AFRL-SN-WP-TR-1998-1105

DEVELOPMENT OF CAD TOOLS FOR POWER ESTIMATION IN CONTINUOUS-TIME AND SWITCHED-CAPACITOR ANALOG CIRCUITS

DR. GIORGIO CASINOVI

GEORGIA TECH RESEARCH CORPORATION
GEORGIA INSTITUTE OF TECHNOLOGY
CENTENNIAL RESEARCH BUILDING
ATLANTA, GA 30332-0420

SEPTEMBER 1998

FINAL REPORT FOR 03/15/1995 – 06/14/1998

SENSORS DIRECTORATE
AIR FORCE RESEARCH LABORATORY
AIR FORCE MATERIEL COMMAND
WRIGHT-PATTERSON AIR FORCE BASE OH 45433-7318

NOTICE

USING GOVERNMENT DRAWINGS, SPECIFICATIONS, OR OTHER DATA INCLUDED IN THIS DOCUMENT FOR ANY PURPOSE OTHER THAN GOVERNMENT PROCUREMENT DOES NOT IN ANY WAY OBLIGATE THE US GOVERNMENT. THE FACT THAT THE GOVERNMENT FORMULATED OR SUPPLIED THE DRAWINGS, SPECIFICATIONS, OR OTHER DATA DOES NOT LICENSE THE HOLDER OR ANY OTHER PERSON OR CORPORATION; OR CONVEY ANY RIGHTS OR PERMISSION TO MANUFACTURE, USE, OR SELL ANY PATENTED INVENTION THAT MAY RELATE TO THEM.

THIS REPORT IS RELEASABLE TO THE NATIONAL TECHNICAL INFORMATION SERVICE (NTIS). AT NTIS, IT WILL BE AVAILABLE TO THE GENERAL PUBLIC, INCLUDING FOREIGN NATIONS.

THIS TECHNICAL REPORT HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION.

Charles L. A. Cerny, Ph.D.
Electronics Engineer

Mr. Kenichi Nakano
Chief, Electron Devices Branch

for
Mr. Robert T. Kemerley
Chief, Aerospace Components and
Subsystems Division

Do not return copies of this report unless contractual obligations or notice on a specific document requires its return.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	SEPTEMBER 1998	FINAL REPORT FOR 03/15/1995 - 06/14/1998

4. TITLE AND SUBTITLE
DEVELOPMENT OF CAD TOOLS FOR POWER ESTIMATION IN CONTINUOUS-TIME AND SWITCHED-CAPACITOR ANALOG CIRCUITS

5. FUNDING NUMBERS
C F33615-95-C-1624
PE 61101
PR C506
TA 02
WU 04

6. AUTHOR(S)
DR. GIORGIO CASINOVI

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
GEORGIA TECH RESEARCH CORPORATION
GEORGIA INSTITUTE OF TECHNOLOGY
CENTENNIAL RESEARCH BUILDING
ATLANTA, GA 30332-0420

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
SENSORS DIRECTORATE
AIR FORCE RESEARCH LABORATORY
AIR FORCE MATERIEL COMMAND
WRIGHT-PATTERSON AFB, OH 45433-7318
POC: DR. CHARLES CERNY, AFRL/SNDD, 937-255-1874 EXT. 3464

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-SN-WP-TR-1998-1105

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The goal of this research contract was the development of CAD tools to compute statistical information (in particular, mean and variance) about the power dissipated in a continuous-time or switched-capacitor circuit, when statistical information about the input signals to the circuit is given. This approach mirrors the one generally used to estimate power dissipation in digital CMOS circuits. Two power estimation CAD programs were developed: one for continuous-time analog circuits, the other for switched-capacitor circuits. Those tools were integrated into an existing commercial CAD environment, namely CADENCE's Design Framework II.

14. SUBJECT TERMS

15. NUMBER OF PAGES
64

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAR

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Project Summary

1.1.1 Scope

The scope of this research contract was the development of CAD tools that can compute statistical information (in particular, mean and variance) about the power dissipated in a continuous-time or switched-capacitor circuit, when the inputs to the circuit belong to a class of signals whose statistics are given. Specifically, two power estimation CAD programs were developed: one for continuous-time analog circuits, the other for switched-capacitor circuits. Those tools were integrated into an existing commercial CAD environment, namely CADENCE's Design Framework II.

1.1.2 Research into Power Estimation Algorithms

This task involved the theoretical study of the algorithms on which the power estimation tools rely. This task developed a sound mathematical theory of power estimation in continuous-time and switched-capacitor circuits. Based on this theory, it was possible to establish a confidence level in the power dissipation estimates given by the developed tools.

1.1.3 Development of Power Estimation Tools

This task involved the development of software based on the algorithms studied as a part of the previous task. This software consists of a CAD program for power estimation in continuous-time analog circuits, and another program for power estimation in switched-capacitor circuits. Those tools were integrated into an existing commercial CAD environment, CADENCE's Design Framework II. CADENCE has developed a set of tools, called Open Simulation System, to encapsulate third-party CAD tools into their design environment.

In order to verify the performance of the developed CAD tools, a set of benchmark circuits was collected. Care was taken to ensure that this set includes circuits that are representative of actual industrial designs.

1.1.4 Generation of Documentation, Papers and Reports

Use of the software will be illustrated later in this report on examples taken from the set of collected benchmark circuits. The theoretical and experimental results obtained in the course of the proposed research effort were also presented in journal and conference papers. The results attained by this research project are summarized in this final report.

1.2 Innovative Claims

Most (if not all) of the published work on CAD tools for power estimation concerns itself only with digital circuits. However, many emerging product technologies, such as personal computing and communication systems, involve mixed signal processing. The goal of the proposed research was to develop power estimation CAD tools for continuous-time and switching analog systems. Note that ordinary circuit simulators (such as SPICE) are not suited for that task, because they can only compute the power dissipated by the circuit *for one specific set of input signals.* In contrast, the goal of the research effort described here was to develop CAD tools that can compute statistical information (in particular, mean and variance) about the power dissipated in a continuous-time or switched-capacitor circuit *for a variety of input signals* (whose statistics are given). This approach mirrors the one generally used to estimate power dissipation in digital CMOS circuits. This approach has the following distinguishing features:

No restrictions on the type of circuit: The methods used to estimate power dissipation are completely general, and they do not rely on any particular assumptions about the type of circuit under consideration (other than it contain **no distributed elements**). As such, this methodology can be applied to a large class of circuits, including continuous-time and switched-capacitor filters, RF mixers and amplifiers, phase-locked loops, and so on.

Arbitrary granularity: The method used computes at once statistics for all voltages and currents in the circuit. As a consequence, power dissipation can be estimated with the same confidence at any desired granularity, down to the component level, if necessary. Moreover, an increase in the granularity of the estimation has only a marginal effect on the total computational effort.

No Monte-Carlo analyses required: The proposed algorithm gives the statistics of all voltages and currents of interest as the solutions of certain sets of *linear, algebraic* equations (even if the circuit in question is nonlinear and not memoryless). The only operations involved are numerical solution of linear equations (e.g. by Gaussian elimination) and numerical quadrature (to evaluate some of the coefficients of the equations to be solved). Therefore, no Monte-Carlo analyses or time-domain simulations of the circuit involved are required. Note that a Monte-Carlo approach to power estimation in analog circuits would be extremely computationally intensive, because collecting each sample would require a time-domain simulation of the circuit involved.

No need to assume uncorrelated inputs: Many of the published power estimation algorithms for digital circuits make the assumption that the primary inputs to the circuit are statistically independent (e.g., [1]). In some cases, this may be an unrealistic assumption (e.g., when the inputs are signals generated by optical sensors which are part of an optical array). Algorithms that do not rely on the uncorrelation assumption are significantly more expensive in terms of computational effort required [2]. In contrast, the algorithms used by our power estimation tools allow the primary inputs to the circuit to be correlated without affecting the overall computational cost.

No memoryless assumption: The vast majority of power estimation algorithms for digital circuits assume instantaneous transitions. While this approximation is acceptable for the purpose of power estimation when dealing with digital or switched-capacitor circuits, it is clearly no longer so in the case of continuous-time analog circuits. The algorithm used by our power estimation tools takes the dynamics of the circuit into full account.

Integration with other tools: CADENCE's Open Simulation System is a set of tools to encapsulate third-party analysis tools into CADENCE's Design Framework II environment. In particular, the SKILL language allows external access to the databases used by other CADENCE tools. Our power estimation tools were integrated into CADENCE's Design Framework II. In this way, it is possible to use them in conjunction with other tools, such as power estimators for digital circuit, thus enabling designers to estimate power consumption in complex mixed-signal systems.

Chapter 2

Development of Power Estimation Tools

2.1 Choice of HDL

Cadence's Design Framework II serves as a front-end to the power estimation tools developed as a result of this research project. For this purpose, it was necessary to develop a suitable interface between our tools and the Cadence environment. It was determined that the best way to achieve this goal is to exploit Cadence's Open Simulation System (OSS), which allows the integration of third-party CAD tools into the Cadence system. OSS provides a user interface for controlling the execution of simulation, the generation of netlists and input vectors, and the display of the simulation results which is consistent with the interface used by the other tools in the Design Framework II environment. Specifically, OSS provides two tools to generate netlists in textual format: a Hierarchical NetLister (HNL) and a Flat NetLister (FNL). Both netlisters are capable of producing output in a user-defined format, described using the Cadence standard language, SKILL.

The netlist format depends upon the choice of the Hardware Description Language to be used. Several factors must be considered, such as portability, popularity among CAD tool users, syntactic structure, and so on. Design Framework II has its own analog hardware description language (HDL), called SpectreHDL, which is understood by Cadence's simulator SPECTRE. SpectreHDL was the first language we considered using for our front-end. However, this idea was set aside after we found out that SpectreHDL is a proprietary language, in order to avoid the legal complications that the use of a proprietary language would entail. Similar considerations ruled out the use of other proprietary languages, such as Analogy's MAST and Anacad's A-VHDL.

Languages that are widespread and in the public domain are preferable to proprietary languages whose use is not very common. In this respect, a SPICE-like language would seem to be the best choice. There are a number of SPICE-like languages in the public domain which might have been used, possibly with some minor modifications, for our front-end. On the other hand, it is much easier to write a netlist generator and a parser for a structured language like VHDL than for a completely unstructured one, like the SPICE family. On the other hand, SPICE-like languages are very unstructured, which makes it difficult to write netlist generators and parsers for them. Moreover, every commercial CAD vendor seems to have its own version of the SPICE language, which is always different in some respect from

those used by other vendors.

Taking all these considerations into account, we concluded that the two best candidates were Verilog-A and VHDL-A, both of which are analog extension (still under development) of the corresponding digital languages. Both languages are in the public domain, are widely used by a variety of commercial CAD tools, and industry-wide standards for their analog extensions are currently being developed. Moreover, LEX and YACC files to write a VHDL parser are already publicly available, and with only minor modifications to them it would be possible to obtain a VHDL-A parser fairly easily. These considerations led us to choose VHDL-A as the interface language between the Design Framework II and our software tools, even if a standard for that language is still under development. Drafts of the proposed language semantics were obtained from the 1076.1 IEEE subcommittee, which is in charge of VHDL-A development. The language to be used by our tools is based on those drafts. When the final version of the VHDL-A standard is officially released, it should be possible to make our language conform to the adopted standard with only minor modifications.

2.2 Netlist Generation

Our tools will interface with the Cadence Design Framework II environment through Cadence's Open Simulation System (OSS). Among other things, OSS has the capability to generate automatically from the Cadence database a netlist formatted according to user specifications. Our plan was to use OSS to generate VHDL-A netlists from Cadence's Composer (their schematic capture tool), and then have our tools use the VHDL-A netlists as input. As this sequence of operations can be performed in a completely automated way, the whole process is completely transparent to the user.

Cadence's Design Framework II supports netlist generation in user-defined formats through customizable functions that must be written in Cadence's SKILL language. These functions as a whole constitute as so called "output formatter." The functions necessary to traverse the design database are included in the Design Framework II. After setting the output variables, the netlister traverses the database and builds a list of all the cells that must be placed in the netlist. For each cell, the traversal functions call one of the output formatter's function to generate appropriate output corresponding to that portion of the design.

In order to write a netlister suitable for our purposes, we had to become familiar with Composer, one of Cadence's schematic entry tools. We also learned the basics of the Cadence language named SKILL and the of the hierarchical netlister tool HNL. As explained above, a knowledge of SKILL is necessary to understand and write appropriate netlister functions that will map the various names, nodes and subcircuit descriptions found in the circuit schematic to the netlist format. Then, using Cadence-supplied files as templates, we wrote a set of netlisting functions that generate output in VHDL-A from Cadence schematic diagrams. As an example, the circuit whose diagram is shown in Fig. 1 generates the following output:

```
architecture structural of amplifier is
begin
```

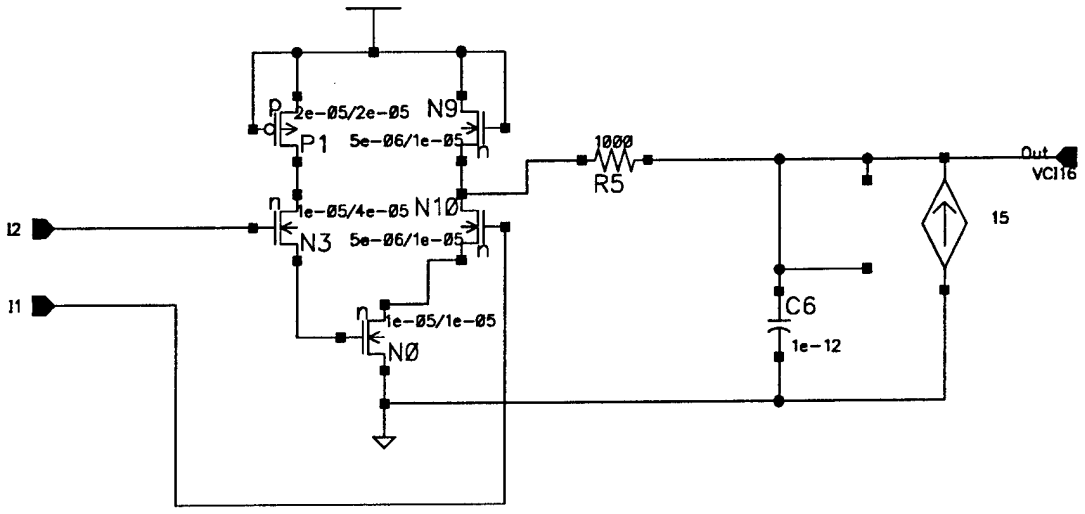

Figure 1: A Sample Schematic from Cadence's Design Framework II

```
    vci16: VCI generic map (15)
    port map (p => out, m => gnd, cp => out, cm => out);
    c6: capacitor generic map (1e-12)
    port map (p => out, m => gnd);
    r5: resistor generic map (1000)
    port map (p => net15, m => out);
    pmp1: pmos generic map (2e-05, 2e-05)
    port map (D => vdd, G => vdd, S => net19, B => vdd);
    nmn10: nmos generic map (5e-06, 1e-05)
    port map (D => net13, G => i1, S => net15, B => gnd);
    nmn9: nmos generic map (5e-06, 1e-05)
    port map (D => net15, G => vdd, S => vdd, B => gnd);
    nmn3: nmos generic map (1e-05, 4e-05)
    port map (D => net19, G => i2, S => net11, B => gnd);
    nmn0: nmos generic map (1e-05, 1e-05)
    port map (D => net13, G => net11, S => gnd, B => gnd);
end structural;
```

A partial listing of the SKILL files that make up the netlister is given in Appendix 1.

2.3 HDL Parser

To generate a parser for the netlists produced by our netlister, we relied on the lexical analyzer generator LEX [3] and on the parser generator YACC [4]. LEX is a program generator designed for lexical processing of character input streams. It accepts a high-level, problem

oriented specification for character string matching, and produces a program in a general purpose language which recognizes regular expressions. The regular expressions are specified by the user in the source specifications given to LEX. The LEX written code recognizes these expressions in an input stream and partitions the input stream into strings matching the expressions. The program that recognizes the expressions is generated in the *C* programming language.

YACC provides a general tool for imposing structure on the input to a computer program. The YACC user prepares a specification of the input process: this includes rules describing the input structure, code to be invoked when these rules are recognized, and a low-level routine to do the basic input. YACC then generates a function to control the input process. This function, called a *parser,* calls the user-supplied low-level input routine (the lexical analyzer generated by LEX) to pick up the basic items (*tokens*) from the input stream. These tokens are organized according to the input structure rules, called grammar rules: when one of these rules has been recognized, then user code supplied for this rule is invoked. In this way, the behavior of the parser can be tailored to the user's needs. Both LEX and YACC are standard commands in most UNIX-derived operating systems, and have been extensively used to implement compilers for the most disparate tasks.

As mentioned earlier, one reason that led us to choose VHDL-A as the input language for our tools was the availability of LEX and YACC files for VHDL. Because VHDL-A is an extension of VHDL, we were able to use those files as a starting point to generate a lexical analyzer and a parser suitable for our purposes. In particular, the VHDL LEX file was written at the University of Dortmund, Germany, based on a scanner included in the ALLIANCE CAD toolset, which is a product of the MASI/CAO-VLSI CAD Team, Universitè Pierre et Marie Curie, Paris, France. This file required only minor modifications to adapt it to VHDL-A. The YACC file, on the other hand, was not readily modifiable to suit our needs. As we did not need a parser for the complete VHDL-A language, but only for a small subset of it, we decided that it would be faster and easier to write our own YACC file, using the VHDL YACC file only as a template.

We now have LEX and YACC files that can be used to parse netlists, such as the one shown in Section 2.2, generated by the netlister that we wrote for Cadence's Framework II environment. These files provide us with a working interface with the Cadence environment: for reference purposes, a listing of those files is given in Appendix 2.

2.4 Development of a hierarchical netlister

The development of a hierarchical netlister that generates VHDL-A output was completed. This included modifying some SKILL code already written so that the netlister's output would conform to the current VHDL standard, and to extract parameter names and values directly from cell properties. The data that must be generated by the netlister includes information that is specific to each element in the netlist (e.g., the number and the names of terminals, the names and the values of element parameters). When the netlister was initially developed, element-specific code was written for each element that the netlister was

to be able to handle. This meant that new code had to be written to give the netlister the capability to handle additional elements, and also that the netlister wouldn't be able to handle a library where the element-specific information was encoded in a different manner. By making appropriate modifications to the netlister's SKILL code, this restriction was removed and the resulting netlister is now able to generate VHDL-A output in a way that is element- and library-independent. Further efforts where pursued to help improve the performance of the netlister. Test circuits were developed and netlisted to verify the proper operation of the netlister. During this process, a number of switched-capacitor circuits were entered into the Cadence environment for future testing.

2.5 Development of a parser for a subset of the VHDL-A language

CADENCE's Design Framework II has a component library dedicated to switched-capacitor circuit design (the *scdslib* library). This library contains ideal components, such as switches and clocks, that are not contained in other libraries. In order to make our software compatible with switched-capacitor designs that use that library, the VHDL-A interface had to be modified slightly. Specifically, design primitives were added for switches and clocks, and the VHDL-A parser was modified accordingly. The SKILL code for the HNL netlister also had to be modified to handle the additional primitives. This is an examples of the netlist generated from a switched-capacitor design that uses the scdslib library:

```
architecture structural of example_1wsrc is
   node  Vout, Vin, net10, net6, net11;
   begin
  V1: entity vdc
  generic map( vdc => 1.000000, srcType => "dc", FNpairs => 0)
  port map( MINUS => gnd, PLUS => Vin);
  I15: entity clock
  generic map( clockName => "phi2", phaseList => "0 1");
  I14: entity clock
  generic map( clockName => "phi1", phaseList => "1 0");
  N0: entity sc0pamp2
  generic map( bw => 1000.000000, Gain => 100000.000000)
  port map( negout => gnd, negref => net11, posout => Vout,
  posref => gnd);
  C1: entity capacitor
  generic map( c => 1e-12 )
  port map( N1 => net10, N2 => net6);
  N7: entity spst
  generic map( clockName => "phi1", Ron => "1")
  port map( N1 => net11, N2 => Vout);
```

```
end architecture;
```

2.6 Code development for the power estimator

The overall configuration of the switched-capacitor power estimator was outlined, and the various data structures that will be needed were defined as dictated by the chosen configuration. To keep the code as modular as possible, it was decided to use an object-oriented approach whenever possible. A single object class will be used to handle all the various element types in the circuit: element specific code will be written only when it is indispensable to do so (e.g. to evaluate the stamp of that particular element). Overall, the switched-capacitor power estimator contains the following modules:

1. An input module
2. A preprocessing module
3. A circuit matrix evaluation module
4. A matrix solution module (to compute average power dissipation)
5. A Lyapunov solution equation module (to compute the standard deviation of the power dissipation).

The functionality of those modules is the following:

1. The input module parses the VHDL-A input netlist and store all the circuit data in suitable data structures.
2. The preprocessing module determines the connected components of the circuit during each clock phase, i.e., it identifies the clusters of nodes that are connected by closed switches during each clock phase. This information is needed to build the modified nodal analysis matrices corresponding to each clock phase.
3. The evaluation module builds one modified nodal analysis matrix for each clock phase, using the information provided by the preprocessing module.
4. The matrix solution module computes the expected values of the circuit's electrical variables (voltages, currents) by solving a system of linear equations in a way that takes advantage of the system's particular structure. The expected average power dissipation can be computed from this information.
5. The Lyapunov equation solution module solves the discrete-time Lyapunov equation and computes the standard deviation of the power dissipation.

After the overall structure of the power estimator was settled upon, the next step was to determine the appropriate data structures and algorithms that would be needed for the power estimator. In particular, a necessary preprocessing step is the identification of the connected components of the circuit during each clock phase. By definition, a connected component is a cluster of nodes in the circuit that are connected by closed switches. Because the configuration of the switches changes at each clock phase, so do the connected components. An efficient method was found for determining the connected components for each phase of the circuit. With this information available, it is possible to build the modified nodal analysis (MNA) matrices for each clock phase using the familiar concept of element stamp. By definition, the stamp of an element contains that element's contribution to the MNA matrix. Because the nodes an element is connected to may belong to different connected components in different clock phases, it is possible for an element to have different stamps in each clock phase. Once the connected components have been identified, it is possible to define the stamps of each element in the circuit and use them to build the MNA matrices. This approach is consistent with the chosen object-oriented approach, according to which the code is kept as modular as possible, with each module being associated with an object (in this case, a particular type of circuit element).

After all the necessary data structures were defined, actual coding was undertaken. However, an examination of the numerical results initially obtained by the switched-capacitor power estimator revealed that the power dissipation computed by the simulator was sometimes incorrect. The reason was that in order to estimate the power dissipated by a generic element, one must compute $E(vi)$, where i and v are the current and the voltage across the element, and $E()$ represents the expected value. In the case of independent sources, the value of v is known with certainty, which implies that $E(vi) = vE(i)$. Thus in this case the power dissipated can be computed from a first-order moment (i.e., $E(i)$). In the case of op-amps, the values of both v and i are uncertain, which means that one must compute $E(vi)$. In general, $E(vi)$ is not equal to $E(v)E(i)$ (which is the incorrect assumption that had been made initially), which means that in this case the power dissipated cannot be computed simply from first-order moments. Instead, the second-order moment $E(vi)$ must be computed, which can be done only by solving a discrete Lyapunov equation.

Because the size of the Lyapunov equation can be very large, a literature search was performed in order to determine if efficient algorithms for the solution of this particular type of equation had already been published. This search revealed that, although a number of such algorithms exist, all have drawbacks that severely limit their usefulness. It was therefore decided to use a "brute force" approach, i.e., to solve the Lyapunov equation using ordinary Gaussian elimination. Although this method does not try to exploit the particular structure of the system of equations, it is the most straightforward to implement, and it is reliable in its numerical performance. Moreover, among the various methods considered, it was the one that could be implemented in the least time. After this particular method has been successfully implemented, and its numerical performance can be measured on actual circuits, we will look for alternative, more efficient algorithms to solve the Lyapunov equation.

The Lyapunov equation generated by the power estimation algorithm consists of a set of equations of the following type:

$$\mathbf{Y}_k\mathbf{P}_{k+1}\mathbf{Y}_k^T = \mathbf{D}_k\mathbf{P}_k\mathbf{D}_k^T + \mathbf{F}\mathbf{Q}\mathbf{F}^T.$$

There is one such equation of each clock phase. The unknowns in this equation are the entries of the matrix $\mathbf{P}_{k+1}$, which a symmetric $n \times n$ matrix, where n is approximately equal to the number of nodes in the circuit. Because the matrix $\mathbf{P}_{k+1}$ is symmetric, the number of unknown entries is $n(n+1)/2$. Thus, if there are p clock phases (typically, $p = 2$ or $p = 4$), the total number of unknowns (that is, the size of the overall system that must be solved) is $pn(n+1)/2$. In order to solve this system using ordinary Gaussian elimination, the equations described above must be assembled in a large system of the type $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{x}$ is a vector containing all the $pn(n+1)/2$ unknowns. The implementation of this algorithm requires mapping the entries of the matrices $\mathbf{P}_k$ into the vector $\mathbf{x}$, and the entries of $\mathbf{Y}_k$ and $\mathbf{D}_k$ into the coefficient matrix $\mathbf{A}$. These modifications were incorporated into the power estimator code.

In its final implementation, the power estimator does all of the necessary preprocessing for each phase of the circuit, and computes the average power dissipated by the circuit under the specified input conditions. The power estimator can read VHDL-A netlists generated by the netlister previously completed, and it computes an estimate of the average switching power dissipated by the circuit for the given input statistics. To test the power estimator, a library containing a number of switched-capacitor circuits was created using CADENCE's Design Framework II tools. The size of the circuits ranged from one op-amp and a few capacitors to several op-amps and tens of capacitors. The netlist generator was then run on each circuit, and the resulting netlists were used as inputs to the power estimator. The results are summarized in Table 1:

Table 1: Results of Power Estimation Simulations

Circuit	Matrix Size	Switching Power (μW)	CPU time (ms)
SCAmp	14	1.500	250
Biquad	44	137.8	310
Ellipt0	96	12.34	550
Ellipt1	60	5.189	350
Ellipt2	66	.2594	340

The operation of the switched-capacitor power estimator and the results of the numerical simulations are described in a paper which was presented at the 1996 International Conference on Computer-Aided Design.

Chapter 3

Power Estimation in Switched-Capacitor Circuits

3.1 Background

A number of low-power designs, such as those for mobile communication equipment, contain switched-capacitor circuits. In such designs it is important to be able to estimate the power dissipated by the switched-capacitor portion of the circuit. One of the tasks of this research project was the development of CAD software for the computation of statistical information about the power dissipated in a switched-capacitor circuit when corresponding statistical information about the inputs to the circuit is known. Ordinary circuit simulators are not suited for this task, because they can only compute the power dissipated by the circuit for one specific set of input signals. The algorithm does not require Monte-Carlo analyses, and it accounts for correlation among the inputs. To demonstrate the software's performance, numerical results obtained on a number of sample switched-capacitor circuits are reported.

Most of the published work on CAD tools for power estimation concerns itself only with digital circuits. However, many emerging product technologies, such as personal computing and communication systems, involve mixed signal processing. Therefore, in order to predict the power dissipation of such systems, it is necessary to estimate the power dissipated by the analog portion. In the specific case of switched-capacitor circuits, the total power dissipation is the sum of two terms: the power absorbed by the op-amps, and the switching power, that is the power dissipated in charging and discharging the capacitors. In traditional switched-capacitor designs, the first term is by far the dominant one. The push towards the reduction of power requirements, however, has spurred the design of op-amps whose power dissipation is in the tens of microwatts [5]. It is precisely in this type of low-power design that the estimation of the switching power becomes important, because it is no longer a negligible fraction of the total power dissipation. It must be noted that in most cases, the power dissipated by an op-amp can be assumed to be constant, regardless of the applied input [6]. Therefore, only the estimation of the switching power requires nontrivial calculations.

This report describes an algorithm for statistical estimation of power dissipation in switched-capacitor analog circuits. Note that ordinary circuit simulators are not suited for this task, because they can only compute the power dissipated by the circuit *for one specific set of input signals.* In contrast, the algorithm described here allows the computation of statistical information about the power dissipated in a switched-capacitor circuit *for a*

variety of input signals whose statistics are given. This approach mirrors the one generally used to estimate power dissipation in digital CMOS circuits. The algorithm can be applied to any switched capacitor circuit whose elements can be modeled as ideal switches, linear capacitors, independent and linear voltage-controlled voltage sources, and ideal operational amplifiers. It does not require Monte-Carlo analyses, and it accounts for correlation among the inputs to the circuit.

3.2 Statistical analysis of switched-capacitor circuits

The analysis of switched-capacitor circuits using conventional simulation techniques is practically impossible. For this reason, certain simplifying assumptions about the circuit are usually made in order to reduce the required computational effort to manageable levels. For instance, MOSFETS are modeled as ideal switches; moreover, it is assumed that all other elements in the circuit can be represented by linear capacitors, ideal operational amplifiers, and independent or linear voltage-controlled voltage sources.

Various methods of analysis have been proposed for circuits of this kind [7, 8, 9]. Broadly speaking, they all rely on equations resulting from the law of conservation of charge and on the branch constitutive equations of the memoryless elements. The analysis method used here is the one described in [8]: at each clock phase, the connected components of the circuit are built, each component consisting of nodes connected together by closed switches. The only elements that can connect two different components are capacitors or memoryless elements (voltage sources, op-amps). For each connected component, charge conservation equations can be written: the total charge present on all the capacitors connected to a given component immediately before the switching instant t_k must equal the total charge present on the same capacitors immediately after t_k plus the charge that has left the component through the memoryless elements. Thus for each connected component the corresponding charge conservation equation is:

$$\sum_j q_j^C(t_k^+) + \sum_j q_j(t_k) = \sum_j q_j^C(t_{k-1}^+),$$

where q_j^C represents the charge on the j-th capacitor, and q_j is the charge flowing through the j-th memoryless element. If the j-th capacitor, C_j, is connected between nodes j_1 and j_2, then $q_j^C = C_j(v_{j_1} - v_{j_2})$, and the charge conservation equation can be rewritten as:

$$\sum_j C_j[v_{j_1}(t_k^+) - v_{j_2}(t_k^+)] + \sum_j q_j(t_k) = \sum_j C_j[v_{j_1}(t_{k-1}^+) - v_{j_2}(t_{k-1}^+)].$$

Branch equations for the closed switches and the memoryless elements are appended to the charge conservation equations. The resulting system has the form [8]:

$$\mathbf{Y}_k \begin{bmatrix} \mathbf{v}(t_k^+) \\ \mathbf{q}(t_k) \end{bmatrix} = \begin{bmatrix} \mathbf{W}_k \mathbf{v}(t_{k-1}^+) \\ \mathbf{E}\mathbf{u}(t_k) \end{bmatrix} \tag{3.1}$$

where $\mathbf{v}$ represents the vector of the node voltages, $\mathbf{q}$ the charges flowing through the memoryless elements, $\mathbf{u}$ the voltages of the independent sources, and t_k is the k-th switching instant. Letting $\mathbf{x}_k = [\mathbf{v}^T(t_k^+), \mathbf{q}^T(t_k^+)]^T$ and $\mathbf{u}_k = \mathbf{u}(t_k)$, eqn. (3.1) can be rewritten as:

$$\begin{aligned} \mathbf{Y}_k \mathbf{x}_k &= \begin{bmatrix} \mathbf{W}_k \mathbf{v}(t_{k-1}^+) \\ \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{E}\mathbf{u}(t_k) \end{bmatrix} = \\ &= \mathbf{D}_k \mathbf{x}_{k-1} + \mathbf{F}\,\mathbf{u}_k, \end{aligned} \tag{3.2}$$

for appropriate matrices $\mathbf{D}_k$ and $\mathbf{F}$.

In most switched-capacitor circuits, the switching schedule is periodic: this implies that $\mathbf{Y}_{k+N} = \mathbf{Y}_k$ and $\mathbf{D}_{k+N} = \mathbf{D}_k$ for some positive integer N. Furthermore, it will be assumed that the inputs to the circuit form a stationary process Markov process of order zero with mean $\bar{\mathbf{u}}$ and covariance matrix $\mathbf{Q}$. Then taking the expected value of both sides of eqn. (3.2) yields:

$$\mathbf{Y}_k \bar{\mathbf{x}}_k = \mathbf{D}_k \bar{\mathbf{x}}_{k-1} + \mathbf{F}\bar{\mathbf{u}}, \tag{3.3}$$

where $\bar{\mathbf{x}}_k$ denotes the mean of $\mathbf{x}_k$. Because the sequences $\{\mathbf{Y}_k\}$ and $\{\mathbf{D}_k\}$ are periodic of period N, so is the sequence $\{\bar{\mathbf{x}}_k\}$. As a consequence, $\bar{\mathbf{x}}_1, \bar{\mathbf{x}}_2, \ldots, \bar{\mathbf{x}}_N$ can be computed as the solution of the following system of linear equations:

$$\begin{aligned} \mathbf{Y}_1 \bar{\mathbf{x}}_1 &= \mathbf{D}_1 \bar{\mathbf{x}}_N + \mathbf{F}\bar{\mathbf{u}} \\ \mathbf{Y}_2 \bar{\mathbf{x}}_2 &= \mathbf{D}_2 \bar{\mathbf{x}}_1 + \mathbf{F}\bar{\mathbf{u}} \\ &\vdots \\ \mathbf{Y}_N \bar{\mathbf{x}}_N &= \mathbf{D}_N \bar{\mathbf{x}}_{N-1} + \mathbf{F}\bar{\mathbf{u}}, \end{aligned}$$

or, in short:

$$(\hat{\mathbf{Y}} - \hat{\mathbf{D}})\hat{\mathbf{x}} = \hat{\mathbf{F}}\hat{\mathbf{u}}, \tag{3.4}$$

where $\hat{\mathbf{x}} = [\bar{\mathbf{x}}_1^T, \bar{\mathbf{x}}_2^T \ldots \bar{\mathbf{x}}_N^T]^T, \hat{\mathbf{u}} = [\bar{\mathbf{u}}^T \bar{\mathbf{u}}^T \ldots \bar{\mathbf{u}}^T]^T$, and:

$$\begin{aligned} \hat{\mathbf{F}} &= \mathrm{diag}(\mathbf{F}, \mathbf{F} \ldots \mathbf{F}) \\ \hat{\mathbf{Y}} &= \mathrm{diag}(\mathbf{Y}_1, \mathbf{Y}_2 \ldots \mathbf{Y}_N) \\ \hat{\mathbf{D}} &= \begin{bmatrix} & & & \mathbf{D}_1 \\ \mathbf{D}_2 & & & \\ & \ddots & & \\ & & \mathbf{D}_N & \end{bmatrix}. \end{aligned}$$

The dimension of $\hat{\mathbf{Y}} - \hat{\mathbf{D}}$ is $N \cdot \dim(\mathbf{Y}_k)$, and can therefore be quite large. Fortunately, because of the particular structure of $\hat{\mathbf{Y}} - \hat{\mathbf{D}}$, it is possible to decompose it using block LU decomposition. More precisely, let $\mathbf{Y}_k = \mathbf{L}_k \mathbf{U}_k$; then it is easy to verify that $\hat{\mathbf{Y}} - \hat{\mathbf{D}} = \hat{\mathbf{L}}\hat{\mathbf{U}}$,

where:

$$\hat{\mathbf{L}} = \begin{bmatrix} \mathbf{L}_1 & & & & \\ -\mathbf{D}_2\mathbf{U}_1^{-1} & \mathbf{L}_2 & & & \\ & -\mathbf{D}_3\mathbf{U}_2^{-1} & \mathbf{L}_3 & & \\ & & \ddots & & \\ & & & -\mathbf{D}_N\mathbf{U}_{N-1}^{-1} & \mathbf{L}'_N \end{bmatrix},$$

$$\hat{\mathbf{U}} = \begin{bmatrix} \mathbf{U}_1 & & & -\mathbf{L}_1^{-1}\mathbf{D}_1 \\ & \mathbf{U}_2 & & -\mathbf{L}_2^{-1}\mathbf{D}_2\mathbf{Y}_1^{-1}\mathbf{D}_1 \\ & & \mathbf{U}_3 & -\mathbf{L}_3^{-1}\mathbf{D}_3\mathbf{Y}_2^{-1}\mathbf{D}_2\mathbf{Y}_1^{-1}\mathbf{D}_1 \\ & & & \vdots \\ & & & \mathbf{U}'_N \end{bmatrix},$$

and

$$\mathbf{L}'_N\mathbf{U}'_N = \mathbf{Y}'_N = \mathbf{Y}_N - \mathbf{D}_N\mathbf{Y}_{N-1}^{-1}\mathbf{D}_{N-1}\mathbf{Y}_{N-2}^{-1}\cdots\mathbf{Y}_1^{-1}\mathbf{D}_1.$$

Therefore, it is not necessary to decompose $\hat{\mathbf{Y}}$: the solution of eqn. (3.4) can be computed using simply the LU factors of $\mathbf{Y}_1, \ldots, \mathbf{Y}_{N-1}$ and $\mathbf{Y}'_N$.

Similarly, let $\mathbf{P}_k$ be the covariance matrix of the random variable $\mathbf{x}_k$. The sequence $\{\mathbf{P}_k\}$ is also periodic of period N, and it can be shown [10] that it satisfies the following difference equation:

$$\mathbf{Y}_k\mathbf{P}_k\mathbf{Y}_k^T = \mathbf{D}_k\mathbf{P}_{k-1}\mathbf{D}_k^T + \mathbf{F}\mathbf{Q}\mathbf{F}^T. \tag{3.5}$$

Consequently, letting $\hat{\mathbf{P}} = \mathrm{diag}(\mathbf{P}_1, \mathbf{P}_2 \ldots \mathbf{P}_N)$ and $\hat{\mathbf{Q}} = \mathrm{diag}(\mathbf{Q}, \mathbf{Q} \ldots \mathbf{Q})$, the matrix $\hat{\mathbf{P}}$ can be computed as the solution of the discrete Lyapunov equation:

$$\hat{\mathbf{Y}}\hat{\mathbf{P}}\hat{\mathbf{Y}}^T = \hat{\mathbf{D}}\hat{\mathbf{P}}\hat{\mathbf{D}}^T + \hat{\mathbf{F}}\hat{\mathbf{Q}}\hat{\mathbf{F}}^T. \tag{3.6}$$

Eqn. (3.6) is linear in the entries of $\hat{\mathbf{P}}$, and in principle it could be solved using standard techniques, such as LU decomposition [11]. In practice, because the size of the system in eqn. (3.6) can be extremely large, it is advisable to compute the solution using special-purpose algorithms [12].

As will be shown in the next section, the estimation of the power dissipation requires also the computation of the expected value of $(\mathbf{x}_k - \bar{\mathbf{x}}_k)(\mathbf{x}_{k-1} - \bar{\mathbf{x}}_{k-1})^T$. Letting $\mathbf{P}_{k,k-1} = \overline{(\mathbf{x}_k - \bar{\mathbf{x}}_k)(\mathbf{x}_{k-1} - \bar{\mathbf{x}}_{k-1})^T}$, it can be shown, after some algebraic manipulation, that the following relationship must be satisfied:

$$\mathbf{Y}_k\mathbf{P}_{k,k-1} = \mathbf{D}_k\mathbf{P}_{k-1}.$$

Therefore $\mathbf{P}_{k,k-1}$ can be computed from $\mathbf{P}_{k-1}$ simply by solving a system of linear equations.

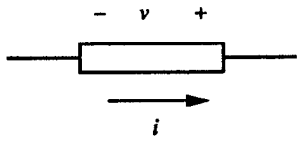

Figure 2: Power Generated by an Element

3.3 Power estimation

The statistical information about the circuit variables, computed as described in the previous section, can be used to obtain probabilistic information about the circuit's power dissipation in the following way. By power conservation, the power dissipated by the circuit must be equal to the power delivered by the active elements, namely voltage sources and operational amplifiers. Consider a generic circuit element, as shown in Fig. 2. The current through the element, $i(t)$, is always zero, except at the switching instants t_k, when it becomes (theoretically) infinite. Assuming that the interval $[t_a, t_b]$ contains a single switching instant t_k, $i(t) = q(t_k)\delta(t - t_k)$, where $q(t_k)$ is the amount of charge that flows through the element at time t_k, and $\delta(\cdot)$ is Dirac's impulse function. The energy delivered by the element at the switching instant t_k is:

$$J = q(t_k) \int_{t_a}^{t_b} v(t)\delta(t - t_k)\, dt.$$

If the voltage $v(t)$ is not discontinuous at t_k, the integral in the expression above has a well-defined value, namely $v(t_k)$; in this case, the energy delivered by the element is: $J = q(t_k)v(t_k)$. For example, let the element under consideration be an independent voltage source, $V_m(t)$ be the source value, and q_m be the variable describing the charge flowing through the source. Then the power delivered by that source over a clock period is:

$$W_m = \frac{1}{T} \sum_{k=1}^{N} V_{m,k} q_{m,k},$$

where $V_{m,k}, q_{m,k}$ are shorthands for $V_m(t_k), q_m(t_k)$, and T denotes the clock period. As a consequence, the expected value of the power delivered by the source is given by:

$$\overline{W}_m = \frac{1}{T} \sum_{k=1}^{N} V_{m,k} \bar{q}_{m,k}. \tag{3.7}$$

The evaluation of the power delivered by other active elements (controlled voltage sources and op-amps) is more difficult, because the voltage across them can have a discontinuity at the switching instants: in this case the value of the integral $\int_{t_a}^{t_b} v(t)\delta(t - t_k)\, dt$ is not well-defined. Thus, in the case of a discontinuity in the voltage across the element, the energy delivered by the element at t_k can be as high as:

$$J = \max(v_{k-1}, v_k) q_k$$

and as low as:

$$J = \min(v_{k-1}, v_k) q_k.$$

To avoid this uncertainty, an average value:

$$J = \frac{v_{k-1} + v_k}{2} q_k$$

can be used. It can be shown that this particular choice satisfies power conservation.

Under this assumption, the average power delivered by the element is given by:

$$\overline{W}_m = \frac{1}{2T} \sum_{k=1}^{N} (\overline{q_{m,k} v_{m,k}} + \overline{q_{m,k} v_{m,k-1}}). \tag{3.8}$$

The quantities $\overline{q_{m,k} v_{m,k}}$ and $\overline{q_{m,k} v_{m,k-1}}$ are second-order moments, and can be obtained from the matrices $\mathbf{P}_k$ and $\mathbf{P}_{k,k-1}$ defined in the previous section. For instance:

$$\overline{q_{m,k} v_{m,k}} = \bar{q}_{m,k} \bar{v}_{m,k} + \overline{(q_{m,k} - \bar{q}_{m,k})(v_{m,k} - \bar{v}_{m,k})},$$

and the last term in the equation above is an entry of the matrix $\mathbf{P}_k$. A similar identity, involving an entry of the matrix $\mathbf{P}_{k,k-1}$, can be obtained for $\overline{q_{m,k} v_{m,k-1}}$.

3.4 Algorithm implementation

Two things must be accomplished so as to implement the analysis method described in Section 3.2. First, the connected components of the circuit must be determined at each clock phase, and the charge conservation and branch constitutive equations must be numbered accordingly. Second, the system of equations (3.2) must be built at each clock phase from the circuit netlist.

The determination of the connected components for an undirected graph is a well-studied problem and relatively fast algorithms exist to solve it. The treatment of this problem can usually be found in most introductory books on algorithms such as [13]. By using the heuristics of union by rank and path compression in the implementation, a practically linear running time can be obtained [13, p. 449]. Once the connected components have been found, determining membership in a set is a constant time operation, assuming that indexing into an array can be done in constant time.

A unique numbering for the system of equations can easily be obtained by a slight modification to the pseudocode for the connected components presented in [13]. Initially, each connected component consists exactly of one node, and is assigned a the same number as the node. When two connected components are merged because they are joined by an edge (i.e. a closed switch), one of the two component numbers is assigned to the resulting connected component, while the other is assigned to the switch. When system of equations (3.2) is assembled, charge conservation equations are numbered according to the connected components, while each closed switch's number determines the number of the corresponding

branch equation. Branch equations for the other memoryless elements are appended as in ordinary Modified Nodal Analysis. This methodology yields a unique numbering scheme for all the equations in the system.

Once the connected components have been identified, the system of equations can be built from the circuit netlist by relying on the usual concept of element stamps.

3.5 Numerical results

In order to facilitate the use of our power estimator, it was decided to interface it with an existing commercial CAD environment. Cadence's Design Framework II was chosen for this purpose. Design Framework II provides a set of tools, known as OSS, that allows third-party developers to incorporate their own simulators into the Cadence system. OSS provides an interface for the developer which allows the execution of a simulation, the generation of netlists and input vectors, and the display of simulations results which are consistent with the interface used by the other tools in the Design Framework II environment.

The first step in integrating a simulator into the Design Framework II environment is the generation of a netlist in a standard format. The netlist is obtained by traversing the Cadence database to obtain relevant information about the circuit to be simulated. OSS provides two tools for automatic netlist generation: HNL and FNL. Both of these netlisters are capable of producing output in a user-defined format by utilizing the standard Cadence language, SKILL.

In order to produce a netlist, it was necessary to choose a HDL. Several factors had to be considered in choosing a HDL. These factors included: portability, popularity among CAD tool users, availability of the language, and so on. The HDL that was finally chosen was VHDL-A. This is an analog extension to the digital language VHDL, which the 1076.1 IEEE Standards Subcommittee is currently developing a standard for. By utilizing the HNL of Cadence's OSS, a rudimentary VHDL-A netlister was developed. A parser for a subset of the VHDL-A language was developed to allow the simulator to read the resulting netlist.

The simulator was used to calculate the expected power dissipation of a number of switched-capacitor circuits of varying sizes. In order to expand the library of benchmark circuits on which to test the power estimator, additional designs were sought from industrial sources. CADENCE contributed the design of a switched-capacitor elliptic filter. Texas Instruments was also contacted for the purpose of obtaining additional examples of switched-capacitor circuits to be added to the set of benchmarks. Texas Instrument agreed to provide one such circuit upon the signing of a nondisclosure agreement by Georgia Tech. Such agreement was signed, and we obtained from Texas Instruments the design of another switched-capacitor circuit, which has been added to the benchmark library.

The clock frequency for each of these circuit was taken to be 1 MHz. The CPU times represent the running times of the simulator on a Sparcstation 5/20 running SunOS 4.1.4. The results of these simulations can be found in Table 2: the second column shows the dimension of eqn. (3.4) for the corresponding circuit, while the expected power dissipation is reported in the third column. It is worth repeating that those figures represent only the

switching power dissipation of the circuit: in order to obtain the total power dissipation, one must add the power dissipated by the op-amps, which, as mentioned in the Introduction, can be taken to be a fixed value.

Table 2: Results of Power Estimation Simulation

Circuit	Matrix Size	Switching Power (μW)	CPU time (ms)
SCAmp	14	1.500	250
Biquad	44	137.8	310
Ellipt0	96	12.34	550
Ellipt1	60	5.189	350
Ellipt2	66	.2594	340

3.6 Summary

An algorithm for statistical estimation of power dissipation in switched-capacitor circuits has been described. The proposed algorithm computes the mean and the variance of all variables of interest as the solution of certain sets of *linear, algebraic* equations. Therefore, no Monte-Carlo analyses or time-domain simulations of the circuit involved are required. A potential drawback is that the size of the system of equation that must be solved to compute $\hat{\mathbf{P}}$ (eqn. (3.6)) can be very large, even for circuits of modest dimensions. On the other hand, this system of equations must be solved only once. For this reason, trying to keep the computational effort required to solve the system to a minimum is not as crucial as it is, for instance, in conventional circuit simulation, where the system must be solved hundreds or even thousands of times. Moreover, it may be possible to compute the solution using special algorithms that take advantage of the particular structure and of the sparsity of $\hat{\mathbf{Y}}$ [14].

The algorithm also accounts for any correlation that may exist among the inputs to the circuit. This is in contrast to some of the published power estimation algorithms for digital circuits, which assume that the primary inputs to the circuit are statistically independent. In some cases, this may be an unrealistic assumption: for instance, when the inputs are signals generated by optical sensors that are part of an optical array. Power estimation algorithms for digital circuits that do not rely on the uncorrelation assumption usually require a significantly larger computational effort [2], while the algorithm described here allows the primary inputs to the circuit to be correlated without affecting the overall computational cost.

Chapter 4

Power Dissipation and Switching Order

4.1 Energy dissipation in switching circuits

When simulating switched-capacitor circuits, a number of assumptions are usually made in order to reduce the computational effort required to manageable levels. The types of elements contained in the circuit are limited to linear capacitors, ideal operational amplifiers, independent or linear voltage-controlled voltage sources, and ideal switches. Moreover, it is assumed that the switch configuration changes instantaneously at each switching instant t_k. As a consequence of those assumptions, it is possible to show that the voltages in the circuit are piecewise constant, changing values only at the switching instants; charge flow also occurs instantaneously at the switching instants, and is zero at any other time [8]. Mathematically speaking, this means that each branch current is represented by a series of Dirac's impulse functions located at the switching instants.

Consider now a generic two-terminal element, such as the one shown in Fig. 3. The total energy dissipated by this element over the time interval $[t_a, t_b]$ is:

$$J = \int_{t_a}^{t_b} v\, dq = \int_{t_a}^{t_b} vi\, dt. \tag{4.1}$$

Assuming that the interval $[t_a, t_b]$ contains a single switching instant t_k, the current through the element in Fig. 3 is given by: $i(t) = q\delta(t - t_k)$, where q is the amount of charge that flows through the element at time t_k. If v is continuous at t_k, the integral in eqn. (4.1) has a well-defined value, namely:

$$J = qv(t_k). \tag{4.2}$$

Instead, if v is discontinuous at t_k (see Fig. 4(a)), the value of the integral is undetermined, as the following analysis shows. Suppose that v, instead of having a discontinuity at t_k, changes

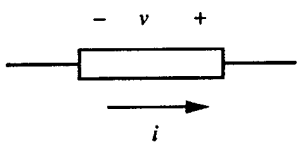

Figure 3: Power Dissipation in a Circuit Element

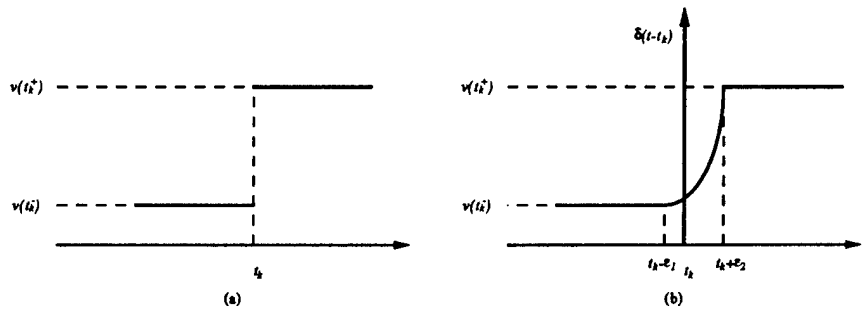

Figure 4: Discontinuity in a Branch Voltage

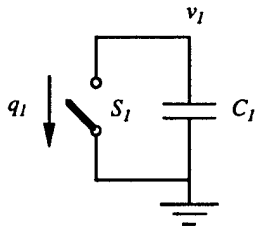

Figure 5: Computing the Energy Dissipated in a Switch

continuously over the interval $[t_k-\epsilon_1, t_k+\epsilon_2]$, as shown in Fig. 4(b). Depending on the values of ϵ_1 and ϵ_2 and on how $v(t)$ changes over $[t_k - \epsilon_1, t_k + \epsilon_2]$, the integral $\int_{t_a}^{t_b} v(t)\delta(t - t_k)\, dt$ can have any value between $\min(v(t_k^-), v(t_k^+))$ and $\max(v(t_k^-), v(t_k^+))$. As a consequence, the integral has no well-defined limit when ϵ_1 and ϵ_2 tend to zero. This means that in order to be able to evaluate the energy dissipated by an element, whose branch voltage is discontinuous at the switching instants, a more in-depth analysis of the circuit is required.

A good starting point is provided by the very simple circuit shown in Fig. 5: a capacitor connected in parallel to a switch. The switch is open for $t < t_k$, and it closes at $t = t_k$. To simplify notation, $v_1(t_k^-)$ (the value of voltage v_1 immediately before t_k) will be denoted by v_1^-; similarly, $v_1^+ = v_1(t_k^+)$ (in this example, $v_1^+ = 0$). Before $t = t_k$, the energy stored in the capacitor is $\frac{1}{2}C_1 v_1^{-2}$; after $t = t_k$ there is no energy in the capacitor, which means that, when the switch closes, it dissipates an amount of energy equal to $\frac{1}{2}C_1 v_1^{-2}$. Noting that the charge q_1 that flows through the switch at $t = t_k$ is equal to $C_1 v_1^-$, the amount of energy dissipated in the switch can be expressed as:

$$J = \frac{1}{2} v_1^- q_1. \tag{4.3}$$

This suggests that, if the element in Fig. 3 is a switch, the energy dissipated when the switch closes is given by $J = \frac{1}{2} v^- q$.

The analysis of a more complex example, such as the one shown in Fig 6, reveals some pitfalls with this expression for the energy dissipated in a switch. First, consider the case in

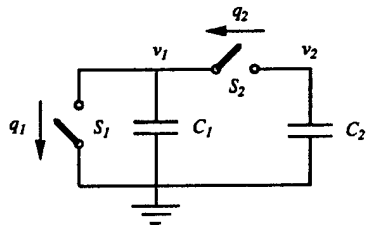

Figure 6: Two switches closing at the same time

which both switches close at the same time. A simple analysis shows that:

$$\begin{aligned} q_1 &= C_1 v_1^- + C_2 v_2^- \\ q_2 &= C_2 v_2^- \end{aligned}$$

According to eqn. (4.3), the energy dissipated in the switches is:

$$\begin{aligned} J_1 &= \frac{1}{2} v_1^- q_1 = \frac{1}{2} C_1 {v_1^-}^2 + \frac{1}{2} C_2 v_1^- v_2^- \\ J_2 &= \frac{1}{2}(v_2^- - v_1^-) q_2 = \frac{1}{2} C_2 {v_2^-}^2 - \frac{1}{2} C_2 v_1^- v_2^- \end{aligned}$$

Note that $J_1 + J_2 = \frac{1}{2} C_1 {v_1^-}^2 + \frac{1}{2} C_2 {v_2^-}^2$, which means that the principle of energy conservation is satisfied. On the other hand, either J_1 or J_2 could be negative, which would indicate that one of the switches would be generating energy: this, of course, is physically impossible.

To gain further insight into this problem, let us compute the energy dissipated in the switches when they close not simultaneously, but one before the other. If S_1 closes before S_2, it is immediate to verify that:

$$\begin{aligned} q_1 &= C_1 v_1^- \\ q_2 &= C_2 v_2^- \\ J_1 &= = \frac{1}{2} v_1^- q_1 = \frac{1}{2} C_1 {v_1^-}^2 \\ J_2 &= = \frac{1}{2} v_2^- q_2 = \frac{1}{2} C_2 {v_2^-}^2, \end{aligned}$$

where v_1^- and v_2^- refer to the voltage values before either switch closes, and q_1 and q_2 denote the charges flowing through the switches *at the instant when the switch is closed.* If S_2 closes first, v_1 and v_2 settle at an intermediate value $\bar{v} = \dfrac{C_1 v_1^- + C_2 v_2^-}{C_1 + C_2}$ while S_1 is open and S_2 is closed. As a consequence, q_1, q_2, J_1 and J_2 are given by the following expressions:

$$\begin{aligned} q_1 &= C_1 v_1^- + C_2 v_2^- \\ q_2 &= \frac{C_1 C_2}{C_1 + C_2}(v_2^- - v_1^-) \end{aligned}$$

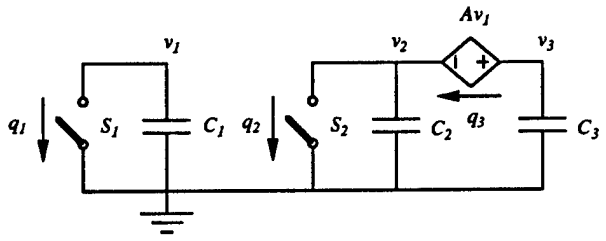

Figure 7: The switching order affects energy dissipation

$$J_1 = \frac{1}{2}\bar{v}q_1 = \frac{1}{2}\frac{(C_1v_1^- + C_2v_2^-)^2}{C_1 + C_2}$$
$$J_2 = \frac{1}{2}(v_2^- - v_1^-)q_2 = \frac{1}{2}\frac{C_1C_2}{C_1 + C_2}(v_2^- - v_1^-)^2$$

It is immediate to verify that, in either case, J_1 and J_2 are positive, as should be expected. With some algebraic manipulation, it can also be verified that energy is conserved, regardless of the order in which the switches close. On the other hand, the energy dissipated in a particular switch does depend on the switching order, even if, in this particular example, the total energy dissipation is the same in both cases, as it is equal to the energy initially stored in the capacitors.

That this is not always the case can be seen from the example shown in Fig. 7. This circuit contains a voltage source, which is an active element: let J_3 be the energy dissipated by it. If S_1 closes before S_2, a charge:

$$q_3 = \frac{C_2C_3}{C_2 + C_3}(v_3^- - v_2^-)$$

flows through the voltage source when S_1 is closed; at the same time, the voltage across the source drops from $v_3^- - v_2^-$ to zero. According to eqn. (4.3), the energy dissipated by the source is:

$$J_3 = \frac{1}{2}\frac{C_2C_3}{C_2 + C_3}(v_3^- - v_2^-)^2.$$

Additionally, v_2 and v_3 settle at an intermediate value given by:

$$\bar{v} = \frac{C_2v_2^- + C_3v_3^-}{C_2 + C_3}.$$

When S_2 closes, the charge q_2 that flows through the switch is equal to:

$$q_2 = (C_2 + C_3)\bar{v} = C_2v_2^- + C_3v_3^-.$$

As a consequence, the energy dissipated by each element is as follows:

$$J_1 = \frac{1}{2}C_1v_1^{-2}$$

$$J_2 = \frac{1}{2}\frac{(C_2 v_2^- + C_3 v_3^-)^2}{C_2 + C_3}$$
$$J_3 = \frac{1}{2}\frac{C_2 C_3}{C_2 + C_3}(v_3^- - v_2^-)^2$$

All elements, including the voltage source, dissipate energy, and the total amount of energy dissipated is equal to the energy initially stored on the capacitors; thus, once again, energy is conserved.

If S_2 closes first, the voltage across the voltage source remains equal to Av_1^-. Letting $q_{3,2}$ be the charge that flows through the source at this time, and $J_{3,2}$ the corresponding energy dissipated by the source, one obtains:

$$q_{3,2} = C_3 v_2^-$$
$$J_{3,2} = Av_1^- q_{3,2} = AC_3 v_1^- v_2^-.$$

When S_1 closes, the voltage across the source drops to zero, while more charge flows through it. An easy calculation shows that the amount of charge flowing through the source and the corresponding energy dissipation is given by:

$$q_{3,1} = C_3 A v_1^-$$
$$J_{3,1} = \frac{1}{2}C_3\left(Av_1^-\right)^2.$$

Thus, the energy dissipation in each element is:

$$J_1 = \frac{1}{2}C_1 v_1^{-2}$$
$$J_2 = \frac{1}{2}(C_2 + C_3)v_2^{-2}$$
$$J_3 = J_{3,1} + J_{3,2} = \frac{1}{2}C_3 A v_1^-(Av_1^- + 2v_2^-).$$

Although the principle of energy conservation is still satisfied, in this case the value of J_3 can be negative. If so, the voltage source would be *delivering* energy, and the total energy dissipated in the switches would be *greater* than the energy initially stored in the capacitors. In a physical realization of this circuit, any energy delivered by the voltage source would have to be drawn from a power supply. This means that, if S_2 closes before S_1, the circuit will draw energy from the power supply, while this does not happen if S_1 closes before S_2. Thus, this example shows that the switching order can affect the power dissipated by a switched-capacitor circuit.

In all the examples examined so far, when the switches were closed one at a time, eqn. (4.3) always yielded a positive value for the energy dissipated in each switch. Unfortunately, this is not always the case, as can be seen by analyzing the network shown in Fig. 8(a). It is immediate to verify that:

$$q_1 = C_1 v_1^-$$
$$J_1 = \frac{1}{2}(v_1^- - v_2^-)q_1 = \frac{1}{2}C_1(1 - A)v_1^{-2}.$$

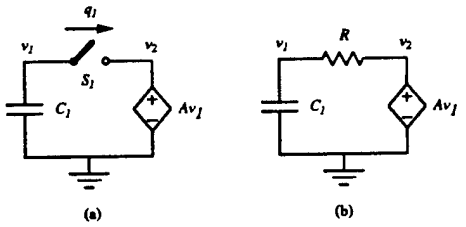

Figure 8: Negative Energy Dissipation in a Switch

Thus $J_1 < 0$ if $A > 1$. A reason for this nonphysical result can be surmised by replacing the switch with a resistor, as shown in Fig. 8(b). It can readily be seen that the behavior of this circuit is described by the following differential equation:

$$\dot{v}_1 + \frac{1}{RC_1}(1 - A)v_1 = 0,$$

which is unstable if $A > 1$. This observation leads to the conjecture that eqn. (4.3) will always yield a positive value for the energy dissipated in a switch whenever the network, obtained by replacing the switch with a positive resistor, is stable.

The results obtained from the analysis of the examples in Figs. 5 through 8 can then be summarized as follows:

- The expression for the energy dissipated by an element, given in eqn. (4.3), appears to be consistent with the principle of energy conservation.
- If the analysis is carried out under the assumption that multiple switches close at the same time, eqn. (4.3) can yield a negative number for the energy dissipated by a switch, which is physically impossible.
- A more realistic assumption about the operation of a switched-capacitor circuit is that the switches close one at a time. In most cases, this leads to the physically correct result of positive energy dissipation in the switches.
- Examples can be found, however, in which eqn. (4.3) still yields a negative value for the energy dissipated in a switch, even if only one switch is closed. It is conjectured that this is an indication that the circuit obtained by replacing the switch with a positive resistor would be unstable.
- If active elements are present in the circuit, the total power dissipation can depend on the order in which the switches are closed.

Formal statements of some of these results, and mathematical proofs of their validity, will be presented in the next section.

4.2 Energy conservation

In this section, some of the properties observed in the examples analyzed in the previous section will be stated formally and proven mathematically. The first step is to extend eqn. (4.3) to the case when v^+, the voltage across the element at time t_k^+, is not zero. The formula must be extended so that it reduces to eqn. (4.2) when $v^- = v^+$, and to eqn. (4.3) when $v^+ = 0$. These considerations lead to the following expression for the energy dissipation in an element:

$$J = \frac{v^- + v^+}{2} q. \tag{4.4}$$

The correctness of this formula is corroborated by the fact, which will be proven next, that the principle of energy conservation is always satisfied if the energy dissipated in each element at switching instant t_k is computed according to eqn. (4.4).

Consider a generic switching circuit containing c linear capacitors and m memoryless elements (switches, independent voltage sources, linear voltage-controlled voltage sources and ideal operational amplifiers); let n be the number of nodes in the circuit (excluding the ground node). Let $\mathbf{A}$ be the circuit's node-branch incidence matrix, with the branches numbered so that the first c columns of $\mathbf{A}$ correspond to the capacitors and the last m to the memoryless elements. Then $\mathbf{A}$ can be partitioned as:

$$\mathbf{A} = [\mathbf{A}_c \, \mathbf{A}_m],$$

where $\mathbf{A}_c$ and $\mathbf{A}_m$ are $n \times c$ and $n \times m$ matrices, respectively.

Let $\mathbf{q}_c = [q_1, \ldots, q_c]^T$ be a vector containing the charges stored in the capacitors, and let $\mathbf{q}_m = [q_{c+1}, \ldots, q_{c+m}]^T$ be another vector containing the charges that flow through the memoryless elements at switching instant t_k. Similarly, let $\mathbf{e}_c = [e_1, \ldots, e_c]^T$ and $\mathbf{e}_m = [e_{c+1}, \ldots, e_{c+m}]^T$ be vectors containing the branch voltages across the capacitors and the memoryless elements, respectively. Finally, let $\mathbf{v} = [v_1, \ldots, v_n]^T$ be the vector of the node voltages, and define:

$$\mathbf{q}^+ = \begin{bmatrix} \mathbf{q}_c^+ \\ \mathbf{q}_m \end{bmatrix}, \mathbf{e} = \begin{bmatrix} \mathbf{e}_c \\ \mathbf{e}_m \end{bmatrix}.$$

Theorem 1 *With the variable definitions given above, the following equality holds:*

$$\frac{1}{2}\mathbf{e}_c^{+T}\mathbf{q}_c^+ + \frac{1}{2}(\mathbf{e}_m^{+T} + \mathbf{e}_m^{-T})\mathbf{q}_m = \frac{1}{2}\mathbf{e}_c^{-T}\mathbf{q}_c^-, \tag{4.5}$$

where superscripts + and - denote variable values immediately before and after the switching instant, respectively.

Proof 1 *Charge conservation at each node implies that:*

$$\sum_j q_{c,j}^+ + \sum_j q_{m,j} = \sum_j q_{c,j}^-, \tag{4.6}$$

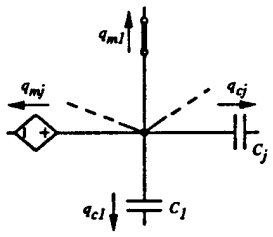

Figure 9: Charge Conservation Equations at a Node

where $q_{c,j}$ and $q_{m,j}$ represent the charge stored on the j-th capacitor, and the charge flowing through the j-th memoryless element, respectively (see Fig. 9). The set of n charge conservation equations can be written in vector form as:

$$\mathbf{A}\mathbf{q}^+ = \mathbf{A}_c\mathbf{q}_c^+ + \mathbf{A}_m\mathbf{q}_m = \mathbf{A}_c\mathbf{q}_c^-.$$

From this equation and from $\mathbf{e}^+ = \mathbf{A}^T\mathbf{v}^+$, one obtains:

$$\mathbf{e}_c^{+T}\mathbf{q}_c^+ + \mathbf{e}_m^{+T}\mathbf{q}_m = \mathbf{e}^{+T}\mathbf{q}^+ = \mathbf{v}^{+T}\mathbf{A}\mathbf{q}^+ = \mathbf{v}^{+T}\mathbf{A}_c\mathbf{q}_c^- = \mathbf{e}_c^{+T}\mathbf{q}_c^-. \tag{4.7}$$

Similarly:

$$\mathbf{e}_c^{-T}\mathbf{q}_c^+ + \mathbf{e}_m^{-T}\mathbf{q}_m = \mathbf{e}^{-T}\mathbf{q}^+ = \mathbf{v}^{-T}\mathbf{A}\mathbf{q}^+ = \mathbf{v}^{-T}\mathbf{A}_c\mathbf{q}_c^- = \mathbf{e}_c^-\mathbf{q}_c^-. \tag{4.8}$$

But:

$$\mathbf{e}_c^{+T}\mathbf{q}_c^- = \sum_j e_{c,j}^+ q_{c,j}^- = \sum_j e_{c,j}^+ C_j e_{c,j}^- = \sum_j q_{c,j}^+ e_{c,j}^- = \mathbf{e}_c^{-T}\mathbf{q}_c^+.$$

Therefore, combining eqns. (4.7) and (4.8) yields:

$$\mathbf{e}_c^{+T}\mathbf{q}_c^+ + \mathbf{e}_m^{+T}\mathbf{q}_m = \mathbf{e}_c^{-T}\mathbf{q}_c^- - \mathbf{e}_m^{-T}\mathbf{q}_m,$$

or, equivalently:

$$\mathbf{e}_c^{+T}\mathbf{q}_c^+ + (\mathbf{e}_m^{+T} + \mathbf{e}_m^{-T})\mathbf{q}_m = \mathbf{e}_c^{-T}\mathbf{q}_c^-,$$

which proves the theorem.

Note that $\frac{1}{2}\mathbf{e}_c^{+T}\mathbf{q}_c^+ = \sum_j \frac{1}{2}C_j(e_{c,j}^+)^2$. Therefore, this term represents the energy stored in the capacitors at t_k^+, i.e., immediately after switching instant t_k. Similarly, the term $\frac{1}{2}\mathbf{e}_c^{-T}\mathbf{q}_c^-$ is the energy stored in the capacitors at t_k^-. Finally, according to eqn. (4.4), the term $\frac{1}{2}(\mathbf{e}_m^{+T} + \mathbf{e}_m^{-T})\mathbf{q}_m$ represents the total energy absorbed by the memoryless elements at t_k. Thus, Theorem 1 states that using eqn. (4.4) to compute the energy dissipated by the memoryless elements satisfies the principle of conservation of energy.

The analysis of the network of Fig. 8(a) showed that, even if only one switch is closed, eqn. (4.3) could yield a negative value for the energy dissipated in the switch. It will be now proven that this cannot happen if the network obtained by replacing the switch with a positive resistor is stable. For this purpose, let $\mathbf{v}^-$ be the vector of the node voltages identifying the state of the switch-capacitor network of Fig. 10(a) before the switch is closed.

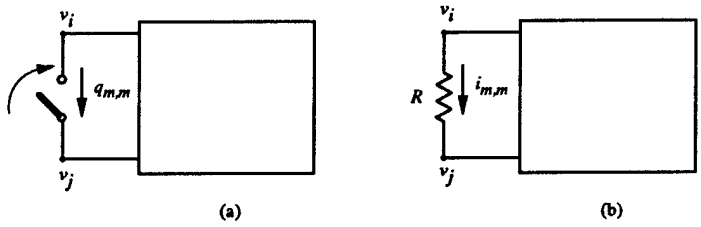

Figure 10: Replacing a switch with a resistor

It will be assumed that $\mathbf{v}^-$ is the quiescent bias point of the network of Fig. 10(a) when the switch is open. This means that $\mathbf{v}^-$ satisfies the branch equations of all the memoryless elements, except that of the closed switch. After the switch closes, the new state of the circuit can be found by writing a set of charge conservation equations, as in eqn. (4.6). Expressing the charges on the capacitors in terms of the node voltages, the charge conservation equations take on the following form:

$$\mathbf{C}\mathbf{v}^+ + \mathbf{A}_m\mathbf{q}_m = \mathbf{C}\mathbf{v}^-.$$

The set of branch constitutive equations for the memoryless elements can be written as:

$$\begin{bmatrix} \mathbf{b}_1^T \\ \mathbf{b}_2^T \\ \vdots \\ \mathbf{b}_m^T \end{bmatrix} \mathbf{v}^+ = \mathbf{B}^T\mathbf{v}^+ = \mathbf{s}.$$

Without loss of generality, it can be assumed that the last equation in this set, $\mathbf{b}_m^T\mathbf{v}^+ = s_m$, is the one for the closed switch ($v_i^+ - v_j^+ = 0$). Combining these two sets of equations together yields the following system [8, 15], which determines the state of the network at t_k^+:

$$\begin{bmatrix} \mathbf{C} & \mathbf{A}_m \\ \mathbf{B}^T & \end{bmatrix} \begin{bmatrix} \mathbf{v}^+ \\ \mathbf{q}_m \end{bmatrix} = \begin{bmatrix} \mathbf{C}\mathbf{v}^- \\ \mathbf{s} \end{bmatrix}. \tag{4.9}$$

For notational convenience, it has been assumed that $q_{m,m}$, the charge flowing through the switch when it closes, is the last entry in $\mathbf{q}_m$. Because $\mathbf{v}^-$ is the quiescent bias point of the network before the switch closes, it satisfies all but the last of the branch equations of the memoryless elements:

$$\mathbf{b}_i^T\mathbf{v}^- = s_i, \qquad i = 1, \ldots, m-1.$$

Consider now the network of Fig. 10(b), in which the switch has been replaced by a resistor, and assume that $\mathbf{v}^-$ is the state of the network at $t = 0$. The following theorem shows that the evolution of this network for $t > 0$ is related to the solution on eqn. (4.9).

Theorem 2 *There exists a constant α such that the voltages and currents in the network of Fig. 10(b) are given by:*

$$\begin{bmatrix} \mathbf{v}(t) \\ \mathbf{i}_m(t) \end{bmatrix} = \begin{bmatrix} \mathbf{v}^+ \\ \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{v}^- - \mathbf{v}^+ \\ -\alpha\mathbf{q}_m \end{bmatrix} e^{\alpha t}, \tag{4.10}$$

where $\mathbf{i}_m$ is the vector of the currents through the memoryless elements in the network.

Proof 2 *For $t > 0$, the evolution of the network of Fig. 10(b) is determined by the following systems af algebraic-differential equations:*

$$\begin{bmatrix} \mathbf{C} & \mathbf{A}_m \\ \mathbf{B}^T & -\mathbf{R} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{v}} \\ \mathbf{i}_m \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{s} \end{bmatrix}, \tag{4.11}$$

where $\mathbf{R} = diag(0, \ldots, 0, R)$. Using the expressions for $\mathbf{v}(t)$ and $\mathbf{i}_m(t)$ from eqn. (4.10), one obtains:

$$\mathbf{C}\dot{\mathbf{v}} + \mathbf{A}_m\mathbf{i}_m = \alpha\mathbf{C}(\mathbf{v}^- - \mathbf{v}^+)e^{\alpha t} - \alpha\mathbf{A}_m\mathbf{q}_m e^{\alpha t} = \mathbf{0},$$

or:

$$\mathbf{C}(\mathbf{v}^+ - \mathbf{v}^-) + \mathbf{A}_m\mathbf{q}_m = \mathbf{0}. \tag{4.12}$$

The second equation in (4.11) becomes:

$$\mathbf{B}^T\mathbf{v}^+ + \mathbf{B}^T(\mathbf{v}^- - \mathbf{v}^+)e^{\alpha t} + \alpha\mathbf{R}\mathbf{q}_m e^{\alpha t} = \mathbf{s},$$

which is equivalent to:

$$\mathbf{B}^T(\mathbf{v}^+ - \mathbf{v}^-) - \alpha\mathbf{R}\mathbf{q}_m = \mathbf{0}. \tag{4.13}$$

Thus $\mathbf{v}(t)$, $\mathbf{i}_m(t)$ given in (4.10) are the solution of eqn. (4.11) if and only if the following equation is satisfied:

$$\begin{bmatrix} \mathbf{C} & \mathbf{A}_m \\ \mathbf{B}^T & -\alpha\mathbf{R} \end{bmatrix} \begin{bmatrix} \mathbf{v}^+ - \mathbf{v}^- \\ \mathbf{q}_m \end{bmatrix} = \mathbf{D}(\alpha) \begin{bmatrix} \mathbf{v}^+ - \mathbf{v}^- \\ \mathbf{q}_m \end{bmatrix} = \mathbf{0}. \tag{4.14}$$

This equation has a nonzero solution if and only if $\det \mathbf{D}(\alpha) = 0$; because of the particular structure of $\mathbf{R}$, $\det \mathbf{D}(\alpha) = 0$ is a first-order polynomial equation in α (as can be seen by developing $\det \mathbf{D}(\alpha)$ along the last row of the matrix). Assuming that α is indeed the solution of this equation, it remains to verify that eqns. (4.12) and (4.13) are satisfied. Equation (4.12) is satisfied because it is equivalent to the first equation in (4.9). The first $m-1$ equations in (4.13) are satisfied, because:

$$\mathbf{b}_i^T\mathbf{v}^+ = s_i = \mathbf{b}_i^T\mathbf{v}^-, \qquad i = 1, \ldots, m-1.$$

This means that $n+m-1$ equations in (4.14), out of $n+m$, are satisfied. But $\det \mathbf{D}(\alpha) = 0$ implies that the equations in (4.14) are not independent. Therefore, if the first $n+m-1$ are satisfied, the last one must be, too.

Corollary 1 *If the network of Fig. 10(b) is stable, the energy dissipated in the memoryless elements between $t = 0$ and $t = +\infty$ is given by eqn. (4.4). In particular:*

$$\frac{1}{2}(v_i^+ - v_j^+)q_{m,m} = \int_0^{+\infty} Ri_{m,m}^2\, dt \geq 0,$$

that is, eqn. (4.4) yields a nonnegative value for the energy dissipated in the switch.

Proof 3 *As a consequence of Theorem 2, every branch voltage in the network of Fig. 10(b) has the following expression:*

$$v(t) = v^+ + (v^- - v^+)e^{\alpha t},$$

while the expressions for the branch currents through the memoryless elements are given by:

$$i(t) = -\alpha q e^{\alpha t}.$$

Therefore the energy dissipated in any memoryless element is:

$$J = \int_0^{+\infty} vi\,dt = \int_0^{+\infty} [v^+ + (v^- - v^+)e^{\alpha t}](-\alpha q e^{\alpha t})\,dt = v^+ q + (v^- - v^+)\frac{q}{2} = \frac{v^+ + v^-}{2} q.$$

In particular, if the element is resistor R, then $vi = Ri^2_{m,m}$, $v^+ = v_i^+ - v_j^+$, $v^- = 0$ *(because* v^- *represents the voltage across a closed switch), which completes the proof.*

Corollary 1 provides an additional proof that eqn. (4.4) satisfies the principle of conservation of energy, even if it is only valid when the network of Fig. 10(b) is stable. In this case, the corollary also shows that eqn. (4.4) will always yield a positive value for the power dissipated in a switch, if only one switch is closed at a time.

4.3 Power dissipation versus switching order

4.3.1 A graph problem

Theoretically, all the switches in a switched-capacitor circuit that are connected to the same clock phase close at the same instant. Of course, this is not the case in practice. The examples analyzed earlier show that the order in which the switches close can affect the circuit's power dissipation. In general, this order will be affected by factors, such as parasitic capacitances, length of clock lines, and so on, which are impossible to predict or control. As a consequence, it is impossible to predict with certainty the power dissipated by a switching circuit. A more realistic estimate can be obtained by trying to compute a range of values, within which the circuit's power dissipation must fall, regardless of the switching order. This leads to the problem described below.

At each clock phase, consider the switches that will close during that phase. Assuming that the switches close one at a time, determine the order which leads to the maximum (or minimum) power dissipation in the circuit.

This problem can be represented on a graph in the following way: let s be the number of switches connected to the clock phase under consideration. The state of each switch can be represented by a bit: for example, zero for an open switch, and one for a closed switch. Correspondingly, any configuration of the switches is represented by a binary string containing s bits. There are 2^s possible switch configurations which, represented as binary strings, form the vertices of an s-cube. Each cube edge corresponds to the closing of one

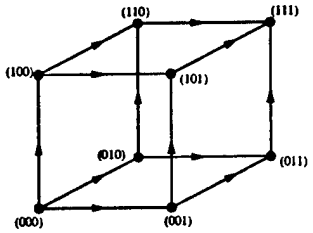

Figure 11: The graph representing all possible switching orders

switch; because it is assumed that the switches, once closed, do not reopen, each edge can be traversed in only one direction. Thus, the vertices and the edges of the s-cube form a directed graph, such as the one shown in Fig. 11.

Each edge in the graph can be assigned a weight, which is the power dissipated by the circuit when the switch corresponding to that edge is closed. Initially, all the switches are open, which is the configuration corresponding to the $(00\ldots0)$ vertex in the cube. Eventually, all the switches are closed, which means that the final configuration is the $(11\ldots1)$ vertex. Thus, finding the switching order that minimizes (maximizes) the circuit's power dissipation means finding the shortest (longest) path from $O = (00\ldots0)$ to $D = (11\ldots1)$ along the cube, where the length of a path is measured as the sum of the weights of the edges contained in the path: in graph theory, this is known as the *single-source shortest paths* problem. This is a well-known problem, and a number of algorithms exist to solve it [16]. Because the graph arising in this specific problem is obviously acyclic, the simplest way to find the shortest and longest paths from O to D is to traverse the graph in topological order, in which each vertex is visited only after all its predecessors have been visited. This traversal is made even simpler by the structure of this particular graph, and can be implemented using a queue, as shown in Fig. 12. The quantity $W(v_1, v_2)$ is the weight of the edge joining

```
Place O = (00...0) on Queue;
while Queue ≠ ∅ {
  Longest(v, O) = max{W(v, p) + Longest(p, O) : p is a predecessor of v};
  Shortest(v, O) = min{W(v, p) + Shortest(p, O) : p is a predecessor of v};
  foreach successor s of v {
    Compute W(s, v);
    Add v to Queue;
  }
}
```

Figure 12: Computation of shortest and longest paths

vertices v_1 and v_2, that is, the power dissipated by the circuit when the switch corresponding to that edge is closed. Obviously, D is the last vertex to be visited, and $Longest(O, D)$ and

Shortest(O, D) give, respectively, the maximum and minimum power that can be dissipated by the circuit under all possible switching orders.

4.3.2 Computing the edge weights

The determination of the edge weights requires computing the values of the voltages and charges in the circuit when a particular switch is closed. This means that a system of linear equations must be solved for each edge in the graph; since there are $s2^{s-1}$ edges in an s-cube, in principle this would entail performing $s2^{s-1}$ matrix LU decompositions. By exploiting the particular structure of the problem, the number of LU decompositions can be reduced, as explained below.

Each vertex in the s-cube is identified by a certain set of closed switches; an edge leading out of that vertex corresponds to closing one of the remaining open switches. The new state of the circuit, after the switch's closure, is determined by the solution of eqn. (4.9). If the last unknown is $q_{m,m}$ (the charge through the closing switch), and the last equation is the one for the closed switch, then the set of equations in (4.9) has the following form:

$$\begin{bmatrix} & & \vdots \\ & & 1 \\ & \mathbf{Y} & \vdots \\ & & -1 \\ & & \vdots \\ \cdots & 1 \cdots -1 & \cdots \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ q_{m,m} \end{bmatrix} = \begin{bmatrix} \mathbf{r}_0 \\ 0 \end{bmatrix}, \tag{4.15}$$

where entries 1 and -1 in the last row and column of the coefficient matrix corresponds to nodes i and j (see Fig. 10(a)).

The coefficient matrix in eqn. (4.15) has the form:

$$\begin{bmatrix} \mathbf{Y} & \mathbf{v} \\ \mathbf{w}^T & z \end{bmatrix}.$$

If $\mathbf{Y} = \mathbf{LU}$ is the LU decomposition of $\mathbf{Y}$, it is immediate to verify that:

$$\begin{bmatrix} \mathbf{Y} & \mathbf{v} \\ \mathbf{w}^T & z \end{bmatrix} = \begin{bmatrix} \mathbf{L} & \\ \mathbf{l}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{U} & \mathbf{u} \\ & t \end{bmatrix}, \tag{4.16}$$

where:

$$\begin{aligned} \mathbf{Lu} &= \mathbf{v} \\ \mathbf{U}^T\mathbf{l} &= \mathbf{w} \\ t &= z - \mathbf{l}^T\mathbf{u} \end{aligned}$$

Then the solution of eqn. (4.15) can be computed by forward elimination and back substitution:

$$\begin{aligned} \mathbf{L}\mathbf{x}_0 &= \mathbf{r}_0 \\ q_0 &= -\mathbf{l}^T\mathbf{x}_0 \\ q_{m,m} &= q_0/t \\ \mathbf{U}\mathbf{x} &= \mathbf{x}_0 - q_{m,m}\mathbf{u}. \end{aligned}$$

For a given cube vertex, each outgoing edge corresponds to closing a different switch: everything else in the circuit remains unchanged. This means that the specific switch being closed will affect only the last row and column of the coefficient matrix in eqn. (4.15). As a consequence, the LU decompositions of all the matrices corresponding to the edges leading out of the same vertex can be performed through eqn. (4.16). This reduces the number of LU decompositions to be performed to one per vertex, that is, 2^s for an s-cube.

4.4 Numerical results

The described in the previous section was used to calculate the power dissipation of a number of switched-capacitor circuits of varying sizes. The circuits ranged from a simple amplifier to various types of elliptic filters. The clock frequency for each of these circuit was taken to be 1 MHz. The results of these simulations can be found in Table 3: the second column shows the dimension of eqn. (4.5) for the corresponding circuit, while the the third and fourth columns report the minimum and maximum power dissipation, respectively. The CPU times represent the running times of the simulator on a Sparcstation 5/20 running SunOS 4.1.4.

It is worth pointing out that the figures reported in Table 3 represent only the *switching* power dissipation of the circuit, that is, the power dissipated in charging and discharging the capacitors. In a physical circuit, an additional source of power dissipation is the static power absorbed by the operational amplifiers, which is not included in the ideal op-amp model. The computation of this power is trivial, as it is equal to the supply voltage times the bias current drawn by the operational amplifiers. Adding this value to the switching power given in Table 3 yields the total power dissipation of the circuit.

4.5 Summary

This chapter has explored a number of issues related to the evaluation of the power dissipated in a switching circuit. It has been shown that the formula normally used to compute the power dissipated in an element can be extended to the case when the current through the element is a Dirac impulse, and the voltage across the element is discontinuous. It has also been proven that such extension satisfies the principle of conservation of energy.

The analysis of a number of selected switched-capacitor circuits has shown that caution must be exercised in the computation of power dissipation, because the assumption that two

Table 3: Effect of Switching Order on Power Dissipation

Circuit	Matrix size	Min. power (μW)	Max power (μW)
SCAmp	14	1.1	1.3
Biquad	44	138	151
Ellipt0	96	9.34	15.5
Ellipt1	60	5.2	7.6
Ellipt2	66	.29	1.34

or more switches close at the same time can lead to nonphysical results. This difficulty does not occur if the switches are closed one at a time; in this case, however, it has been shown that the power dissipation is affected by the switching order. This observation raises the question of how to compute the maximum and minimum power that can be dissipated by a switching circuit.

It has been demonstrated that this is equivalent to the problem of finding the shortest and longest paths between two points in a weighted directed graph (the so-called single-source shortest paths problem). Because the graph that arises in this particular instance is acyclic, the shortest and longest paths can be found simply by traversing the graph in topological order. Most of the computational effort is spent in determining the edge weights, which correspond to the power dissipated in the circuit when a particular switch is closed. The computation of each edge weight requires the solution of a system of linear equations, and it has been shown that the computational effort can be reduced by exploiting the particular structure of the coefficient matrices of those systems. Finally, the algorithm has been tested on a number of switched-capacitor circuits.

Chapter 5

Power Estimation in Continuous-Time Circuits

5.1 Mathematical Framework

The behavior of a time-invariant lumped-element circuit is completely characterized by the following Modified Nodal Analysis (MNA) equations [17]:

$$\dot{\mathbf{x}} = -\mathbf{f}(\mathbf{x}) + \mathbf{u}, \tag{5.1}$$

where $\mathbf{x}$ is the state vector (capacitor charges and inductor fluxes) and $\mathbf{u}$ is the vector of inputs to the circuit (represented by independent sources). The goal is to obtain information about the statistics of $\mathbf{x}$ (e.g., its mean and covariance matrix) from corresponding information about $\mathbf{u}$. For this purpose, the inputs to the circuit will be modeled as a Wiener process with mean 0 (if the mean of the inputs is not 0, we write $\mathbf{u} = \bar{\mathbf{u}} + \hat{\mathbf{u}}$, where $\bar{\mathbf{u}}$ is the mean of $\mathbf{u}$, and we define a new function $\hat{\mathbf{f}}(\mathbf{x}) = \mathbf{f}(\mathbf{x}) + \bar{\mathbf{u}}$). Under these assumptions, it can be shown [10] that $\mathbf{x}(t)$ is a realization of a Markov process whose probability density $p(\mathbf{x}, t)$ is a solution of the Fokker-Plank equation:

$$\frac{\partial p}{\partial t} = \nabla \cdot \left[\mathbf{f}(\mathbf{x})p + \frac{1}{2} \nabla \cdot (\mathbf{Q}p) \right].$$

Because we are only interested in the stationary distribution of $\mathbf{x}$, $\frac{\partial p}{\partial t} = 0$ and the above equation becomes:

$$0 = \nabla \cdot \left[\mathbf{f}(\mathbf{x})p + \frac{1}{2} \nabla \cdot (\mathbf{Q}p) \right]. \tag{5.2}$$

In general, the solution $p(\mathbf{x})$ of the above equation cannot be expressed in closed form. However, an approximate solution can be computed numerically as explained below. The method is better understood by examing first the simple case when $\mathbf{f}$ is a linear function of the form $\mathbf{f}(\mathbf{x}) = \mathbf{A}\mathbf{x}$. Then eqn. (5.2) becomes:

$$\nabla \cdot \left[\mathbf{A}\mathbf{x}p + \frac{1}{2} \nabla \cdot (\mathbf{Q}p) \right] = 0, \tag{5.3}$$

or, in short:

$$\mathcal{F}p = 0,$$

where $\mathcal{F}$ denotes the *Fokker-Plank operator* on the left-hand side of eqn. (5.3). Let $w_0(\mathbf{x}) = e^{-\phi(\mathbf{x})}$, where $\phi(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{P}^{-1}\mathbf{x}$, and $\mathbf{P}$ is the solution of the equation:

$$\mathbf{AP} + \mathbf{PA}^T = \mathbf{Q}. \tag{5.4}$$

It can be shown that $w_0(\mathbf{x})$ satisfies eqn. (5.3). Therefore $p(\mathbf{x}) = c_0 w_0(\mathbf{x})$, with c_0 chosen so that $\int p(\mathbf{x})\, d\mathbf{x} = 1$, is the solution of eqn. (5.2) when $\mathbf{f}(\mathbf{x}) = \mathbf{Ax}$, i.e., when eqn. (5.1) is linear. This shows that *the probability distribution of all voltages and currents in a linear circuit can be expressed in closed form.* In particular, the probability distribution is gaussian with covariance matrix $\mathbf{P}$. Thus, the statistical distribution of all voltages and currents in a linear circuit is determined completely by the solution of eqn. (5.4), which is a set of linear algebraic equations in the entries of $\mathbf{P}$. Numerous methods for the solution of eqn. (5.4) have been published in the literature [18, 19].

If the circuit is nonlinear, the solution $p(\mathbf{x})$ of eqn. (5.2) cannot be expressed in closed form. However, relying on the knowledge of the expression for $p(\mathbf{x})$ found in the linear case, an approximate solution can be computed numerically in the following way. Let:

$$\mathbf{f}(\mathbf{x}) = \mathbf{A}(\mathbf{x} - \mathbf{x}_0) + \mathbf{r}(\mathbf{x}),$$

be a first-order series expansion of $\mathbf{f}$ around $\mathbf{x}_0$, where $\mathbf{A} = \left.\dfrac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}_0}$ and $\mathbf{x}_0$ is a solution of the equation $\mathbf{f}(\mathbf{x}) = \mathbf{0}$. Without loss of generality, we can assume that $\mathbf{x}_0 = \mathbf{0}$ (otherwise it can be made so by a translation of the reference system, which leaves eqn. (5.2) unchanged). Then eqn. (5.2) can be rewritten as:

$$\nabla \cdot \left[\mathbf{A}\mathbf{x}p + \frac{1}{2}\nabla \cdot (\mathbf{Q}p)\right] = -\nabla \cdot [\mathbf{r}(\mathbf{x})p], \tag{5.5}$$

or, in short:

$$\mathcal{F}p = -\nabla \cdot [\mathbf{r}(\mathbf{x})p]. \tag{5.6}$$

An approximate solution of eqn. (5.6) can be computed by using a series expansion for $p(\mathbf{x})$ [20]:

$$p(\mathbf{x}) = \sum_n c_n w_n(\mathbf{x}) = \sum_n c_n w_0(\mathbf{x}) h_n(\mathbf{x}), \tag{5.7}$$

where the functions $h_n(\mathbf{x})$ are Hermite polynomials of increasing order. It can be shown that the functions $w_n(\mathbf{x})$ are eigenvectors of the Fokker-Plank operator $\mathcal{F}$, i.e., that:

$$\mathcal{F}w_n = \lambda_n w_n.$$

Therefore substituting the series expansion from eqn. (5.7) in eqn. (5.6) the following equation is obtained:

$$\sum_n c_n \lambda_n w_n(\mathbf{x}) = -\nabla \cdot [\mathbf{r}(\mathbf{x})p].$$

The coefficients c_n must satisfy the following relationships [20]:

$$c_n \lambda_n = -\int \nabla \cdot [\mathbf{r}(\mathbf{x})p] v_n(\mathbf{x})\, d\mathbf{x}, \tag{5.8}$$

where the functions $v_n(\mathbf{x})$ are the eigenvectors of the adjoint of the Fokker-Plank operator $\mathcal{F}$:

$$\mathcal{F}^T v_n = \lambda_n v_n.$$

Using Stokes' theorem, it can be shown that:

$$-\int \nabla \cdot [\mathbf{r}(\mathbf{x})p] v_n(\mathbf{x})\, d\mathbf{x} = \int p < \mathbf{r}(\mathbf{x}), \nabla v_n(\mathbf{x}) > \, d\mathbf{x},$$

so that eqn. (5.8) can be rewritten in the following form:

$$\begin{aligned} c_n \lambda_n &= \int p < \mathbf{r}(\mathbf{x}), \nabla v_n(\mathbf{x}) > \, d\mathbf{x} = \\ &= \sum_k c_k \int w_k(\mathbf{x}) < \mathbf{r}(\mathbf{x}), \nabla v_n(\mathbf{x}) > \, d\mathbf{x} = \sum_k c_k a_{kn}. \end{aligned} \tag{5.9}$$

The values of the coefficients a_{kn} can be computed by numerical integration:

$$a_{kn} = \int w_k(\mathbf{x}) < \mathbf{r}(\mathbf{x}), \nabla v_n(\mathbf{x}) > \, d\mathbf{x}.$$

If the series expansion for $p(\mathbf{x})$ is truncated after a finite number of terms, eqn. (5.9) translates into a set of linear equations:

$$\begin{aligned} \lambda_0 c_0 &= a_{00}c_0 + a_{10}c_1 + \ldots + a_{N0}c_N \\ \lambda_1 c_1 &= a_{01}c_0 + a_{11}c_1 + \ldots + a_{N1}c_N \\ &\vdots \\ \lambda_N c_N &= a_{0N}c_0 + a_{1N}c_1 + \ldots + a_{NN}c_N \end{aligned}$$

The solution to this set of linear equations, together with the requirement that $\int p(\mathbf{x})\, d\mathbf{x} = 1$, determines the values of the coefficients c_n. Once the coefficients c_n are known, it is easy to obtain any statistical parameter of the random variable $\mathbf{x}$. For instance, the mean $\bar{\mathbf{x}}$ can be computed by evaluating the following integral numerically:

$$\bar{\mathbf{x}} = \int \mathbf{x} p(\mathbf{x})\, d\mathbf{x} = \sum_n c_n \int \mathbf{x} w_n(\mathbf{x})\, d\mathbf{x}$$

(this value of $\bar{\mathbf{x}}$ should be interpreted as a *correction* to $\mathbf{x}_0$, which gives a first-cut approximation of the mean of $\mathbf{x}$, based on a linear approximation of $\mathbf{f}(\mathbf{x})$).

In conclusion, *even when the circuit is nonlinear, the computation of the approximate statistical distribution of the circuit voltages and currents can be performed in a straightforward way.* In addition to the computation of the matrix $\mathbf{P}$, the only other required operations are numerical quadrature (to evaluate integrals) and the solution of an additional set of linear equations to compute the coefficients c_n. Therefore, the method described above is an efficient and reliable approach to the problem of power estimation in analog circuits.

5.2 Numerical Solution of the Fokker-Plank Equation

Given the ordinary differential equation:

$$\dot{\mathbf{x}} = -\mathbf{A}\mathbf{x} + \mathbf{u}, \tag{5.10}$$

the associated stationary Fokker-Plank equation is:

$$\nabla \cdot \left[\mathbf{A}\mathbf{x}p + \frac{1}{2}\nabla \cdot (\mathbf{Q}p)\right] = 0. \tag{5.11}$$

We want to find the eigenvalues, λ_n, and the associated eigenfunctions, p_n, of this equation, that is, the solutions of the following partial differential equation:

$$\nabla \cdot \left[\mathbf{A}\mathbf{x}p_n + \frac{1}{2}\nabla \cdot (\mathbf{Q}p_n)\right] = \lambda_n p_n. \tag{5.12}$$

We will show that the eigenfunctions can be expressed as:

$$p_n(\mathbf{x}) = e^{-\psi(\mathbf{x})} H_n(\mathbf{v}^T\mathbf{x}), \tag{5.13}$$

where $\psi(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{P}^{-1}\mathbf{x}$, $\mathbf{P}$ is the solution of the Lyapunov equation:

$$\mathbf{A}\mathbf{P} + \mathbf{P}\mathbf{A}^T = \mathbf{Q}, \tag{5.14}$$

and $H_n(\cdot)$ is an Hermite polynomial of order n.

Let $\phi_n(\mathbf{x}) = H_n(\mathbf{v}^T\mathbf{x})$, so that $p_n = e^{-\psi}\phi_n$. Then:

$$\begin{aligned}\nabla \cdot p = -\phi_n e^{-\psi}\mathbf{P}^{-1}\mathbf{x} + e^{-\psi}\nabla \cdot \phi_n = \\ -p_n\mathbf{P}^{-1}\mathbf{x} + e^{-\psi}\nabla \cdot \phi_n.\end{aligned} \tag{5.15}$$

Since $\nabla \cdot (\mathbf{Q}p_n) = \mathbf{Q}\nabla \cdot p_n$, we have:

$$\begin{aligned}&\nabla \cdot \left[\mathbf{A}\mathbf{x}p_n + \frac{1}{2}\nabla \cdot (\mathbf{Q}p_n)\right] = \\ &\quad \nabla \cdot \left[\mathbf{A}\mathbf{x}p_n - \frac{1}{2}p_n\mathbf{Q}\mathbf{P}^{-1}\mathbf{x} + \frac{1}{2}e^{-\psi}\mathbf{Q}\nabla \cdot \phi_n\right] = \\ &\quad \nabla \cdot \left[p_n\left(\mathbf{A} - \frac{1}{2}\mathbf{Q}\mathbf{P}^{-1}\right)\mathbf{x}\right] + \frac{1}{2}\nabla \cdot \left[e^{-\psi}\mathbf{Q}\nabla \cdot \phi_n\right].\end{aligned} \tag{5.16}$$

But $\nabla \cdot (f\mathbf{w}) = < \nabla \cdot f, \mathbf{w} > + f\nabla \cdot \mathbf{w}$, therefore:

$$\begin{aligned}&\nabla \cdot \left[p_n\left(\mathbf{A} - \frac{1}{2}\mathbf{Q}\mathbf{P}^{-1}\right)\mathbf{x}\right] = \\ &\quad < -\phi_n e^{-\psi}\mathbf{P}^{-1}\mathbf{x} + e^{-\psi}\nabla \cdot \phi_n, \left(\mathbf{A} - \frac{1}{2}\mathbf{Q}\mathbf{P}^{-1}\right)\mathbf{x} > + p_n \mathrm{tr}\left(\mathbf{A} - \frac{1}{2}\mathbf{Q}\mathbf{P}^{-1}\right).\end{aligned} \tag{5.17}$$

But $< \phi_n e^{-\psi} \mathbf{P}^{-1}\mathbf{x}, \left(\mathbf{A} - \frac{1}{2}\mathbf{Q}\mathbf{P}^{-1}\right)\mathbf{x} >= 0$, because:

$$< \mathbf{P}^{-1}\mathbf{x}, \left(\mathbf{A} - \frac{1}{2}\mathbf{Q}\mathbf{P}^{-1}\right) > \;=\; < \mathbf{x}, \left(\mathbf{P}^{-1}\mathbf{A} - \frac{1}{2}\mathbf{P}^{-1}\mathbf{A}\mathbf{P}^{-1}\right)\mathbf{x} > \qquad (5.18)$$

$$< \mathbf{x}, \left(\mathbf{P}^{-1}\mathbf{A} - \frac{1}{2}\mathbf{P}^{-1}\mathbf{A}\mathbf{P}^{-1}\right)\mathbf{x} > \;=\; < \mathbf{x}, \left(\mathbf{A}^T\mathbf{P}^{-1} - \frac{1}{2}\mathbf{P}^{-1}\mathbf{Q}\mathbf{P}^{-1}\right)\mathbf{x} > . \qquad (5.19)$$

Combining eqns. (5.18) and (5.19), one obtains:

$$< \mathbf{P}^{-1}\mathbf{x}, \left(\mathbf{A} - \frac{1}{2}\mathbf{Q}\mathbf{P}^{-1}\right) >= \frac{1}{2} < \mathbf{x}, \left(\mathbf{P}^{-1}\mathbf{A} + \mathbf{A}^T\mathbf{P}^{-1} - \mathbf{P}^{-1}\mathbf{Q}\mathbf{P}^{-1}\right)\mathbf{x} > .$$

But:

$$\mathbf{P}^{-1}\mathbf{A} + \mathbf{A}^T\mathbf{P}^{-1} - \mathbf{P}^{-1}\mathbf{Q}\mathbf{P}^{-1} = \mathbf{P}^{-1}(\mathbf{A}\mathbf{P} + \mathbf{P}\mathbf{A}^T - \mathbf{Q})\mathbf{P}^{-1} = 0,$$

which proves the claim.

Similarly, $\mathrm{tr}\left(\mathbf{A} - \frac{1}{2}\mathbf{Q}\mathbf{P}^{-1}\right) = 0$, because:

$$\begin{aligned}
\mathrm{tr}\left(\mathbf{A} - \frac{1}{2}\mathbf{Q}\mathbf{P}^{-1}\right) &= \mathrm{tr}\left(\mathbf{A}\mathbf{P} - \frac{1}{2}\mathbf{Q}\right)\mathrm{tr}(\mathbf{P}^{-1}) \\
\mathrm{tr}(\mathbf{A}\mathbf{P}) &= \mathrm{tr}(\mathbf{P}\mathbf{A}^T) \\
\mathrm{tr}\left(\mathbf{A}\mathbf{P} - \frac{1}{2}\mathbf{Q}\right) &= \frac{1}{2}\mathrm{tr}(2\mathbf{A}\mathbf{P} - \mathbf{Q}) = \\
&\quad \frac{1}{2}\mathrm{tr}(\mathbf{A}\mathbf{P} + \mathbf{P}\mathbf{A}^T - \mathbf{Q}) = 0.
\end{aligned}$$

As a consequence, eqn. (5.16) becomes:

$$\nabla \cdot \left[\mathbf{A}\mathbf{x}p_n + \frac{1}{2}\nabla \cdot (\mathbf{Q}p_n)\right] =$$

$$e^{-\psi} < \nabla \cdot \phi_n, \left(\mathbf{A} - \frac{1}{2}\mathbf{Q}\mathbf{P}^{-1}\right)\mathbf{x} > -\frac{1}{2}e^{-\psi} < \mathbf{P}^{-1}\mathbf{x}, \mathbf{Q}\nabla \cdot \phi_n > +$$

$$\frac{1}{2}e^{-\psi}\nabla \cdot [\mathbf{Q}\nabla \cdot \phi_n] =$$

$$e^{-\psi} < \nabla\phi_n, \left(\mathbf{A} - \mathbf{Q}\mathbf{P}^{-1}\right)\mathbf{x} > +\frac{1}{2}e^{-\psi}\nabla \cdot (\mathbf{Q}\nabla \cdot \phi_n) . \qquad (5.20)$$

But: $\mathbf{A} - \mathbf{Q}\mathbf{P}^{-1} = (\mathbf{A}\mathbf{P} - \mathbf{Q})\mathbf{P}^{-1} = -\mathbf{P}\mathbf{A}^T\mathbf{P}^{-1}$, so that:

$$\nabla \cdot \left[\mathbf{A}\mathbf{x}p_n + \frac{1}{2}\nabla \cdot (\mathbf{Q}p_n)\right] =$$

$$-e^{-\psi} < \mathbf{P}^{-1}\mathbf{A}\mathbf{P}\nabla \cdot \phi_n, \mathbf{x} > +\frac{1}{2}e^{-\psi}\nabla \cdot (\mathbf{Q}\nabla \cdot \phi_n) , \qquad (5.21)$$

and eqn. (5.12) becomes:

$$\frac{1}{2}e^{-\psi}\nabla \cdot (\mathbf{Q}\nabla \cdot \phi_n) - e^{-\psi} < \mathbf{P}^{-1}\mathbf{A}\mathbf{P}\nabla \cdot \phi_n, \mathbf{x} >= \lambda_n e^{-\psi}\phi_n,$$

or, equivalently:

$$\frac{1}{2}\nabla \cdot (\mathbf{Q}\nabla \cdot \phi_n) - < \mathbf{P}^{-1}\mathbf{A}\mathbf{P}\nabla \cdot \phi_n, \mathbf{x} >= \lambda_n \phi_n. \tag{5.22}$$

Since $\phi_n(\mathbf{x}) = H_n(< \mathbf{v}, \mathbf{x} >)$, we have:

$$\begin{aligned} \nabla \cdot \phi_n &= \mathbf{v}H_n' \\ \nabla \cdot (\mathbf{Q}\nabla \cdot \phi_n) &= < \mathbf{Q}\mathbf{v}, \mathbf{v} > H_n''. \end{aligned}$$

Let $\mathbf{v}$ be an eigenvector of $\mathbf{P}^{-1}\mathbf{A}\mathbf{P}$, namely: $\mathbf{P}^{-1}\mathbf{A}\mathbf{P}\mathbf{v} = \mu\mathbf{v}$. Then eqn. (5.22) can be rewritten in the following way:

$$\frac{1}{2} < \mathbf{Q}\mathbf{v}, \mathbf{v} > H_n'' - \mu < \mathbf{v}, \mathbf{x} > H_n' = \lambda_n H_n,$$

or, equivalently:

$$H_n'' - 2\frac{\mu}{< \mathbf{Q}\mathbf{v}, \mathbf{v} >} < \mathbf{v}, \mathbf{x} > H_n' - 2\frac{\lambda_n}{< \mathbf{Q}\mathbf{v}, \mathbf{v} >} H_n = 0. \tag{5.23}$$

Since the n-th Hermite ploynomial satisfies the differential equation:

$$H_n''(t) - 2tH_n'(t) + 2nH_n(t) = 0,$$

the function $p_n(\mathbf{x}) = e^{-\psi(\mathbf{x})}H_n(< \mathbf{v}, \mathbf{x} >)$ is an eigenfunction of the Fokker-Plank equation if we choose:

$$\begin{aligned} < \mathbf{Q}\mathbf{v}, \mathbf{v} > &= \mu \\ \lambda_n &= -n\mu. \end{aligned}$$

Appendix

1 Sample SKILL File for HNL Netlister

```
; This function prints out the line for an NMOSfet element.  It is used by
; the element 'nmos' in sample circuit.

procedure( hnlCktsimPrintNMOSfetElement()
  let( ( tmpString0 )
    sprintf( tmpString0
            "   nm%s: nmos generic map (%s, %s)"
            hnlMapInstName( hnlCurrentInstName )
            hnlCktsimGetPropVal( "w" )
            hnlCktsimGetPropVal( "l" )
           )
    hnlPrintString( strcat( tmpString0 "\n" ) )
    t
  )
  let( ( tmpString )
    sprintf( tmpString
            "\t   port map (D => %s, G => %s, S => %s, B => %s);"
             hnlCktsimNetOnTerm( "D" 0 )
             hnlCktsimNetOnTerm( "G" 0 )
             hnlCktsimNetOnTerm( "S" 0 )
             hnlCktsimNMOSBulkNetName
           )
    hnlPrintString( strcat( tmpString "\n" ) )
    t
  )
)


; This function prints out the model statement for a mosfet model.  It is
; used by elements 'nmos', 'pmos' in the sample circuit.
```

```
procedure( hnlCktsimPrintMOSfetModel()
;  hnlPrintString("mosfet model\n")
;  let( ( tmpString )
;    if( hnlIncrementalMode
;    then sprintf( tmpString ".model %s %s"
;                  hnlCktsimUniqueBlockName( hnlCurrentMaster )
;                  hnlCurrentMaster~>modelType )
;    else sprintf( tmpString  ".model %s %s"
;                  hnlCktsimUniqueBlockName( hnlCurrentMaster )
;                  hnlCurrentMaster~>modelType )
;    )
;    hnlCktsimSprintf( tmpString "%s gamma=%s" tmpString
;                      hnlCktsimGetPropVal( "gamma" ) )
;    hnlCktsimSprintf( tmpString "%s lambda=%s" tmpString
;                      hnlCktsimGetPropVal( "lambda" ) )
;
;    if( hnlIncrementalMode
;    then fprintf( hnlIncludeFile strcat( tmpString "\n" ) )
;    else hnlPrintString( strcat( tmpString "\n" ) )
;    )
     t
;  )
)


; This procedure prints out the line for a PMOSfet element.  It is used by
; the element 'pmos' in the sample circuit.
;   hnlMapInstName( hnlCurrentInstName )
procedure( hnlCktsimPrintPMOSfetElement()
  let( ( tmpString0 )
    sprintf( tmpString0
            "   pm%s: pmos generic map (%s, %s)"
            hnlMapInstName( hnlCurrentInstName )
            hnlCktsimGetPropVal( "w" )
            hnlCktsimGetPropVal( "l" )
           )
    hnlPrintString( strcat( tmpString0 "\n" ) )
    t
  )

  let( ( tmpString )
```

```
    sprintf( tmpString
            "\t   port map (D => %s, G => %s, S => %s, B => %s);"
             hnlCktsimNetOnTerm( "D" 0 )
             hnlCktsimNetOnTerm( "G" 0 )
             hnlCktsimNetOnTerm( "S" 0 )
             hnlCktsimPMOSBulkNetName
           )
    hnlPrintString( strcat( tmpString "\n" ) )
    t
  )
)


; This procedure prints out the line for a resistor element.  It is used by
; the element "resistor" in the sample circuit.

procedure( hnlCktsimPrintResistorElement()
  let( ( tmpString )
    sprintf( tmpString
          "   %s: resistor generic map (%s)\n\t   port map (p => %s, m => %s);"
             hnlMapInstName( hnlCurrentInstName )
             hnlCktsimGetPropVal( "r" )
             hnlCktsimNetOnTerm( "PLUS" 0 )
             hnlCktsimNetOnTerm( "MINUS" 0 )
           )
    hnlCktsimSprintf( tmpString "%s" tmpString )
    hnlPrintString( strcat( tmpString "\n" ) )
    t
  )
)


; This procedure prints out the line for a capacitor element.  It is used
; by the element 'capacitor' in the sample circuit.

procedure( hnlCktsimPrintCapacitorElement()
  let( ( tmpString )
    sprintf( tmpString
          "   %s: capacitor generic map (%s)\n\t   port map (p => %s, m => %s);"
             hnlMapInstName( hnlCurrentInstName )
             hnlCktsimGetPropVal( "c" )
             hnlCktsimNetOnTerm( "PLUS" 0 )
```

```
            hnlCktsimNetOnTerm( "MINUS" 0 )
          )
    hnlCktsimSprintf(tmpString "%s" tmpString)
    hnlPrintString( strcat( tmpString "\n" ) )
    t
  )
)


procedure( hnlCktsimPrintVCIElement()
  let( ( tmpString )
    sprintf( tmpString
    "   %s: VCI generic map (%s)\n\t   port map (p => %s, m => %s, cp = > %s, cm => %s);
          hnlMapInstName( hnlCurrentInstName )
          hnlCktsimGetPropVal( "G" )
          hnlCktsimNetOnTerm( "p" 0 )
          hnlCktsimNetOnTerm( "m" 0 )
          hnlCktsimNetOnTerm( "cp" 0 )
          hnlCktsimNetOnTerm( "cm" 0 )
        )
    hnlCktsimSprintf(tmpString "%s" tmpString)
    hnlPrintString( strcat( tmpString "\n" ) )
    t
  )
)
```

2 LEX Input File for VHDL-A

```
%{
/************** VHDL-A scanner in LEX format **********
 *
 * Derived from a scanner of the ALLIANCE CAD toolset,
 * release 1.1, written by:
 *
 *   MASI/CAO-VLSI CAD Team
 *   Laboratoire MASI/CAO-VLSI
 *   Tour 55-65, 2eme etage, Porte 13
 *   Universite Pierre et Marie Curie (PARIS VI)
 *   4, place Jussieu 75252 PARIS Cedex 05, FRANCE
 *
 * and further modified by:
```

```
 *
 * Thomas Dettmer
 * Dortmund University
 * Dept. of Computer Scienc, LS1
 * PB 500 500
 * D-44221 Dortmund (Germany)
 * Phone: +49-231-755-6464
 * e-mail: dettmer@ls1.informatik.uni-dortmund.de
 *
 ************************************************************
 *
 * The original file, obtained from pub/src/VHDL/Grammar at
 * ftp.cs.utwente.nl contained the following notices:
 *
 * This file is not intended to be used for commercial purposes
 * without permission of the University of Dortmund
 *
 * NOTE THAT THERE IS NO WARRANTY FOR CORRECTNES, COMPLETENESS,
 * SUPPORT OR ANYTHING ELSE.
 ************************************************************/

#include "vhdllex.h"

%}

UCLETTER [A-Z]
DIGIT [0-9]
SPECCHAR [\#\&\'\(\)\*\+\,\-\.\/\:\;\<\=\>\_\|]
SPACE [ \t]
FEFFECT [\t\v\r\l\f]
EOL \n
LCLETTER [a-z]
OTHERCHAR [\!\$\@\?\[\\\]\^\`\{\}\~]

GRCHAR ({BASEGRCHAR}|{LCLETTER}|{OTHERCHAR})
BASEGRCHAR ({UCLETTER}|{DIGIT}|{SPECCHAR}|{SPACE})
LETTER ({UCLETTER}|{LCLETTER})
LETTORDIGIT ({LETTER}|{DIGIT})
DLIT {INT}(\.{INT})?({EXP})?
INT {DIGIT}(_?{DIGIT})*
EXP ([eE][-+]?{INT})
BASE {INT}
```

```
BINT {EDIGIT}(_?{EDIGIT})*
EDIGIT ({DIGIT}|[a-fA-F])
BSPEC (B|b|O|o|X|x)

%%

{SPACE} { /* nothing */ }
\& { ECHO; return(T_Ampersand); }
\' { ECHO; return(T_Apostrophe); }
\( { return(T_LeftParen); }
\) { return(T_RightParen); }
"**" { ECHO; return(T_DoubleStar); }
\* { ECHO; return(T_Star); }
\+ { ECHO; return(T_Plus); }
\, { return(T_Comma); }
\- { ECHO; return(T_Minus); }
":=" { ECHO; return(T_VarAsgn); }
\: { return(T_Colon); }
\; { return(T_Semicolon); }
"<=" { ECHO; return(T_LESym); }
">=" { ECHO; return(T_GESym); }
\< { ECHO; return(T_LTSym); }
\> { ECHO; return(T_GTSym); }
= { ECHO; return(T_EQSym); }
\/= { ECHO; return(T_NESym); }
"=>" { return(T_Arrow); }
"<>" { ECHO; return(T_Box); }
\| { ECHO; return(T_Bar); }
! { ECHO; return(T_Bar); }
\. { ECHO; return(T_Dot); }
\/ { ECHO; return(T_Slash); }

{LETTER}(_?{LETTORDIGIT})*  {
int itoken;
itoken=find_mc(yytext);
if (itoken== -1)
{ TokenValue *tkvalue = (TokenValue *) malloc( sizeof(TokenValue) );
  tkvalue->Length=strlen(yytext);
  tkvalue->Value = (char *) malloc(tkvalue->Length + 1);
  strcpy(tkvalue->Value, yytext);
  tkvalue->TokenType = T_Identifier;
#ifdef YYDEBUG
```

```
  tkvalue->Position = yycolumno;
  tkvalue->Line = yylineno;
#endif /* YYDEBUG */
  yylval.Ptr = (VoidStar) tkvalue;
  return ( tkvalue->TokenType );
}
else
{ return ( itoken ); }
}

({DLIT})|({BASE}#{BINT}(\.{BINT})?#({EXP})?)|({BASE}:{BINT}(\.{BINT})?:({EXP})?) {
TokenValue *tkvalue = (TokenValue *)
malloc( sizeof(TokenValue) );
tkvalue->Length=strlen(yytext);
tkvalue->Value = (char *) malloc(tkvalue->Length + 1);
strcpy(tkvalue->Value, yytext);
tkvalue->TokenType = T_AbstractLit;
#ifdef YYDEBUG
tkvalue->Position = yycolumno;
tkvalue->Line = yylineno;
#endif /* YYDEBUG */
yylval.Ptr = (VoidStar) tkvalue;
return ( tkvalue->TokenType );
}

'({GRCHAR}|\"|\%)' { ECHO; return ( T_CharacterLit ); }

(\"({GRCHAR}|(\"\")|\%)*\")|(\%({GRCHAR}|(\%\%)|\")*\%) {
ECHO; return ( T_StringLit ); }

{BSPEC}(\"{EDIGIT}(_?{EDIGIT})*\"|\%{EDIGIT}(_?{EDIGIT})*\%) {
ECHO; return ( T_BitStringLit ); }

\n { /* end of line */
MVL_LINNUM++;
/* tobuf( "\n%4d\t", MVL_LINNUM);*/
yycolumno=0;
/*return(T_NEWLINE);*/
}

\-\-.*$  { /* comment */ }
```

```
. { ECHO; /*return (T_UNKNOWN);*/ }

%%
```

3 YACC Input File for VHDL-A

```
/****** Syntax for a subset of VHDL-A in YACC format **********/

%{
#include <stdio.h>
#include "hdl.h"

extern char yytext[];
extern int yylineno;

%}

%union{
  int Ivalue;
  VoidStar Ptr;
}

%start Start

%token
T_ACCESS
T_AFTER
T_ALIAS
T_ALL
T_AND
T_ARCHITECTURE
T_ARRAY
T_ASSERT
T_ATTRIBUTE
T_BEGIN
T_BLOCK
T_BODY
T_BUFFER
T_BUS
T_CASE
T_COMPONENT
T_CONFIGURATION
T_CONSTANT
T_DISCONNECT
T_DOWNTO
T_ELSE
```

T_ELSIF
T_END
T_ENTITY
T_EXIT
T_FILE
T_FOR
T_FUNCTION
T_GENERATE
T_GENERIC
T_GUARDED
T_IF
T_IN
T_INOUT
T_IS
T_LABEL
T_LIBRARY

T_LINKAGE
T_LOOP
T_MAP
T_NAND
T_NEW
T_NEXT
T_NODE
T_NOR
T_NULL
T_OF
T_ON
T_OPEN
T_OR
T_OTHERS
T_OUT
T_PACKAGE
T_PORT
T_PROCEDURE
T_PROCESS
T_RANGE
T_RECORD
T_REGISTER
T_REPORT
T_RETURN
T_SELECT

```
T_SEVERITY
T_SIGNAL
T_SUBTYPE
T_THEN
T_TO
T_TRANSPORT
T_TYPE
T_UNITS
T_UNTIL
T_USE
T_VARIABLE
T_WAIT
T_WHEN
T_WHILE
T_WITH
T_XOR

/* VHDL binary operators */

%nonassoc T_EQSym T_NESym T_LTSym T_LESym T_GTSym T_GESym
%left T_Plus T_Minus T_Ampersand
%left MED_PRECEDENCE
%left T_Star T_Slash T_MOD T_REM
%nonassoc T_DoubleStar T_ABS T_NOT MAX_PRECEDENCE

/* misc syms */

%token T_Apostrophe
T_LeftParen
T_RightParen
T_Comma
T_VarAsgn
T_Colon
T_Semicolon

T_Arrow
T_Box
T_Bar
T_Dot

%token  < Ptr > T_Identifier
T_AbstractLit
```

```
T_CharacterLit

T_StringLit
T_BitStringLit


%type  < Ptr >
ArchBody
ArchDecl
ArchDeclList
ArchHeader
ArchStmt
ArchStmtList
AssocActual
AssocElmt
AssocFormal
AssocList
BlockDeclItem
CompAsp
CompInstStmt
ConcurStmt
EntityAsp
GenericMap
InstUnit
NodeDecl
NodeList
PortMap
SignalList

%%
Start:
| DesignList
;

DesignList:
ArchBody | DesignList ArchBody
;

/* ----------- ARCHITECTURE BODY ----------------- */
ArchBody:
{ $$ = HDLArchBodyBefore(); }
ArchHeader ArchDecl T_BEGIN ArchStmt ArchEnd T_Semicolon
```

```
{ $$ = HDLArchBodyAfter($2, $3, $5); }
;

ArchHeader:
T_ARCHITECTURE T_Identifier T_OF T_Identifier T_IS
{ $$ = HDLArchHeader($2, $4); }
;

ArchDecl:
{ $$ = (VoidStar) NULL; }
| ArchDeclList { $$ = HDLArchDecl($1); }
;

ArchDeclList:
BlockDeclItem { $$ = HDLCatNodeLists( (VoidStar) NULL, $1); }
| ArchDeclList BlockDeclItem { $$ = HDLCatNodeLists($1, $2); }
;

ArchStmt:
{ $$ = (VoidStar) NULL; }
| ArchStmtList { $$ = HDLArchStmt($1); }
;

ArchStmtList:
ConcurStmt { $$ = HDLNewListElmt((VoidStar) NULL, $1); }
| ArchStmtList ConcurStmt { $$ = HDLNewListElmt($1, $2); }
;

ArchEnd:
T_END | T_END T_ARCHITECTURE
;

BlockDeclItem:
NodeDecl
| SignalDecl { $$ = (VoidStar) NULL; }
| ConfigSpec { $$ = (VoidStar) NULL; }
;

NodeDecl:
T_NODE NodeList T_Semicolon { $$ = $2; }
;
```

```
NodeList:
T_Identifier { $$ = HDLNewListElmt((VoidStar) NULL, $1); }
| NodeList T_Comma T_Identifier { $$ = HDLNewListElmt($1, $3); }
;

SignalDecl:
T_SIGNAL SignalList T_Semicolon
;

SignalList:
T_Identifier | SignalList T_Comma T_Identifier
;

/* -------- CONFIGURATION SPECIFICATION --------- */

ConfigSpec:
T_FOR CompSpec BindIndic T_Semicolon
;

CompSpec:
InstList T_Colon T_Identifier
;

InstList:
LabelList | T_OTHERS | T_ALL
;

LabelList:
T_Identifier | LabelList T_Comma T_Identifier
;

BindIndic:
| T_USE EntityAsp GenericMap PortMap
;

EntityAsp:
T_ENTITY T_Identifier ArchIdent { $$ = $2; }
;

ArchIdent:
| T_LeftParen T_Identifier T_RightParen
;
```

```
/* ----------- CONCURRENT STATEMENT ------------- */

ConcurStmt:
CompInstStmt
;

/* ----- COMPONENT INSTANTIATION STATEMENT ------ */

CompInstStmt:
T_Identifier T_Colon InstUnit GenericMap PortMap T_Semicolon
{ $$ = HDLComponentInst($1, $3, $4, $5); }
;

InstUnit:
CompAsp | EntityAsp
;

CompAsp:
T_Identifier | T_COMPONENT T_Identifier {$$ = $2;}
;

GenericMap:
{ $$ = (VoidStar) NULL; }
| T_GENERIC T_MAP T_LeftParen AssocList T_RightParen { $$ = $4; }
;

PortMap:
{ $$ = (VoidStar) NULL; }
| T_PORT T_MAP T_LeftParen AssocList T_RightParen {$$ = $4; }
;

AssocList:
AssocElmt { $$ = HDLNewListElmt((VoidStar) NULL, $1); }
| AssocList T_Comma AssocElmt { $$ = HDLNewListElmt($1, $3); }
;

AssocElmt:
AssocActual | AssocFormal T_Arrow AssocActual {$$ = $3;}
;

AssocFormal:
```

```
T_Identifier
;

AssocActual:
T_Identifier | T_AbstractLit
;

%%

extern FILE *yyout;

void yyerror(str)
char* str;
{
  fprintf( yyout, "ERROR: %s at line %d while reading `%s'\n", str,
yylineno, yytext );
}
```

Bibliography

[1] F. N. Najm, "Transition Density: A New Measure of Activity in Digital Circuits," *IEEE Transactions on CAD*, vol. 12, pp. 310–323, February 1993.

[2] K. Roy and S. C. Prasad, "Circuit Activity Based Logic Synthesis for Low Power Reliable Operations," *IEEE Transactions on VLSI Systems*, vol. 1, pp. 503–513, December 1993.

[3] M. E. Lesk, "LEX – A Lexical Analyzer Generator," Comp. Sci. Tech. Rep. 39, AT&T Bell Laboratories, Murray Hill, NJ, October 1975.

[4] S. C. Johnson, "YACC – Yet Another Compiler Compiler," Comp. Sci. Tech. Rep. 32, AT&T Bell Laboratories, Murray Hill, NJ, 1975.

[5] P. E. Allen, B. J. Blalock, and G. A. Rincon, "A 1V CMOS Opamp Using Bulk-Driven MOSFETs," in *Proceedings of the 1995 IEEE International Solid-State Circuits Conference*, pp. 192–193, IEEE, February 1995.

[6] P. R. Gray. Personal communication, 1995.

[7] M. L. Liou and Y. L. Kuo, "Exact Analysis of Switched Capacitor Circuits with Arbitrary Inputs," *IEEE Transactions on Circuits and Systems*, vol. CAS-26, pp. 213–223, April 1979.

[8] Y. P. Tsividis, "Analysis of Switched Capacitive Networks," *IEEE Transactions on Circuits and Systems*, vol. CAS-26, pp. 935–946, November 1979.

[9] J. Vlach, K. Singhal, and M. Vlach, "Computer Oriented Formulation of Equations and Analysis of Switched-Capacitor Networks," *IEEE Transactions on Circuits and Systems*, vol. CAS-31, pp. 753–765, September 1984.

[10] J. L. Melsa and A. P. Sage, *An Introduction to Probability and Stochastic Processes.* Englewood Cliffs, NJ: Prentice-Hall, Inc., 1973.

[11] G. H. Golub and C. F. Van Loan, *Matrix Computations.* Baltimore: Johns Hopkins University Press, 1983.

[12] R. H. Bartels and G. W. Stewart, "Solution of the Matrix Equation $AX + XB = C$," *Communications of the ACM*, vol. 15, pp. 820–826, 1972.

[13] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms.* New York St. Louis San Francisco: McGraw-Hill Book Company, 1990.

[14] A. S. Hodel and K. Poolla, "Parallel Solution of Large Lyapunov Equations," *SIAM Journal on Matrix Analysis and Applications*, vol. 13, no. 4, pp. 1189–1203, 1992.

[15] H. De Man, J. Rabaey, G. Arnout, and J. Vandewalle, "Practical Implementation of a General Computer Aided Design Technique for Switched Capacitor Circuits," *Journal of Solid-State Circuits*, vol. SC-15, pp. 190–200, April 1980.

[16] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *Data Structures and Algorithms.* Reading, MA: Addison-Wesley Publishing Co., 1987.

[17] G. Casinovi, "A Macromodeling Algorithm for Analog Circuits," *IEEE Transactions on Computer-Aided Design*, vol. CAD-10, pp. 150–160, February 1991.

[18] E.-C. Ma, "A Finite Series Solution of the Matrix Equation $\mathbf{AX - XB = C}$," *SIAM Journal of Applied Mathematics*, vol. 14, May 1966.

[19] R. Smith, "Matrix Equation $\mathbf{AX + XB = C}$," *SIAM Journal of Applied Mathematics*, vol. 14, no. 1, 1968.

[20] H. Risken, *The Fokker-Planck Equation.* Berlin Heidelberg: Springer-Verlag, 1984.