POWER ELECTRONIC CIRCUIT RELIABILITY ANALYSIS INCORPORATING PARALLEL SIMULATIONS

Linda A. Kamas *Seth R. Sanders*
Department of Electrical Engineering and Computer Sciences
University of California at Berkeley
email: kamas@eecs.berkeley.edu and sanders@eecs.berkeley.edu

ABSTRACT

This paper presents and applies reliability analysis methods with power electronic circuit simulation. The focus is on the First-Order Reliability Method (FORM). Two circuits are analyzed: an open-loop updown converter and a similar closed-loop circuit that is currently in mass production. The open-loop circuit provides an example from which rich quantitative data can be obtained for various methods. The closed-loop production design circuit provides a proof-of-concept for a FORM application on a complex design. The implementation includes parallel gradient computations across six networked workstations. The parallel environment is described in detail.

1 INTRODUCTION

An important problem within the power electronics design community is the determination of the effects of parameter variation on circuit performance. Various aspects of this problem have been studied in [1, 2, 3, 4, 5].

The integrated circuit (IC) computer-aided design literature offers a wealth of ideas addressing the need to incorporate parameter variations into the design of ICs [6], [7]. In IC manufacturing, however, several parameter variations can be correlated due to their being created in the same process, significantly reducing the number of actual parameters to be considered. Thus, the fundamental difference that must be considered in discrete circuitry is that the approach must be generalizable to circuits with higher dimensional parameter spaces. We introduce the First-Order Reliability Method (FORM) [8], developed in the civil engineering reliability community as a useful method for circuit reliability analysis.

An interesting element of this work is that we analyze a commercial flyback converter with PWM feedback. This introduces some difficulties not apparent in typical idealized examples, including a longer simulation time to incorporate specific device model simulation. As a result, compensation for the longer simulation time neccesitates doing some simulations in parallel. Because FORM is a gradient-based method, and since finite-difference gradient calculations are independent in each dimension, the costly gradient calculations are computed across six computers in parallel. This is described in the "Parallel Simulation Environment" section.

2 THEORETICAL BACKGROUND

If there are n variable parameters of a circuit, then an n-vector of parameter values (a point in the parameter space) can be used as a representation for that circuit. Typical parameters are component values, including, but not limited to, resistances, capacitances, inductances, and transistor parameters. Statistical variations in each component value result in parameter variation, possibly centered around the ideal value (design center) for each component. Typically a lengthy set of simulations is required to determine the functionality of a circuit with respect to a particular parameter set.

2.1 Terminology

Performance Function. The *performance functions* are the circuit outputs by which functionality is evaluated. These may include, for example, average steady-state voltages and currents and/or peak-to-peak ripple values.

Region of Acceptability. The *region of acceptability*, R_A, is the set of vectors in the space of circuit parameters that results in a circuit whose *performance functions* are deemed acceptable.

0-7803-3977-0/96 $10.00

2.2 Reliability Methods: Monte Carlo, Importance Sampling, FORM, SORM

Here we describe four sampling methods to analyze reliability. Our focus is on FORM, the First-Order Reliability Method. Monte Carlo, importance sampling, and second-order reliability method (SORM) estimates are compared to FORM estimates.

Monte Carlo Monte Carlo analysis, a statistical sampling method, is generally believed to be the most accurate for yield estimation. Monte Carlo methods employ a random selection of parameter values in such a way that they reflect the physical random processes of the original problem. From computations based on the random parameter values, one can estimate the probability of failure, P_f. Results from a Monte Carlo analysis are often used to compare the accuracy of other methods, when possible. However, Monte Carlo cannot be used in cases where the required number of simulations is prohibitive. The number of points needed for a calculation of P_f will be roughly proportional to the reciprocal of the failed fraction. Hence, the required number of points is very large when yield is very close to unity.

Experiments in [9] demonstrated advantages of FORM over Monte-Carlo analysis when failure rates are low. Specifically, consider a design that aspires to six sigma [5, 10] standards (about 4 failures per million). If we want to have 95% confidence that the percent error on a yield estimation is 100% or less for an estimated probability of failure, $P_f = 4 \times 10^{-6}$, then approximately 10^6 simulations will be necessary. If each simulation only took one second, and 10^6 simulations were required, that would take about 11-12 days of continuous simulation using one computer.

Importance Sampling Importance sampling is a refined version of the Monte Carlo method, seeking to reduce the variance in the resulting approximation for a given number of samples. Equivalently, one could use it to reduce the number of samples needed for a given variance. This introduces a bias, but one for which we can compensate via a modified estimator. The importance sampling analysis can use the points of most-likely failure, obtained by FORM, to determine where the "important" regions in the sampling space are. We pursue that approach.

An overview of Importance Sampling and other Monte Carlo methods, including a brief historical background of its use can be found in [11].

FORM Given a design center and parameter distributions, a FORM analysis [8] methodically conducts simulations in a gradient-based search for the statistically most-likely point of failure for a particular performance specification. A FORM analysis then determines the tangent hyperplane along the boundary of the region of acceptability, R_A, at the most-likely point of failure. The yield is computed based on that first-order model. For this, the parameter space is divided into pass/fail half-spaces. The calculations are done by first transforming the distribution space to the standard normal space, which facilitates the yield computation. When several performance criteria are specified, the R_A approximation is a logical intersection of the half-spaces. The FORM algorithm was developed to analyze the structural reliability of structures such as buildings and bridges, but is applicable to the reliability of a circuit design under parameter variation. Whereas structural reliability emphasizes the determination of P_f of one structure, the mathematics behind the analysis applies to the determination of P_f of an ensemble of structures (circuits in our case) that is equivalent to a yield calculation.

See earlier work [9] and also [8] for more details.

SORM Because our emphasis will be on FORM, we only briefly describe the Second-Order Reliability Method (SORM) [12]. A SORM analysis approximates the limit-state surface near the point of most-likely failure (PMLF) with a model that incorporates curvature.

3 EXPERIMENTAL RESULTS

In this section, we analyze an open-loop updown converter and a production designed closed-loop updown converter. The open-loop circuit is simpler and thus simulates faster. More detailed data can be extracted from its analysis. The production design circuit provides a proof-of-concept of a FORM application on an industrial design.

3.1 Open-loop Updown Converter

Here we analyze the reliability of the open-loop updown converter of Figure 1 with six varying parameters and four performance criteria which will be converted into limit-state functions. The performance criteria are that the output voltage is between an upper and a lower limit and the inductor current is between an upper and

g-func $x =$ $[N, C, V_1, R, d_{area}, L]^T$	FORM P_f	SORM P_f	MC 2000 pts $E(P_f)$ $(\sigma =)$	Isamp. 80 pts. $E(P_f)$ $(\sigma =)$
$g_1(x) = V_o(x) - (-2.2)$	.025	.023	.017 (.0029)	.029 (.013)
$g_2(x) = -1.9 - V_o(x)$	.016	.017	.013 (.0025)	.016 (.006)
$g_3(x) = I_L(x) - .70$	.0012	.0014	.0005 (.0005)	.0013 (.0011)
$g_4(x) = .90 - I_L(x)$	.034	.023	.036 (.0042)	.038 (.013)
total P_f	.058		.056	.054
std. dev.			(.0051)	(.014)

Table 1: Open loop updown converter P_f approximations for each limit-state function using FORM, Monte Carlo, and importance sampling.

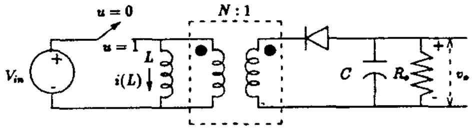

Figure 1: Model of an updown converter with isolation transformer.

a lower limit. The six-dimensional parameter vector is $x = [N, C, V_{in}, R_o, d_{area}, L]^T$, where N is the turns ratio, C is the output capacitance, V_{in} is the input voltage, R_o is the output resistance, d_{area} is a multiplier of the diode junction area, and L is the transformer magnetizing inductance. These parameters are shown in Figure 1. The four limit-state functions, based on the performance criteria, are:

$$g_1(x) = V_o(x) - (-2.2) \quad (1)$$

$$g_2(x) = -1.9 - V_o(x) \quad (2)$$

$$g_3(x) = I_L(x) - .70 \quad (3)$$

$$g_4(x) = .90 - I_L(x). \quad (4)$$

We considered uncorrelated, normally distributed values with a standard deviation of 2% of the nominal value. All simulations were executed with the SABER simulator [13], which ran a transient simulation and computed the average values of $v_o(x,t)$ and $i_L(x,t)$ over one period after steady-state was approximately reached. The simulation end time was fixed for all simulations and preselected to be long enough to assume that steady-state would be approximately reached.

Results of FORM, SORM, 2000-point Monte Carlo, and 80-point importance sampling analyses generally agreed to within 50% of each other in their probabilitys of failure (P_fs) for the four limit-state functions and are listed in Table 1. Importance sampling centers were determined by the FORM calculation.

The parallel implementation used for gradient calculations resulted in a significant savings in the effective number of simulations in the FORM analysis. If there had been no parallelization, the analysis would have taken 79 + 79 × 6 = 553 serial simulations: 79 point simulations and 79 gradient simulations, requiring 6 simulations each. Since the gradient calculations were parallelized, the effective number of serial simulations was 158 instead of 553.

3.2 Closed-loop Commercial Flyback Converter

g-function, $x =$ $[N_4, N_2, R_{N4}, R_{10}, R_{11}, V_{ref}]^T$	FORM P_f	Isamp. P_f 10 pts.
$g(x) = 17.5 - V_o(x)$	.07	.10
std. dev.		.033

Table 2: FORM and importance sampling results for the commercially-based circuit.

Experiments with a production-based flyback converter circuit presented the greatest technical challenges due to the lengthy simulation times resulting from its complexity. Repeated simulations of this circuit provided the impetus for the parallel gradient calculation. A block diagram of the flyback converter with PWM feedback is shown in Figure 2. The netlist for this circuit is approximately 120 lines long.

The reliability analysis evaluated one limit-state func-

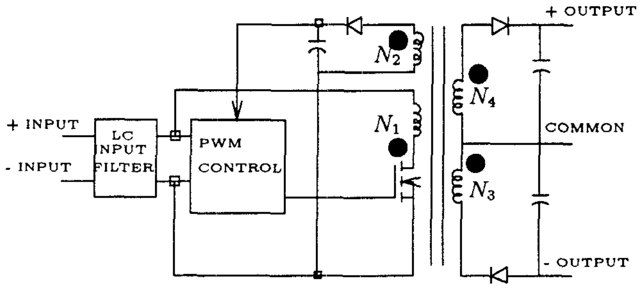

Figure 2: Flyback Converter with PWM Feedback

tion: $g(x) = 17.5 - V_{output}$. The output voltage when parameters are at their central values is 17.2 Volts. Thus the failure criterion is a deviation of about 2% of the average steady-state value. Six parameters were varied: $x = [N_2, N_4, R_{N4}, R_{10}, R_{11}, V_{ref}]^T$, where N_2 is the number of turns on the sense winding, N_4 is the number of turns on the positive output winding, R_{N4} is the DC resistance of the N_4 winding, R_{10} and R_{11} are resistors which are part of a divider in the feedback loop; and V_{ref} is the reference voltage provided by the PWM chip. The actual tolerances for the constituents of the parameter vector, x, were taken to be 1%, 1%, 10%, 1%, 5%, and 2%, respectively, by inspection of the design. In our analysis, parameter distributions were assumed to be uncorrelated and normal with the ideal values as central values and standard deviations of 20% of the stated tolerance for the 5% and 10% parts. This distribution is based on a study in [4] where resistance value distributions were investigated by measuring a sample of 400 resistors. The distribution was approximately normal with a standard deviation of approximately 20% of the 10% tolerance band. In this study, the standard deviations of the 1% and 2% parts were assumed to be 50% of the tolerance. For example, the 1% parameters that vary by ±.1 were assigned a standard deviation $\sigma = .05$ in our analysis. We assumed that these more tightly toleranced parameters had distributions that were closer to uniform.

All simulations were executed with the SABER simulator [13], which ran a transient simulation and computed the average of $v_o(x,t)$ over several switching periods. A single transient simulation took about four hours to reach a pseudo-steady-state on a DEC 5000 computer. We alleviated the problem of long transient simulation times by extracting an initial point from a

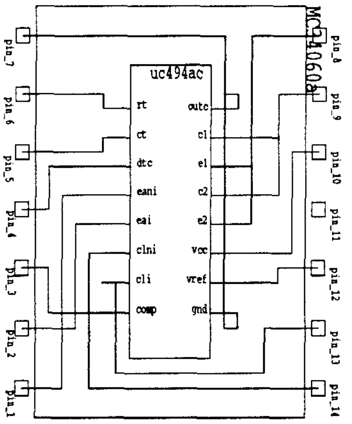

Figure 3: MC34060A PWM modelled using SABER's UC494AC PWM model

steady-state run and using that as an initial point for subsequent simulations with perturbed parameter values. This eliminates some of the initial transient behavior and reduces simulation time to about one hour. We were not interested in transient behavior in our performance criterion. The transformer model included the magnetizing inductance, leakage inductance, DC winding resistances, and core material and shape. The core modelling was facilitated by finding an existing model within the SABER library which closely matched the actual transformer characteristics. The PWM chip, an MC34060a [14], was modelled by a modification of the existing UC494ac [15] chip model available within the SABER library. The resulting pin to pin diagram is shown in Figure 3. The output control pin (labeled "outc") of the UC494ac should be tied to ground.

The results of a FORM and importance sampling analysis are listed in Table 2. These show a P_f due to component parameter variation of about .07 to which makes sense in view of the parameter tolerances. When we analyzed the test results of 77 such manufactured units, there were no failures due to tolerance deviation; failures were due to errors in assembly. In future work, factors in the assembly process and their distributions need to be accounted for. Other than the effects of the assembly process, the differences between the design we analyzed in simulation and the actual design were that the simulated design did not include the additional linear regulation circuitry. The order of importance of the parameters on V_{out} is: $V_{ref}, N_4, N_2, r_{10}, r_{11}$, and r_{N4}, with V_{ref} being the most influential parameter.

This analysis shows a proof-of-concept for a FORM analysis applied to a relatively complex power electronic circuit design.

Once again, the parallel implementation resulted in a significant savings in the effective number of simulations. If there had been no parallelization, the analysis would have taken $8 + 5 \times 6 = 38$ serial simulations: 8 point simulations and 5 gradient simulations, requiring 6 simulations each. Since the gradient calculations were parallelized, the effective number of simulations was 13 instead of 38. The computation time was about 18 hours on DEC5000 and DEC3100 machines. The point calculation, on the DEC5000 took about an hour. The gradient calculation, executed on both types of computers, took about 2 hours due to competition in the time to write to disks and the fact that the calculation only ends when the slowest machine is done with its part of the calculation. Data for the 5ms of transient required about 50Mbytes of disk storage space for each dimension of the calculation.

4 PARALLEL SIMULATION ENVIRONMENT

The FORM algorithm is implemented here by CALREL [16] software, which was developed to analyze structural reliability, but which is applicable to the reliability of a circuit design. Recall that the FORM algorithm is a gradient-based method and thus requires a gradient calculation. The SABER simulation environment is a commercial simulation package which does not allow access to the internal simulation engine code. Thus, gradient calculations were computed using finite-differences. Finite difference gradients are usually the most computationally expensive portion of gradient-based methods. This bottleneck is alleviated by conducting these calculations in parallel across six networked computers, using PERL [17] scripts as the "glue" to interface between CALREL and parallel calls to SABER [13]. This procedure is shown graphically in Figure 4.

PERL is a versatile scripted language that is useful for pattern matching, text manipulation, process manipulation, and networking. It is available for free (see Appendix A) and runs on several operating systems. The given implementation is in a Unix networked environment. In particular, PERL programs facilitated the multiple "child processes" that needed to be "spawned" in parallel. In each child process, a remote shell command was executed that called SABER via batch files on a different computer.

Important shell commands for remote processing are listed below. Definitions and other relevant information come from one or more of the sources [18, 19, 17, 20]. We then outline the procedure of performing parallel gradient calculations.

Shell Commands for Remote Processing:

rcp Remote copy. Copies files to or from a local filesystem from or to a remote host filesystem.

rsh Remote shell. Executes a command on a remote host.

batch commands A sequence of simulator commands taken from a file instead of from the terminal.

4.1 Procedure for Parallel Calculations

When CALREL needs a gradient, it communicates at what point the gradient is needed by writing the point to a file. A Perl script is then called by CALREL to write n_w batch files, where n_w is the number of workstations, that distribute the calculation. We wrote the Perl programs to create batch files in such a way that the user could distribute the workload however she or he saw fit. For example, one could execute two parts of the gradient calculation on a very fast machine and one part on each of the other machines. The batch files are copied to the filesystems of n_w remote workstations via *rcp*. Then a *remote shell* (via *rsh*) is started on each remote machine to invoke the simulator to run the batch file. This portion is conducted in parallel by spawning n_w child processes to compute the performance values by perturbation in the n_p dimensions of the parameter vector. The results are copied back to the local file system (again by *rcp*), where the multiple files are edited to a single file format of gradient components that is read by CALREL. All steps are automated: the user does not need to be involved except to start the procedure by invoking CALREL, and then to interpret results when CALREL is finished.

In order to copy to a remote filesystem or invoke a remote shell, one must have access to computer accounts

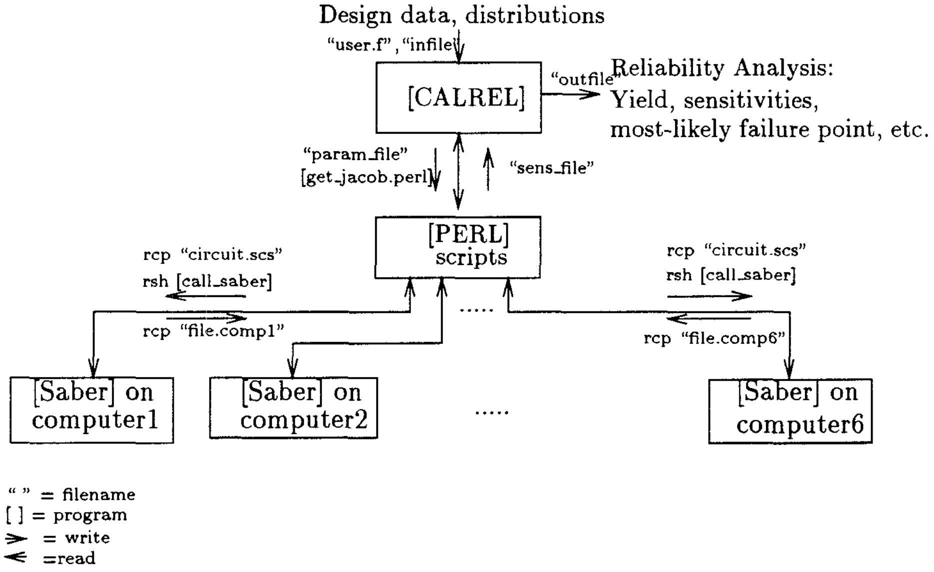

Figure 4: Software communication diagram of gradient computed in parallel on six networked computers

on the remote machines and set them up as *remote hosts* to the local machine. This can introduce a security risk[19].

4.2 Suggestions

Below are suggestions for use when implementing a parallel simulation environment:

- Use good programming habits. Documentation, modularity, etc. are particularly important in a parallel environment.
- Consider simplicity at the expense of some efficiency, at least initially.
- Take measurements to determine the effective speed of the computer at performing the transient simulation. This is what really counts, not the quoted processor speed under ideal conditions. Use this information to load balance the simulations (e.g. slowest starts first).
- Determine how much storage space is needed. Long transient simulations of switching converters can require a significant amount of disk space.

5 SUMMARY AND CONCLUSIONS

A FORM analysis computes a point of most-likely failure by including parameter distributions into the directed simulation scheme and provides a good estimate of the probability of failure, P_f, with a large savings in simulations compared to Monte-Carlo analysis when failure probabilities are small.

A FORM analysis in conjunction with importance sampling were used to compute an estimate of P_f. Importance sampling can reduce the variance of a Monte Carlo estimate or reduce the number of required samples needed for a given variance.

A FORM analysis in conjunction with parallel calculations were demonstrated on an open-loop updown converter and on a closed-loop production design updown converter. Simulations were conducted across six networked workstations to implement the parallelizable portion of the reliability analysis–the gradient calculations. In both cases six components were simultaneously varied. In the open-loop circuit, results were similar for the FORM, SORM, Monte Carlo and importance sampling analyses. FORM and importance sampling analyses for the production design showed similar results. The production design showed a proof-of-concept for a FORM analysis applied to a relatively complex power electronic circuit design.

Parallelizing repeated circuit simulations can result in practical statistical reliability analysis when serial computing times are impractical. Details were given for this particular implementation.

Appendix A. Perl can be obtained by anonymous

FTP at the following sites:

- ftp.uu.net (137.39.1.9)
- archive.cis.ohio-state.edu (128.146.8.52)
- jpl-devvax.jpl.nasa.gov (128.149.1.143)

REFERENCES

[1] A. Cirillo, N. Femia, and G. Spagnuolo, "An interval mathematics approach to tolerance analysis of switching converters," *Proceeding of IEEE Power Electronics Specialists Conference*, pp. 1349–1355, 1996.

[2] V. Thottuvelil, "Accelerated tolerance analysis of power-supply protection circuits using simulation," *IEEE Applied Power Electronics Conference Proceedings*, pp. 169–175, 1995.

[3] S. Kelkar, R. Wunderlich, and L. Hitcock, "A general computer aided tool to assess the impact of component value tolerance on power supply performance," *IEEE Applied Power Electronics Conference*, 1989.

[4] S. Hageman, "Powerful circuit-analysis programs eliminate need for preproduction runs," *Personal Engineering and Instrumentation News*, pp. 49–56, Oct. 1990.

[5] R. White, "An introduction to six sigma with a design example," *IEEE APEC*, pp. 28–35, 1992.

[6] R. Spence and R. Singh Soin, *Tolerance Design of Electronic Circuits.* Addison-Wesley, 1988.

[7] S. Director, P. Feldmann, and K. Krishna, "Stastical integrated circuit design," *IEEE Journal of Solid-State Circuits*, pp. 193–202, March, 1993.

[8] H. Madsen, S. Krenk, and N. Lind, *Methods of Structural Safety.* Prentice-Hall, Inc., 1986.

[9] L. Kamas and S. Sanders, "Reliability analysis via numerical simulation of power electronic circuits," *IEEE Workshop on Power Electronics and Computers*, 1994.

[10] R. White, "Component tolerance and circuit performance: A case study," *IEEE APEC*, pp. 922–927, 1993.

[11] J. Hammersley and D. Handscomb, *Monte Carlo Methods.* Chapman and Hall, N.Y., 1983.

[12] A. DerKiureghian, "Structural reliability methods for seismic safety assessment: a review," *Engineering Structures*, pp. 412–424, Vol.18,No.6,1996.

[13] Analogy, Inc., Beaverton, Oregon, *SABER user's Manual, Release 3.2.*

[14] Motorola Inc., *Linear and Interface Integrated Circuits, 1988.*

[15] Unitrode Integrated Circuits Corporation, *Linear Integrated Circuits Data and Applications Handbook, 1990.*

[16] P. Liu, H. Lin, and A. Der Kiureghian, *CALREL User Manual, Report No. UCB/SEMM-89/18.* Dept. of Structural Engineering, Mechanics and Materials, UC Berkeley, August, 1989.

[17] L. Wall and R. Schwartz, *Programming Perl.* O'Reilly and Associates, Inc., March, 1992.

[18] P. Abrahams and B. Larson, *UNIX for the Impatient.* Addison-Wesley Publishing Company, Inc., 1992.

[19] W. Stevens, *UNIX Network Programming.* Prentice Hall Software Series, Prentice-Hall, Inc., 1990.

[20] B. Kernighan and R. Pike, *The UNIX Programming Environment.* Prentice-Hall, Inc., 1984.