An Initialization Technique for the Waveform-Relaxation Circuit Simulation

S. E.-D. HABIB* and G. J. AL-KARIM

Electronics Department, Faculty of Engineering, Cairo University, Giza, Egypt

(Received 26 June 1992; In final form 18 September 1992)

This paper reports the development of the Cairo University Waveform Relaxation (CUWORX) simulator. In order to accelerate the convergence of the waveform relaxation (WR) in the presence of logic feedback, CUWORK is initialized *via* a logic simulator. This logic initialization scheme is shown to be highly effective for digital synchronous circuits. Additionally, this logic initialization scheme preserves fully the multi-rate properties of the WR algorithm.

Keywords: Circuit simulation, waveform relaxation, logic initialization

I. INTRODUCTION

Classical circuit simulators (*e.g.*, SPICE program [1]) are of limited value for large scale circuits with circuit size, $n > 1000$. Relaxation techniques for the solution of n coupled equations display an execution time that grows linearly with n . The application of these techniques to circuit simulation at the nonlinear algebraic equation level produced several successful circuit simulators, *e.g.*, MOTIS [2] and SPLICE [3].

In 1982, Lelarsmee *et al.* [4] exploited, for the first time, the relaxation techniques at the differential equation level. The waveform relaxation (WR) algorithm proceeds as follows. The circuit is partitioned into loosely coupled subcircuits, which

are arranged in ascending order along the signal flow path. Next, each subcircuit is analysed in turn for the full analysis period required. A classical circuit simulator can be employed at this subcircuit level. In a sense, the signal is propagated through the sequence of subcircuits that make up the circuit. A single sweep through the whole circuit is termed a WR cycle. Several WR cycles are generally needed to account for the coupling between subcircuits.

Several excellent reviews are available for this new circuit simulation technique [5]. In addition, several packages were built around the WR algorithm, *e.g.*, RELAX 2.3 [6], SWAN [7], TOGGLE [8], and MOZART [9]. The main advantages of the WR technique include linear

* Corresponding author.

growth of the execution time with the circuit size; guaranteed convergence under very mild conditions [4, 5]; natural exploitation of the multi-rate phenomena in LSI circuits; and flexibility, since a classical circuit simulator is embedded at the subcircuit level. The main problems of the WR algorithm are slow convergence in the presence of wide feedback loops spanning several subcircuits, and Large memory requirements.

The first problem is particularly demanding. On one hand, one can not define a wide, strongly-coupled feedback loop as a single subcircuit due to the large accompanied increase in the subcircuit solution time. On the other hand, a large number of WR cycles would be needed to achieve convergence between the subcircuits encompassed by the feedback loop. Several techniques have been proposed to resolve this convergence problem, which included windowing, multi-level WR schemes and dynamic partitioning schemes. Windowing involved the break up of the analysis period, T , into several windows, so as to limit the error propagation within one WR cycle [5]. Note, however, that a short window size affects adversely the multi-rate advantages of the WR method. A multi-level WR scheme was proposed by Jun *et al.* [10], where several local WR cycles are carried around wide feedback loops, in addition to the global WR cycles. Dumlugol *et al.* [7] employed a dynamic partitioning scheme, called segmented WR suitable only for digital synchronous circuits.

This paper describes the development of the Cairo University Waveform Relaxation (CU-WORX) simulator. The program structure and features are described in Section II. Section III presents some results on several circuits. The issue of the program accuracy is also addressed. The logic initialization scheme for solving the WR convergence problem of the wide feedback loop is presented in Section IV.

II. CUWORX PROGRAM DETAILS

CUWORX, written in C language, is partitioned into the five segments shown in Figure 1. The main

FIGURE 1 Main segments of CUWORX.

routine calls the data I/O utility and WR manager segments. The WR manager calls the individual subcircuits according to the sequence specified by the user on the data input files. A Gauss–Seidel block-wise WR relaxation scheme is adopted. The WR manager also calls the subcircuit solver routine to analyse the chosen subcircuit. Additionally, the WR manager observes the convergence of the WR process. The “subcircuit setup and solution” segment of Figure 1 is responsible for analysing a given subcircuit for the required simulation interval. This segment is essentially a classical circuit simulator. Time discretization

employs either the trapezoidal or backward Euler implicit formulas, according to the user's choice. The time step is dynamically varied *via* the local truncation error (LTE) scheme of Nagel [1]. On the linearized equations level, a full LU solver is adopted. The final part of CUWORX is the utilities segment (*cf.* Fig. 1). It contains a set of general purpose routines such as vector copy, coefficient insertion, *etc.*

Efficient interpolation is essential for the WR algorithm. A cubic-spline interpolator was ultimately chosen for CUWORX.

At present, CUWORX has built in models for linear resistors and capacitors, grounded voltage sources, *n*- and *p*-type enhancement or depletion MOSFETS. The Shichman–Hodges model is adopted for the MOSFETS model, (equivalent to Level 1 model in SPICE [1]).

III. RESULTS

CUWORX was tested against SPICE2G.6 for several circuits to ensure its accuracy and proper functioning, which included dynamic shift registers, full adders and ring oscillators of different sizes. Identical responses were consistently obtained for all the tested circuits. A full exposition of these tests is given in [11]. The efficiency of our code can be tested by running CUWORX in the direct mode, where the whole circuit is treated as a single subcircuit, and comparing CUWORX *versus* SPICE2G.6. CUWORX consistently achieved smaller execution times than SPICE2G.6 for all the tested circuits. (*cf.* Tab. I).

TABLE I

Example	CPU time (sec)**	
	SPICE2G.6	CUWORX (direct mode)
1 bit dynamic SR	11.07	3.33
8 bit dynamic SR	919.51	918.51*
1 bit full adder	56.06	17.93
7 inverter ring oscillator	49.25	18.44

* Sparsity is not exploited in CUWORX.

** On μ VAX II.

We have noticed that the waveforms calculated during successive WR cycles display an oscillatory, or underdamped, behaviour around the final solution. This behavior is effectively remedied by under-relaxing the outer WR loop as suggested by Carlin *et al.* [9]. This under relaxation scheme was implemented in CUWORX and can typically achieve 50% reduction of the number of WR cycles required for convergence.

IV. LOGIC INITIALIZATION

Iterative schemes generally converge faster when the starting guess is closer to the final solution. To the best of our knowledge, no reference attacked seriously the initialization problem of the WR algorithm. In fact, if we can initialize the WR with approximate waveforms that describe the general behaviour of the circuit, then the slow convergence problem of the WR scheme in the presence of wide feedback loops would also be solved. This latter problem results from the inability of subcircuits analyzed first in a given WR sweep to predict the feedback coming from subcircuits analyzed later in the same cycle.

We chose to initialize CUWORX with the output generated by running the same circuit on a logic simulator, specifically the one in the SPLICE1.7 mixed-mode simulator [12]. The idea is to verify the success of logic initialization especially with circuits containing logic feedback paths. Several synchronous and asynchronous circuits were used to test this idea. Although these circuits are small scale circuits, they are carefully chosen to emphasize the feedback problem. A four-bit dynamic ring counter was used (Fig. 2A) as an example of a synchronous sequential digital circuit. It was initialized with a binary value of 1000. Simulation was made with and without logic initialization for an observation period T equivalent to more than 16 clock cycles. Thus, the “1” is allowed to cycle four times. Figure 2D depicts the slow build up of the waveform without logic initialization as the signal propagates around the

FIGURE 2 The 4-bit dynamic ring counter initialized by a binary value 1000. (A) Circuit diagram schematics; (B) The 1st bit waveform according to the logic simulator; (C) The 1st bit waveform finally obtained by CUWORX; (D) The 1st bit waveform without logic initialization at different WR cycles.

feedback path. Table II summarizes the results for this example. Note that only four WR cycles are needed with logic initialization. Since the "1" is cycled 4 times during T , then 4 WR cycles are the minimum number of WR cycles that can be achieved. This clearly shows that logic initializa-

tion is highly effective in accelerating the convergence of synchronous sequential circuits, which are the main stay of digital sequential circuits design today.

The performance of logic initialization was also tested for asynchronous sequential circuits. No

TABLE II

Example	Class	WR without LI		WR with LI*		SPLICE CPU (sec)	Simula- tion T (μ sec)
		No. of cycles	CPU time (sec)	No. of cycles	CPU time (sec)		
4 bit counter	synchron.	7	1550.71	4	897.5	4.54	0.2
7 inv. ring osc.	asynchron.	13	149.73	12	142.3	1.61	0.015
7 inv. ring osc.	asynchron.	29	673.97	26	627.3	1.61	0.03
7 inv. ring osc.**	asynchron.	15	352.12	12	302.9	1.61	0.03

* LI $\equiv$ Logic initialization.

** Under-relaxation factor = 0.93. For all other cases, this factor = 1.

appreciable gain was achieved from logic initialization (*cf.* the 7 inverter ring oscillator example of Tab. II). This negative result is partially due to the nature of the logic simulator and partially due to the nature of asynchronous circuits. For synchronous circuits, the clock aligns events. Timing errors incurred during the analysis process are not allowed to propagate freely through the circuit despite the simplicity of the models used. In asynchronous circuits, however timing errors accumulate and the feedback paths compound these errors again. Errors are amplified in proportion to the observation period T . Additionally, logic simulators do not provide accurate delay values. A better approach may be to use a switch level simulator instead of a gate level logic simulator.

We have also tried logic initialization with several examples of combinational circuits. No appreciable improvement was remarked. The probable explanation of this result is the small number of WR cycles needed at any rate (3–4 cycles), which leaves little room for improvement via the logic initialization scheme.

V. CONCLUSIONS

Cairo University Waveform Relaxation simulator (CUWORX) has been developed and is freely available from the authors. CUWORX contains over 1500 lines of C language source code distributed among 35 subroutines.

A novel logic initialization technique for starting the WR algorithm is introduced. For all circuits

studied, logic initialization was as good as, or better than the uninitialized WR scheme. For the important synchronous sequential digital circuits with wide feedback loops, logic initialization substantially improves the convergence rate of the WR algorithm. In addition, logic initialization does not compromise in any way the multi-rate advantages of the WR scheme.

References

- [1] Nagel, L. N. (1975). "SPICE2: A computer program to simulate semiconductor circuits", Univ. of California, Berkeley, Memo. No. RRL-M520.
- [2] Chen, C. F., Lo, C. Y., Nham, H. N. and Subramaniam, P. (1984). Paper 2.2, *Proceedings of 21st Design Automation Conf.*, p. 10.
- [3] Saleh, R. A. and Newton, N. R. (1987). *Proceeding of IEEE ISCAS*, **600**.
- [4] Lelarsmee, E., Ruehli, A. E. and Sangiovanni-Vincentelli, A. L. (1982). *IEEE Trans. on CAD of IC and Syst.*, **CAD-I**, 131.
- [5] White, J. *et al.* (1985). *Transc. Soc. for Computer Simulation*, 2, p. 95. Also White, J. and Sangiovanni-Vincentelli, A. S. (1986). "Relaxation Techniques for the Simulation of VLSI Circuits", Kluwer Academic Publishers.
- [6] White, J. and Sangiovanni-Vincentelli, A., *Proc. Int. Custom Integrated Circuits Conference*, Rochester, N. Y., June (1984).
- [7] Dumlugol, D. *et al.* (1987). *IEEE Transc. on CAD for IC and Syst.*, **CAD-6**, 992.
- [8] Defebve, P. *et al.*, *Proc. Int. Conf. on Computer Design*, Rye, N.Y., Oct. (1984).
- [9] Carlin, H. and Vachoux, A., *European Conf. on Circuits Theory and Design*, **227**, Sept. (1987).
- [10] Jun, Y. H., Lee, C. W., Lee, K. J. and Park, S. B. (1987). *IEEE ISCAS*, **608**.
- [11] Gad El-Karim, G. J., "Large scale circuit simulation", *M. Sc. Thesis*, Cairo University, July (1988).
- [12] Saleh, R. A., Kleckner, J. E. and Newton, A. R., "SPLICE version 1.7 User's Guide", Dept. of Electrical Engineering and Computer Sciences, Univ. of California, Berkeley.

Authors' Biography

Dr. Serag E. D. Habib received his Ph.D. in Electrical Engineering From Toronto Univ., Canada in 1978. He is Presently a Professor of Electronics at the Electronics and Communications department, Faculty of Engineering, Cairo University, Giza, Egypt.

Dr. Habib research interests includes VLSI design, CAD tools for VLSI design, and Semiconductor devices, especially Power MOSTs and Solar cells.

In 1986, Dr. Habib received the Egyptian State Prize for his research on Semiconductor devices. He is the author of one British patent and over 30 Papers in international Journals and conferences.