mathcal{K}_m(A, B)$. For easy identification, we identify the first Krylov subspace by the specific γ chosen, the second one by $\gamma = 0$, and the third one by $\gamma = \infty$. We chose a 200-stage RLC line so that the model order is 400, and the exact Gramian is solved by Bartels-Stewart algorithm [36]. For simplicity, we assume that the RLC values are uniform with $R = 10\,\Omega$, $L_i = 1\,H$, and $C_i = 1\,F$ for all i. Although the state space model is controllable in theory, the Gramian computed from MATLAB has only a rank 31, which means that the controllable space has prominent low-dimensional dominance and the full order model has quite much redundancy.

TABLE I
MEASURES BY USING THREE KRYLOV SUBSPACES.

	$\gamma = \infty$	$\gamma = 0$	$\gamma = 0.002$
σ_{err}	15.69	10.54	8.26
$d(U, \widetilde{U})$	3.29	2.34	2.02
ε_{res}	0.002634	0.009446	0.005470

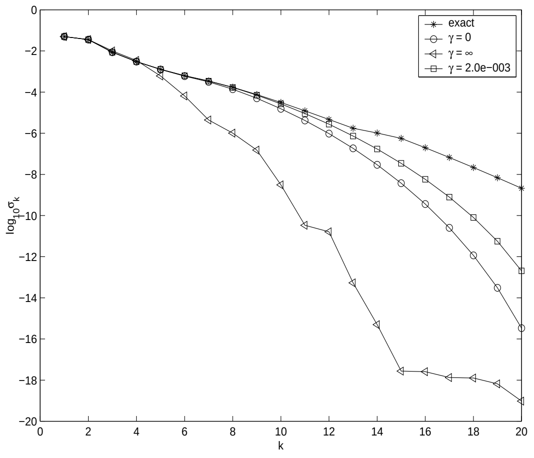


Fig. 2. Approximation of singular values by using $\gamma = \infty$ ($\triangleright$), $\gamma = 0$ ($\circ$), and $\gamma = 2 \times 10^{-3}$ ($\square$).

We compute the approximate dominant controllable subspace by using the three candidate Krylov subspaces, all with order $q = 20$. To verify the different approximation effects as predicted by the two observations (a) and (b) in Section IV-C, the *compaction* procedure in the Dominant Subspace Computation Scheme is not used for this test case. The three measures introduced in the previous subsection are computed and listed in Table I. Shown in Fig. 2 is the approximation of the singular values by using the three optional Krylov subspaces. For better visualization, the 10-based logarithms of the singular values are plotted. We see that the Krylov subspace at $\gamma = 0$ is better than that at $\gamma = \infty$ by the measures of singular values and subspace distance. For this example, the best approximation is achieved by a rational Krylov subspace at $\gamma = 0.002$. The numerical result matches very well with our previous theoretical analysis. In terms of residual, the Krylov subspace at $\gamma = \infty$ looks the best, which indicates that the residual measure might not be reliable if used as the only measure for dominant subspace approximation. Note that one can also use the equivalent Lyapunov equation (27) to compute the residual, which in fact resulted in a better residual in our experiment for the case $\gamma = 0$. But to have a comparison on the same basis, we keep on using the Lyapunov equation (9) for the residual comparison.

TABLE II
MEASURES BY USING THREE KRYLOV SUBSPACES.

	$\gamma = \infty$	$\gamma = 0$	$\gamma = 3 \times 10^8$
σ_{err}	15.72	11.20	11.00
$d(U, \widetilde{U})$	3.24	2.40	1.99
ε_{res}	0.066	0.321	0.020

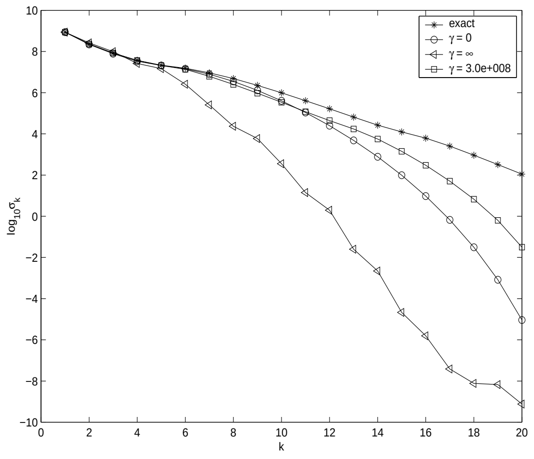


Fig. 3. Approximation of singular values by using $\gamma = \infty$ ($\triangleright$), $\gamma = 0$ ($\circ$), and $\gamma = 3 \times 10^8$ ($\square$).

Next we choose another set of RLC values for the same circuit model, with $R = 20\,\Omega$, $L = 1\,nH$ and $C = 20\,pF$ to change the model modes but without changing the model order. The test results are summarized in Table II and Fig. 3 where the compaction procedure is again not used. For this case, the exact Gramian has a computed rank of 33. We see that the Krylov subspace at $\gamma = 0$ still better captures the dominant singular values than that at $\gamma = \infty$, while the Krylov subspace at $\gamma = 3 \times 10^8$ performs superior among all. We observe that this set of circuit element values has driven the fastest mode to the GHz oscillation level, hence a larger γ is chosen here.

The preceding two test cases both demonstrate the consistency with the analytical results established in Section IV-C. Moreover, they show that all three Krylov subspaces can capture the leading several dominant singular values very well, but not for the trailing singular values. To improve the overall approximation accuracy, the compaction procedure in the *Dominant Subspace Computation Scheme* can be used.

C. Convergence of the compaction procedure

By the compaction procedure we need to choose an intermediate order m slightly larger than q, the dimension of the dominant subspace, then use the steps in the Dominant

Subspace Computation Scheme to obtain a better dominance approximation. The test cases in this subsection are used to demonstrate the convergence effect by using a sequence of intermediate orders m.

The same circuit example in Fig. 1 is used again but with a new set of uniform RLC values $R = 30\,\Omega$, $L = 0.1\,nH$, and $C = 5\,pF$. We also choose a larger model order with $n = 2000$. For such a large scale model, solving the exact Gramian is not feasible. Hence we compute the distance between two consecutive subspaces and the residual of Lyapunov equation as the measures for comparison. We also plot the computed singular values for visualization of the convergence. To test the convergence, we choose a sequence of m from 20 up to 100 with increment 20, denoted in MATLAB as $m = 20 : 20 : 100$. All intermediate subspaces are compacted to dimension $q = 20$. Listed in Table III are the distance measure and the residual measure for different m but with the same $\gamma = 10^8$ for the shift parameter. The notation $d(U, U_{pre})$ denotes the distance between the subspaces computed at two consecutive m's, with U for the current m and U_{pre} for the previous m, where both U and U_{pre} are basis matrices in $\mathbb{R}^{n\times q}$. The initial subspace is assumed to be the zero subspace. The data in Table III show that the dominant subspace has well converged. Shown in Fig. 4 is the convergence behavior of the computed singular values.

For comparison, the test results by using $\gamma = 0$ are shown in Table IV and Fig. 5. We see from the table that $d(U, U_{pre}) = 1.00$ at $m = 100$, which means the convergence is not as good as that of $\gamma = 10^8$.

Finally the the same example is tested again to see the performance of the Krylov subspace at $\gamma = \infty$. The results are shown in Table V and Fig. 6. The convergence of this option is clearly worse than the previous two, which again demonstrates the different approximation performance of the three Krylov subspaces as predicted.

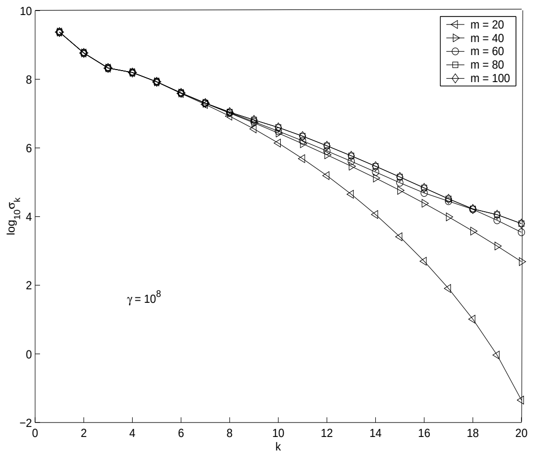


Fig. 4. Convergence of singular values with $\gamma = 10^8$ for $m = 20 : 20 : 100$.

To summarize, the dominant subspace approximated by the Krylov subspace $\mathcal{K}_m(A, B)$ is always the worst comparing to the other two options. For this reason, in the application to model order reduction to be presented next, the Krylov subspace $\mathcal{K}_m(A, B)$ will not be used.

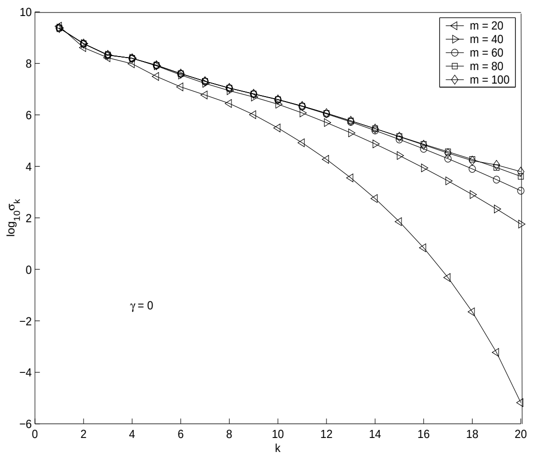


Fig. 5. Convergence of singular values with $\gamma = 0$ for $m = 20 : 20 : 100$.

TABLE V

MEASURES AS A FUNCTION OF m FOR $\gamma = \infty$.

m	20	40	60	80	100
$d(U, U_{pre})$	4.47	2.48	1.78	1.46	1.27
ε_{res}	0.0176	0.0064	0.0035	0.0023	0.0016

D. Application to model order reduction

The RLC circuit in Fig. 1 is now used for testing model order reduction. This time node V_1 is chosen as the output and the uniform RLC values are $R = 20\,\Omega$, $L = 1\,nH$ and $C = 20\,pF$. The full model order is 1000 and the reduced model order is 20. For dominant subspace compaction, we first generate a Krylov subspace with an intermediate order 80 which is further reduced to order 20 by compaction. Shown in Fig. 7 is the reduction result by using the Krylov subspace at $\gamma = 0$ and the projection to the dominant *controllable* subspace V_q (i.e. choosing $W_q = V_q$ in (8)). Shown in Fig. 8 is the reduction result by using the dominant *observable* subspace W_q also computed at $\gamma = 0$ (i.e., choosing $V_q = W_q$ in (8)). The accuracy of the two reduced models are comparable.

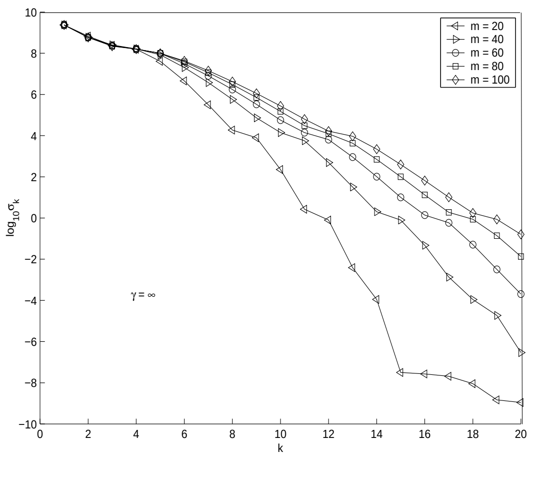


Fig. 6. Convergence of singular values with $\gamma = \infty$ for $m = 20 : 20 : 100$.

TABLE III

MEASURES AS A FUNCTION OF m FOR $\gamma = 10^8$.

m	20	40	60	80	100
$d(U, U_{pre})$	4.47	2.12	1.49	1.15	0.10
ε_{res}	0.0354	0.0102	0.0054	4.23×10^{-4}	6.0×10^{-5}

TABLE IV

MEASURES AS A FUNCTION OF m FOR $\gamma = 0$.

m	20	40	60	80	100
$d(U, U_{pre})$	4.47	2.38	1.68	1.30	1.00
ε_{res}	0.2343	0.0764	0.0152	8.28×10^{-3}	5.15×10^{-4}

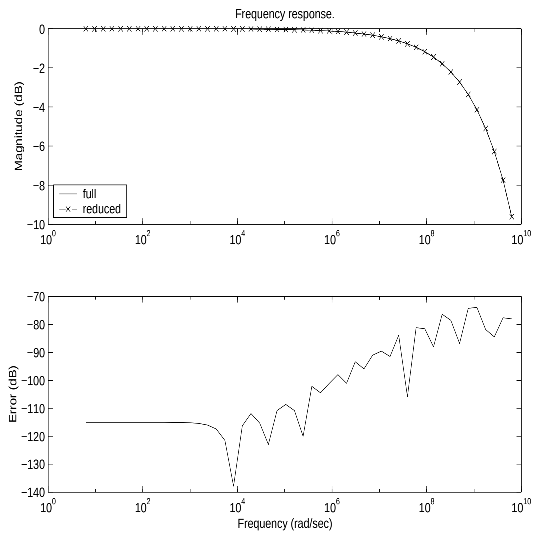


Fig. 7. Reduction of RLC line from 1000th order to 20th order using dominant controllable subspace computed at $\gamma = 0$.

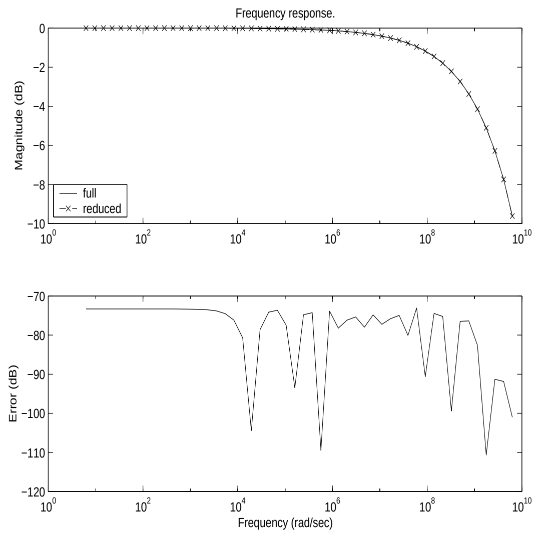


Fig. 8. Reduction of RLC line from 1000th order to 20th order, using dominant observable subspace computed at $\gamma = 0$.

The reduction result becomes better if the oblique projection is performed with V_q computed from the pair (A, B) and W_q from the pair $(A^{\mathrm{T}}, C^{\mathrm{T}})$, as shown in Fig. 9, where still $\gamma = 0$ is used. Note that the error in this case is significantly smaller than the previous two. For comparison, the reduction result by using moment matching up to the 20th order without compaction is shown in Fig. 10. We see that by matching moments only at the low frequency the error, although is very small at the low frequency band, increases remarkably at the high frequency. Comparatively, the errors from using dominant controllable/observable subspaces are fairly flat.

The second circuit example used for model order reduction is the two coupled RLC lines shown in Fig. 11. For this circuit

$$x = (V_{11}, \cdots, V_{1N}, V_{21}, \cdots, V_{2N}, \\ I_{11}, \cdots, I_{1N}, I_{21}, \cdots, I_{2N})^{\mathrm{T}} \in \mathbb{R}^{4N}$$

is the state vector and $u = V_s$ is the input. The output will be specified later. The uniform RLC values are chosen as $R_{1i} = 10\,\Omega$, $R_{2i} = 5\,\Omega$, $L_{1i} = L_{2i} = 10.0\,nH$, $C_{1i} = C_{2i} = 1.0\,pF$, $CC_i = 20\,pF$ for $i = 1, \cdots, N$, where N is the number of stages. We choose $N = 300$ stages so that the model is of 1200th order. This circuit is used to test the

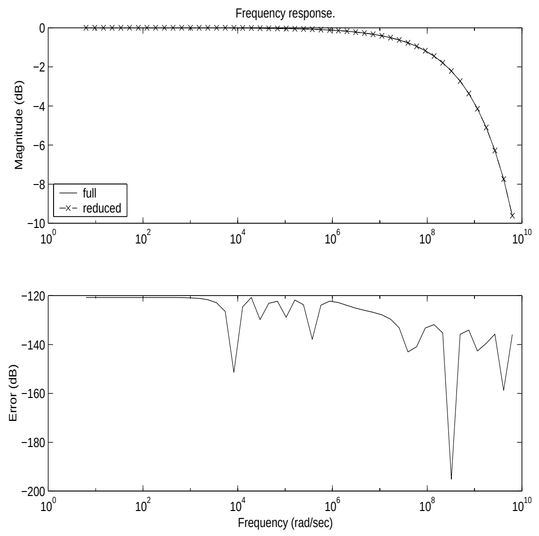


Fig. 9. Reduction of RLC line from 1000th order to 20th order using both dominant subspaces computed at $\gamma = 0$ (oblique projection).

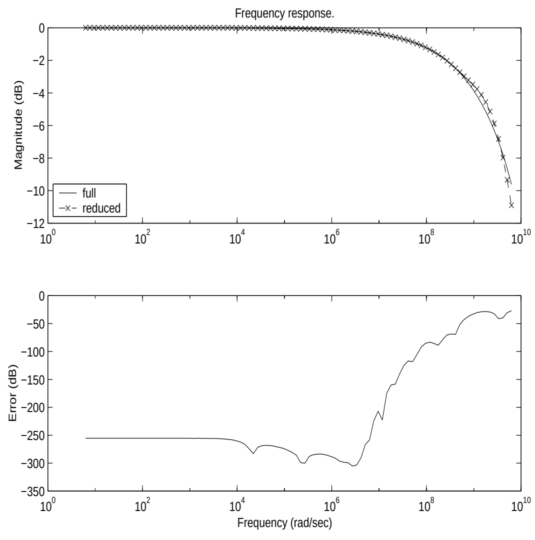


Fig. 10. Reduction of RLC line from 1000th order to 20th order by moment matching at $s = 0$ (without compaction).

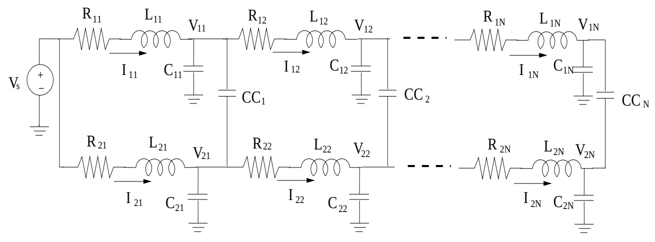


Fig. 11. Two coupled RLC lines.

reduction effect by using V_q only, W_q only, and both V_q and W_q. The first two belong to orthogonal projection and the third belongs to oblique projection. It usually happens that if the circuit model is not in the port formulation, the passivity cannot be preserved even using the congruent transformation. This example demonstrates that in case the reduced matrix $\hat{A}$ has unstable poles, removing those few unstable poles by further projecting $\hat{A}$ to the stable subspace would not lose much accuracy for practical application. Nevertheless, it is important to be aware that this simple technique only works for high order models.

First we choose $y = V_{1N}$ as the output and compute the dominant controllable and observable subspaces using the Dominant Subspace Computation Scheme. We use a shift parameter $\gamma = 2 \times 10^8$ and choose $m = 100$ for the intermediate order and $q = 40$ for the reduced model order. The computed basis matrices for the controllable and observable subspaces are first used separately for model order reduction and the results are shown in Figs. 12 and 13, respectively. Note that in the case of controllable subspace 8 poles of the reduced matrix are unstable, they are removed by further reducing the model to order 32 after projection to the stable subspace. Also in the case of observable subspace 6 poles of the reduced matrix are unstable, they are removed by further reducing the model to order 34. The plots in Figs. 12 and 13 are results with the unstable poles removed, but the accuracy up to a high frequency point remains very good. The low accuracy at

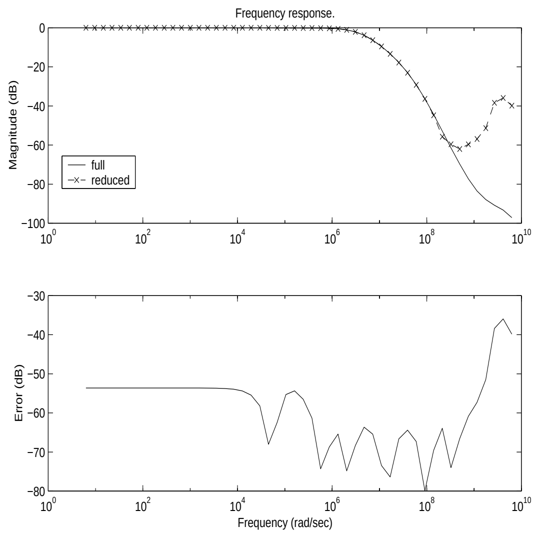


Fig. 12. Reduction of the coupled RLC line from 1200th order to 40th order using dominant controllable subspace computed at $\gamma = 2 \times 10^8$.

the high frequency part is probably due to the low reduced order, which in general can be improved by choosing a higher reduced order. Since for this example the high frequency part is the significantly roll-off band, the error at that band normally would not cause a serious problem for simulation not reaching that frequency band. Also we see that both the reduced models by using V_q or W_q have comparable accuracy, implying that in practice there is no reason to favor one from the other for model order reduction.

Since the outer product of the computed two basis matrices $W_q^{\mathrm{T}} V_q$ is singular for this example, they cannot be used for oblique projection. This is usually caused by the fact that the input/output matrices B and C are inherently orthogonal. In such cases only the orthogonal projections as done above are feasible. However, if we choose a different output, say, $y = I_{11} + I_{21}$, then W_q and V_q have a nonsingular outer product, hence can be used for oblique projection. The reduction result is shown in Fig. 14. In this reduction, two out of the 40 poles are unstable. Hence the model is further reduced to order 38 by a projection to the stable subspace. We can see again from Fig. 14 that the reduction remains quite accurate.

VI. Conclusion

Large-scale models appearing in circuit simulation and other areas bring challenges to conventional model order reduction techniques. To overcome the limitation of balanced truncation for large-scale models, approximate dominant subspaces have been used for practical model order reduction with many good results. However, a theoretical error bound for unbalanced dominant subspace projection was unknown. This paper has established an $\mathcal{L}_2$ error bound for model order reduction using unbalanced dominant subspace. Furthermore, it has investigated the performances of using three types of Krylov subspaces for approximate dominant subspace computation. It is analytically justified that the conventionally used Krylov subspace $\mathcal{K}_m(A, B)$ in fact is not a good candidate for

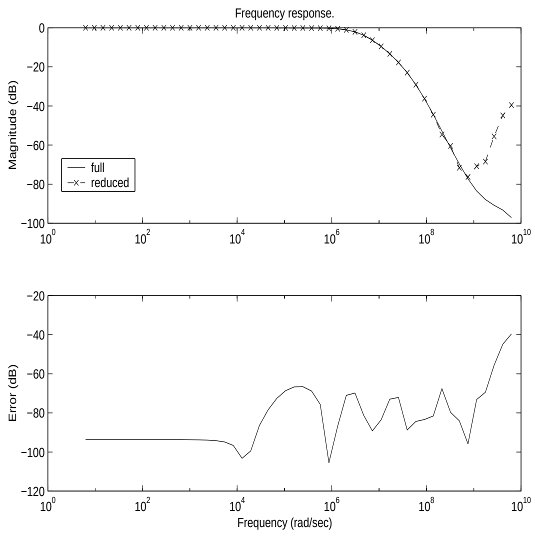


Fig. 13. Reduction of the coupled RLC line from 1200th order to 40th order using dominant observable subspace computed at $\gamma = 2 \times 10^8$.

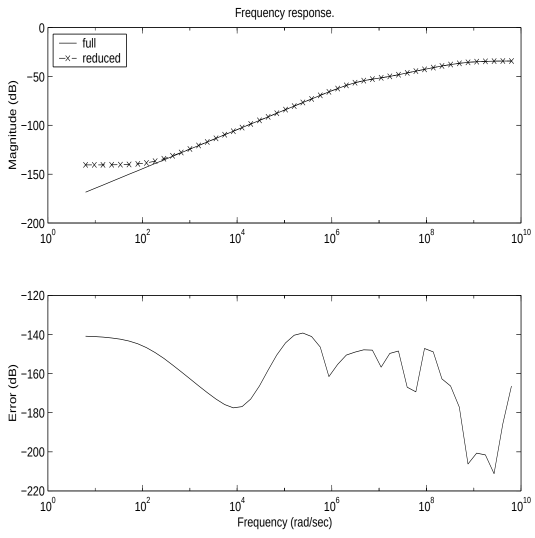


Fig. 14. Reduction of the coupled RLC line from 1200th order to 40th order using both dominant subspaces computed at $\gamma = 2 \times 10^8$ (oblique projection).

effective dominant subspace computation. Rather the Krylov subspace that matches moments at low frequencies and a rational Krylov subspace with an appropriately chosen real shift parameter are capable of producing superior approximate dominant subspaces. Numerical experiments have demonstrated that the theoretical analysis well predicts the computation results. Furthermore, numerical results have demonstrated that the approximate dominant subspaces computed from the Dominant Subspace Computation Scheme can be used effectively for large-scale model order reduction.

VII. Acknowledgement

The authors are grateful to the reviewers for their constructive comments. In particular, the authors would like to thank one reviewer for bringing the reference [24] to their attention.

Appendix

Proof of Property 2: We only prove the Property for the single-input case. Its extension to the multiple-input case is straightforward.

The Arnoldi algorithm gives rise to the identity

$$A^{-1}V_m = V_m H + h v_{m+1} e_m^{\mathrm{T}} \tag{54}$$

where H is an upper Hessenberg matrix, v_{m+1} is the $(m+1)$th basis vector in the Arnoldi algorithm, and $h \geq 0$ is the normalization factor. By the fact that H is upper Hessenberg, it is readily verified that

$$A^{-i}V_m e_1 = V_m H^i e_1, \qquad \text{for } i = 1, \cdots, m. \tag{55}$$

It also follows from (54) that

$$I_m = V_m^{\mathrm{T}} A V_m H + h V_m^{\mathrm{T}} A v_{m+1} e_m^{\mathrm{T}}$$

i.e.

$$\hat{A}^{-1} = H + h\hat{A}^{-1} V_m^{\mathrm{T}} A v_{m+1} e_m^{\mathrm{T}}. \tag{56}$$

One can show by induction that

$$\hat{A}^{-i} e_1 = H^i e_1, \qquad \text{for } i = 1, \cdots, m-1. \tag{57}$$

Indeed, eqn. (56) implies that $\hat{A}^{-1}e_1 = He_1$. Assume that (57) holds for some i satisfying $1 \leq i \leq m-2$. Then

$$\begin{aligned} \hat{A}^{-(i+1)} e_1 &= \hat{A}^{-1} H^i e_1 \\ &= \left(H + h\hat{A}^{-1} V_m^{\mathrm{T}} A v_{m+1} e_m^{\mathrm{T}} \right) H^i e_1 \\ &= H^{i+1} e_1 \end{aligned}$$

because of the fact that H is an upper Hessenberg matrix and $e_m^{\mathrm{T}} H^i e_1 = 0$ for $1 \leq i \leq m-2$.

Let $\beta = \|A^{-1}B\|$. Then $A^{-1}B = \beta V_m e_1$, i.e. $B = \beta A V_m e_1$, from which we obtain $\hat{B} = \beta \hat{A} e_1$, i.e.

$$\hat{A}^{-1}\hat{B} = \beta e_1. \tag{58}$$

Thus we have shown $A^{-1}B = V_m \hat{A}^{-1}\hat{B}$ which is the identity (33) for $i = 1$. Applying the identities (57), we obtain for $2 \leq i \leq m$

$$\begin{aligned} A^{-i}B &= A^{-(i-1)} A^{-1} B \\ &= \beta A^{-(i-1)} V_m e_1 = \beta V_m H^{i-1} e_1 \\ &= \beta V_m \hat{A}^{-(i-1)} e_1 \\ &= V_m \hat{A}^{-i} \hat{B} \end{aligned}$$

where we also used (54) and (58). Consequently, the identities (33) hold for $i = 1, \cdots, m$. ■

References

[1] A. Antoulas, D. Sorensen, and S. Gugercin, "A survey of model reduction methods for large-scale systems," in *Structured Matrix in Operator Theory, Numerical Analysis, Control, Signal and Image Processing,* Contemporary Mathematics, AMS Publications, 2001.

[2] Z. Bai, "Krylov subspace techniques for reduced-order modeling of large-scale dynamical systems," *Applied Numerical Mathematics*, vol. 43, pp. 9–44, 2002.

[3] S. Gugercin and A. Antoulas, "A comparative study of 7 algorithms for model reduction," in *Proc. 39th IEEE Conf. on Decision and Control,* (Sydney, Australia), pp. 2367–2372, 2000.

[4] B. Moore, “Principal component analysis in linear systems: controllability, observability, and model reduction,” *IEEE Trans. on Automatic Control*, vol. AC-26, no. 1, pp. 17–32, February 1981.

[5] K. Glover, “All optimal Hankel-norm approximations of linear multivariable systems and their L_∞ error bounds,” *International Journal of Control*, vol. 39, no. 6, pp. 1115–1193, 1984.

[6] M. Safonov and R. Chiang, “A Schur method for balanced-truncation model reduction,” *IEEE Trans. on Automatic Control*, vol. AC-34, no. 7, pp. 729–733, July 1989.

[7] C. Villemagne and R. Skelton, “Model reduction using a projection formulation,” *Int. J. Control*, vol. 46, pp. 2141–2169, 1987.

[8] L. Pillage and R. Rohrer, “Asymptotic waveform evaluation for timing analysis,” *IEEE Trans. on Computer-Aided Design*, vol. 9, pp. 352–366, April 1990.

[9] P. Feldmann and R. Freund, “Efficient linear circuit analysis by Padé approximation via the Lanczos process,” *IEEE Trans. on Computer-Aided Design*, vol. 14, no. 5, pp. 639–649, May 1995.

[10] A. Odabasioglu, M. Celik, and L. Pileggi, “PRIMA: Passive reduced-order interconnect macromodeling algorithm,” *IEEE Trans. on Computer-Aided Design*, vol. 17, no. 8, pp. 645–654, August 1998.

[11] E. Grimme, D. Sorensen, and P. Dooren, “Model reduction of state space systems via an implicitly restarted Lanczos method,” *Numerical Algorithms*, vol. 12, pp. 1–31, 1996.

[12] K. Gallivan and E. Grimme, “A rational Lanczos algorithm for model reduction,” *Numerical Algorithms*, vol. 12, pp. 33–63, 1996.

[13] I. Jaimoukha and E. Kasenally, “Implicitly restarted Krylov subspace methods for stable partial realizations,” *SIAM J. Matrix Anal. Appl.*, vol. 18, no. 3, pp. 633–652, 1997.

[14] V. Balakrishnan, Q. Su, and C.-K. Koh, “Efficient balance-and-truncate model reduction for large-scale systems,” in *Proc. 2001 IEEE American Control Conference*, (Arlington, VA), pp. 4746–4751, 2001.

[15] T. Gudmundsson and A. Laub, “Approximate solution of large spase Lyapunov equations,” *IEEE Trans. on Automatic Control*, vol. 39, no. 5, pp. 1110–1114, 1994.

[16] J.-R. Li, F. Wang, and J. White, “An efficient Lyapunov equation-based approach for generating reduced-order models of interconnect,” in *Proc. 36th ACM/IEEE Design Automation Conference*, (New Orleans, LA), 1999.

[17] P. Rabiei and M. Pedram, “Model order reduction of large circuits using balanced truncation,” in *IEEE Proc. Asia South Pacific Design Automation Conference (ASPDAC)*, pp. 237–240, Feb. 1999.

[18] S. Lall, J. Marsden, and S. Glavaski, “A subspace approach to balanced truncation for model reduction of nonlinear control systems,” *Int. J. on Robust and Nonlinear Control*, vol. 12, no. 5, pp. 519–535, 2002.

[19] A. Hodel and K. Poolla, “Heuristic approaches to the solution of very large sparse Lyapunov and algebraic Riccati equations,” in *Proc. 27th IEEE Conf. on Decision and Control*, (Austin, TX), pp. 2217–2222, 1988.

[20] Y. Saad, “Numerical solution of large Lyapunov equations,” in *Signal Processing, Scattering, Operator Theory and Numerical Methods* (M. Kaashoek, J. Schuppen, and A. Ran, eds.), pp. 503–511, Boston, MA: Birkhäuser, 1990.

[21] I. Jaimoukha and E. Kasenally, “Krylov subspace methods for solving large Lyapunov equations,” *SIAM J. Numer. Anal.*, vol. 31, no. 1, pp. 227–251, 1994.

[22] A. Hodel, B. Tenison, and K. Poolla, “Numerical solution of the Lyapunov equation by approximate power iteration,” *Linear Algebra Appl.*, vol. 236, pp. 205–230, 1996.

[23] T. Penzl, “A cyclic low-rank Smith method for large sparse Lyapunov equations,” *SIAM J. Scientific Computing*, vol. 21, no. 4, pp. 1401–1418, 2000.

[24] J.-R. Li and J. White, “Low rank solution of Lyapunov equations,” *SIAM J. Matrix Analysis and Applications*, vol. 24, no. 1, pp. 260–280, 2002.

[25] V. Klema and A. Laub, “The singular value decomposition: its computation and some applications,” *IEEE Trans. on Automatic Control*, vol. AC-25, no. 2, pp. 164–176, April 1980.

[26] D. Boley and G. Golub, “The Lanczos-Arnoldi algorithm and controllability,” *Systems & Control Letters*, vol. 4, pp. 317–324, 1984.

[27] K. Zhou, *Essentials of Robust Control*. Upper Saddle River, New Jersey: Prentice Hall, 1998.

[28] Z. Bai, R. Slone, W. Smith, and Q. Ye, “Error bound for reduced system model by Padé approximation via the Lanczos process,” *IEEE Trans. on Computer-Aided Design*, vol. 18, no. 2, pp. 133–141, February 1995.

[29] E. Grimme, *Krylov Projection Methods for Model Reduction*. PhD thesis, ECE Dept., University of Illinoise at Urbana-Champaign, 1997.

[30] A. Lu and E. Wachspress, “Solution of Lyapunov equations by ADI iteration,” *Computers and Mathematics with Applications*, vol. 21, no. 9, pp. 43–58, 1991.

[31] A. Greenbaum, *Iterative Methods for Solving Linear Systems*. Philadelphia: SIAM, 1997.

[32] R. Smith, “Matrix equation $XA + BX = C$,” *SIAM J. Appl. Math.*, vol. 16, pp. 198–201, 1968.

[33] E. Chiprout and M. Nakhla, “Analysis of interconnect networks using complex frequency hopping (CFH),” *IEEE Trans. on Computer-Aided Design*, vol. 14, no. 2, pp. 186–200, February 1995.

[34] A. Varga, “Model reduction software in the SLICOT library,” in *Proc. IEEE Int’l Symp. on Computer Aided Control System Design (CACSD)*, (Anchorage, Alaska), pp. 183–198, 2000.

[35] C.-W. Ho, A. E. Ruehli, and P. A. Brennan, “The modified nodal approach to network analysis,” *IEEE Trans. on Circuits and Systems*, vol. CAS-22, no. 6, pp. 504–509, 1975.

[36] R. Bartels and G. Stewart, “Algorithm 432: Solution of the matrix equation $AX + XB = C$,” *Commun. ACM*, vol. 15, pp. 820–826, 1972.