Briefs

Uniform Repeater Insertion in RC Trees

Victor Adler and Eby G. Friedman

***Abstract*—Repeater insertion can be used to overcome the quadratic increase in the time required for a signal to propagate through an RC interconnect. A new timing model, based on short-channel $I-V$ equations, has been developed to characterize the signal delay through a resistive line. These analytical expressions provide the foundation for algorithms used to insert uniform repeaters into RC tree structures. Both local and global optimization algorithms for repeater insertion are presented. While the local optimization algorithm provides a computationally fast solution to the repeater insertion problem, the resulting circuit implementation is less power, area, and speed efficient than applying global optimization techniques. The global optimization algorithm for repeater insertion is achieved through the downhill simplex method. The circuit equations, algorithms, and software implementation of this repeater insertion system are presented in this paper.**

Results from these insertion methodologies improve delay from 25% to 60% versus typical cascaded buffer methodologies. Global repeater insertion further decreases delay times by up to 22% over the local repeater insertion method. The accuracy of the timing model characterizing the repeater insertion process as compared to SPICE simulations is generally within 10%. Applications of these algorithms for minimizing the signal delay through an RC tree, such as in data paths, and targeting signal delays through an RC tree, such as in clock distribution networks, are also discussed.

***Index Terms*—Buffers, delay, digital CMOS, repeaters, VLSI.**

I. Introduction

Interconnect delay has become a dominant performance limitation in high-speed integrated circuits. A common method of driving long interconnect is to insert a buffer at the beginning and the end of the interconnect line to improve the delay and slew rate of the signal. This method, however, does not necessarily minimize the delay caused by the large resistance encountered in long lines.

Bakoglu presents a methodology for inserting repeaters in a line to overcome the quadratic increase in delay due to a linear increase in interconnect length so that the RC interconnect impedance does not dominate the delay of a critical path [1]. Extensions to this repeater insertion methodology have also been reported in [2] and [3]. In [2] and [4], Wu and Shiau describe a repeater implementation to reduce interconnect delay. Their method uses a linearized form of the Shichman–Hodges equations [5] at a specific operating point to determine the proper repeater insertion locations. Nekili and Savaria consider optimal methods for driving resistive interconnect in [3]. They introduce the concept of parallel regeneration in [6], in which precharge circuitry is added to the repeaters to decrease the evaluation time. This technique requires fewer repeaters, however, extra area is necessary, adding parasitic capacitance. Furthermore, this technique requires a precharge signal for the circuit to operate correctly.

Dhar and Franklin present a mathematical treatment for optimal repeater insertion in [7]. They present elegant solutions to optimize repeaters with and without area constraints; however, the repeater is modeled as a simple linear resistor and capacitor and no closed form solution is provided. Other repeater insertion methods are described in [8]–[10]. In [11] and [12], buffer placement methodologies in multisink topologies based on minimizing the Elmore delay are presented.

An accurate timing model of a CMOS inverter driving an RC impedance has been presented by the authors in [13]. This circuit model is expanded to characterize repeater insertion in RC lines. This timing model for inserting CMOS inverters into RC lines, published previously by the authors in [13], [14] and briefly reviewed in Section II, has been applied to the development of a repeater design methodology and related algorithms for efficiently driving RC tree structures, such as a clock distribution network, so as to reduce both the signal delay and slew rate. In this methodology, the number and size of the repeaters to minimize the propagation delay and transition time from the root node to each leaf node are determined. The repeaters are restricted to the same geometric size and equal RC impedance per interconnect section within each branch. The equal size and section impedance conditions are known as uniform repeater insertion [1], [7], in which balancing the interconnect and repeater delay minimizes the total path delay along an RC line. As an alternative to uniform repeaters, tapered-buffer repeaters are examined in [14], and found to be less effective than simple uniform repeaters.

The focus of this paper is on introducing a methodology for determining the size and location for inserting uniform repeaters into RC trees. The algorithm and software implementation of two proposed methodologies, a local RC branch optimization methodology and a global RC tree methodology, are described in this paper. Both optimization methodologies are implemented with the downhill simplex algorithm. The efficacies of these two repeater insertion methodologies are compared to a standard cascaded buffer methodology [15]–[18]. Furthermore, the analytical equations characterizing the CMOS repeaters are shown to be accurate, generally within 10% of SPICE. The application of these local and global algorithms is also discussed in terms of relative run time and global optimality.

This paper is organized as follows: in Section II, a methodology for determining an optimal uniform repeater placement within an RC branch is presented. The local repeater insertion algorithm for RC trees is discussed in Section III. The global repeater insertion algorithm is discussed in Section IV. A comparison of the analytic model versus circuit simulation is presented in Section V. A comparison of the efficiency of the local- and global-optimal repeater insertion methodology versus using cascaded buffers to drive resistive tree-based interconnect is also described in Section V. Finally, some concluding comments are offered in Section VI.

II. Analytical Delay Model for RC Trees

An analytical model for determining the delay and placement of uniformly sized and spaced repeaters in RC trees based on Sakurai's α-power law is presented in this section [13], [14], [19], [20]. This model assumes that the transistor operates in the linear region, when

Manuscript received January 25, 1999; revised June 5, 2000. This work was supported in part by the National Science Foundation under Grant MIP-9423886 and Grant MIP-9610108, the Army Research Office under Grant DAAH04-93-G-0323, a grant from the New York State Science and Technology Foundation to the Center for Advanced Technology-Electronic Imaging Systems, and by grants from Xerox, IBM, and Intel.

V. Adler is with Sun Microsystems, Palo Alto, CA 94303 USA (e-mail: victor.adler@eng.sun.com).

E. G. Friedman is with the Department of Electrical and Computer Engineering, University of Rochester, Rochester, NY 14627 USA (e-mail: friedman@ece.rochester.edu).

Publisher Item Identifier S 1057-7122(00)08341-0.

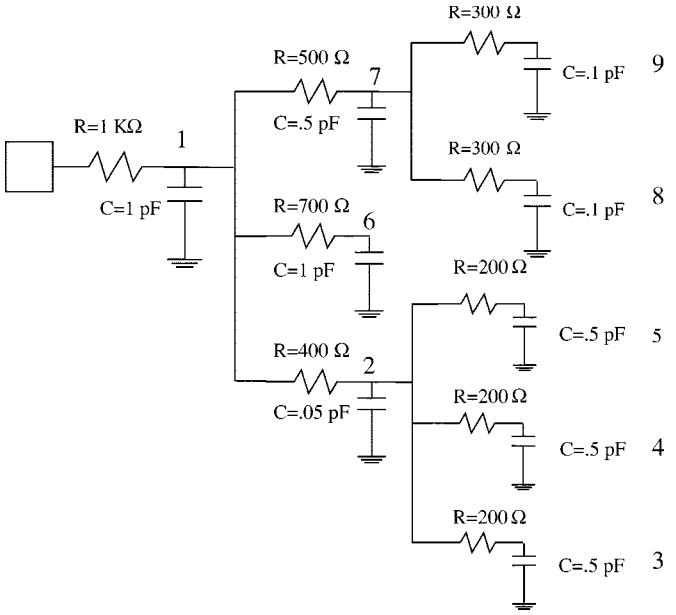

Fig. 1. Example of an RC tree. The large numbers are used to identify specific branches (note that the downstream nodes are to the right of the upstream nodes).

driving an RC load, since the linear region is the dominant region of operation when operating with fast input signals.

The structure of an RC tree is composed of a primary trunk with branching points. Each branch is modeled as a lumped resistance and capacitance, exemplified by Fig. 1. The total path delay is from the signal input at the root of the trunk to each end point of the tree (or leaf node).

The time required to drive a single branch of an RC tree using uniform repeaters, as shown in Fig. 2, is

$$t_{\text{branch}} = t_{\text{first stage}} + (n-2)t_{\text{int. stage}} + t_{\text{final stage}}. \quad (1)$$

The first component $t_{\text{first stage}}$ is the time required for the output of the first repeater in a branch to reach the turn-on voltage of the second repeater. The $t_{\text{int. stage}}$ component describes the time required for each repeater between the first and last stage to transition from $V_{DD}+V_{TP}$ to V_{TN} or vice versa. The last component $t_{\text{final stage}}$ is the time required to reach a given output voltage from either $V_{DD}+V_{TP}$ or V_{TN} [13], [20], [21]. $t_{\text{final stage}}$ also considers the effect of the additional capacitance C_{branch} of the downstream repeaters at a branching point.

The components $t_{\text{first stage}}$, $t_{\text{int. stage}}$, and $t_{\text{final stage}}$ utilize an expression derived from the Sakurai α-power law [19] for the delay of a CMOS inverter reaching an output voltage V_{out} given a step input signal [13]

$$t_{\text{out}} = \frac{(1+\mho_{\text{do}}R_{\text{int}})(C_{\text{rep/branch}}+C_{\text{int}})}{\mho_{\text{do}}} \ln\left(\frac{V_{DD}}{V_{\text{out}}}\right). \quad (2)$$

$\mho_{\text{do}}$ is the saturation conductance, a device parameter from the α-power law model derived from $(I_{\text{do}}/V_{\text{do}})$. I_{do} is the saturation current of the device when $V_{DS} = V_{DD}$. V_{do} is the voltage at which the device begins to operate in the saturation region [13], [19]. $C_{\text{rep/branch}}$ and C_{int} are the capacitances of the following inverting repeater and the interstage load capacitance, respectively. R_{int} is the resistance of the section of interconnect being driven by the repeater as shown in Fig. 2.

Each term in (1) is characterized by a step input to a single inverter driving an RC load, permitting a tractable solution of the delay time. This assumption permits the output waveform to be approximated by (2). The output waveform of the first stage is the input waveform of the following repeater. An example of this series of piecewise connections is shown in Fig. 3. The signal information describing the waveform shape permits a more accurate delay estimation as compared to estimating the path delay based on the classical Elmore delay model [22], [23]. Since the Elmore delay adds the products of a resistor (composed of the sum of the linearized repeater output resistance and the interconnect resistance) and all of the downstream capacitors, the Elmore delay does not account for the interaction of a repeater with the RC interconnect nor does the Elmore delay consider the shape of the output signal waveform. Thus, by integrating a more accurate timing model of a CMOS repeater into an algorithm for inserting repeaters into an RC tree, a more efficient circuit implementation can be achieved.

III. Local Branch Repeater Insertion Algorithm

A local optimization methodology and algorithm for inserting uniform repeaters into RC trees is presented in this section. This methodology is particularly appropriate if specific branch delays are being targeted. With the assumption that each branch has a repeater at its source, the minimum delay of each branch is initially determined. The total path delay from the root to each leaf is then minimized, according to the expressions summarized in Section II. The method for optimization is depth first, in which the lowest level branches are optimized first followed by each upstream branch. Thus, the RC tree is optimized locally, terminating at the root of the RC tree.

The algorithm to perform this repeater insertion process utilizes *a priori* information, characterizing the RC impedances and the number of sub-branches of each branch of the RC tree beginning at the root. The lowest level of the RC tree hierarchy is reached when all of the leaf nodes have zero branches. The RC tree is constructed in this top-down fashion.

A plot of the delay of branch 1 derived from (1) versus the size and number of repeater stages n in a branch is shown in Fig. 4 for $C_{\text{branch}} = 0$. That is, there is no final load capacitance at the end of the branch due to downstream repeaters of a different branch; however, there is a capacitance contributed by the repeaters within the branch. The optimal implementation of a repeater system for a specific RC load in terms of the number and geometric size of each repeater is represented by the minimum point on the graph. A similar graph can be drawn for each RC branch. The optimal number of repeaters inserted within a branch to minimize the total delay is determined from a numerical solution of the data illustrated in Fig. 4.

Once the tree has been constructed, it is traversed in a depth-first manner to determine the optimal repeater insertion for the final leaf nodes. When all of the branches of a parent have been optimized, the immediate upstream branch (or parent) is optimized, while considering the input capacitance of the repeaters of the downstream branches according to the method described in Section II. In Fig. 1, the branches 3, 4, and 5 are downstream from branch 2.

The pseudocode of the algorithm used to locally insert repeaters into each branch is shown in Fig. 5. The first function, `build_RCtree`, recursively builds each branch starting from the root and its sub-branches based on the specific branch resistances and capacitances. The second function, `insert_repeater`, is a recursive function, in which the minimum delay for inserting a uniform repeater system in a particular branch is determined. Note that the shape of the delay function describing a system of inserted repeaters in an RC branch is convex, so the local branch optimal repeater insertion system is quickly reached.

The performance improvement and accuracy are discussed more thoroughly in Section V. As described in greater detail in Section V, the path delay from the input of the RC tree to the final leaf nodes is

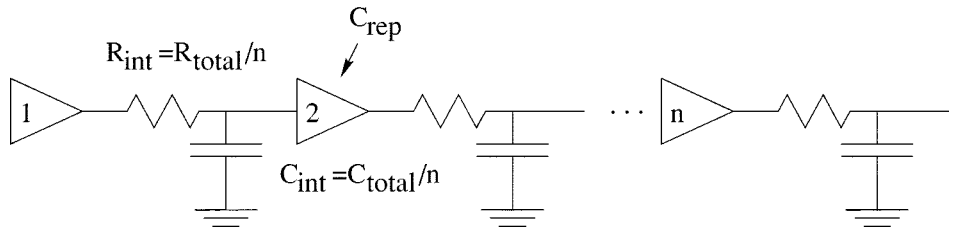

Fig. 2. n equal sized CMOS inverting repeaters driving a branch in an RC tree.

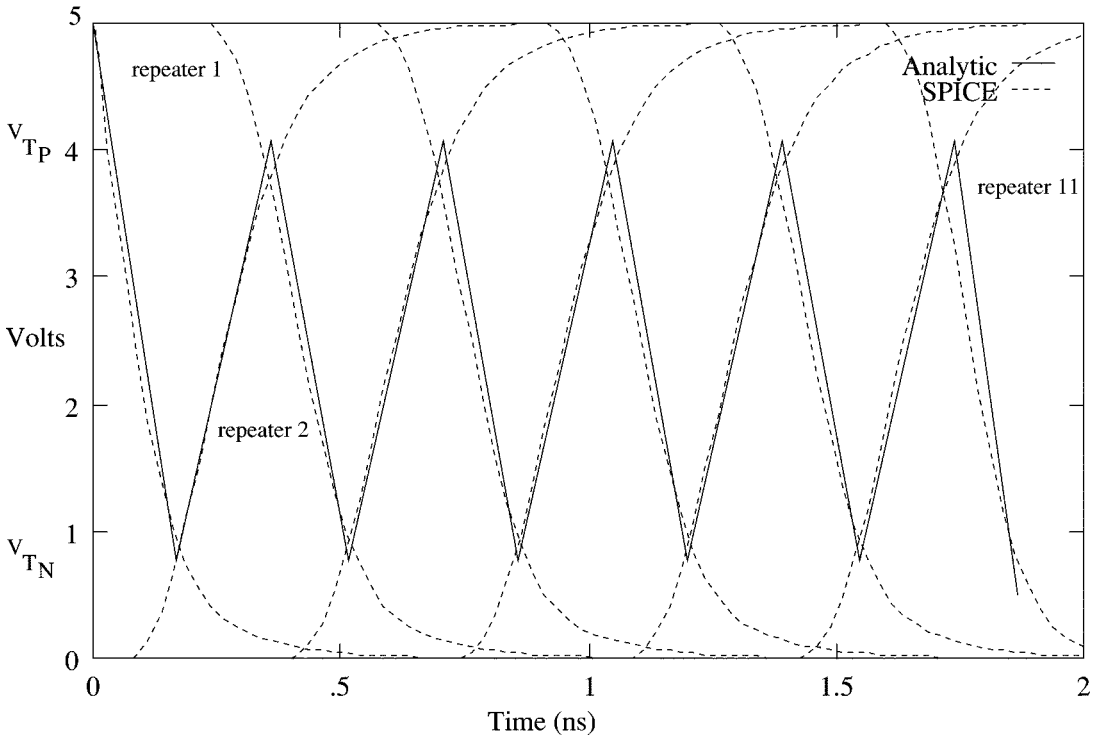

Fig. 3. Analytic and SPICE derived output waveforms of an 11-stage repeater chain driving an evenly distributed RC load of 1 kΩ and 1 pF.

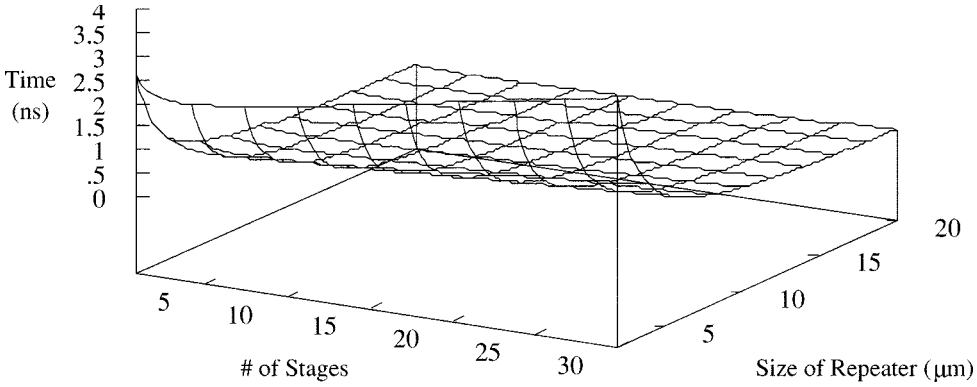

Fig. 4. Total delay for a branch as a function of the number of repeaters and repeater sizes. 0.8-μm CMOS technology, $C_{\text{branch}} = 0$, $R = 1$ kΩ, and $C = 1$ pF.

improved from 25% to 50% by the application of the local repeater insertion algorithm over a typical cascaded buffer insertion method. The accuracy of the local repeater methodology is within at least 10% of SPICE and typically within 5%. An example of the RC tree, shown in Fig. 1 after the local repeater insertion process is applied, is depicted in Fig. 6. Note that the number of repeaters inserted in each branch is shown inside the last repeater of that branch.

IV. Global Tree Repeater Insertion Algorithm

A global optimization algorithm to determine the size and number of uniform repeaters inserted within each branch of an RC tree is discussed in this section. The same timing model as described in Section II is used in the global optimization algorithm. The downhill simplex method of Nelder and Mead [24], [25] is used to implement the multidimensional optimization process.

```
(1)
function build_RCtree(node);
begin
   get R;
   get C;
   get number_of_branches;
   if (number_of_branches = 0)
      build_RCtree(branch);
   number_of_branches--;
end

(2)
function insert_repeater(tree);
begin
   if (number_of_branches > 2)
      insert_repeater(branch)
   optimize_delay[width,number_of_repeaters]
   number_of_branches--;
end
```

Fig. 5. Pseudocode of the local branch repeater insertion algorithm.

The flow of the repeater insertion methodology for determining the optimal size and location of each repeater is shown schematically in Fig. 7. In the downhill simplex method, each parameter variable being optimized is an element in an n-dimensional vector $\mathbf{x}$. To insert repeaters into an RC tree, the vector $\mathbf{x}$ contains the width and number of the uniformly sized and spaced repeaters within each branch. For example, in the RC tree shown in Fig. 1, x [1] is the width and x [2] is the number of repeaters to be inserted into branch 1. In this example, 18 elements are in $\mathbf{x}$, nine repeater widths and numbers, one pair for each of the nine branches. The number attached to each branch designates that branch.

The RC tree data is converted to a set of analytical expressions, describing the delays from the root node to each leaf node. This set of analytical expressions, in addition to the initial set of vectors and the objective function, are the inputs to the optimization routine. In order to initialize the downhill simplex algorithm, not just one starting point, but $(n+1)$ different arbitrary vectors are required. The n-dimensional initialization vectors are not permitted to lie along a straight line. The other input, the objective function, is the single value being minimized. Two useful objective functions appropriate for a repeater insertion algorithm are:

1) to minimize the delay from the trunk node to the leaf nodes such as in data paths with multiple fanout points;
2) to target the delay to each node such as in a clock signal path within a clock distribution network [26].

The former objective is specified by minimizing the average delay at each leaf node while the latter objective function minimizes the standard deviation of the predicted delay minus the target delay at each leaf node. In the example RC tree shown in Fig. 1 and in the example RC trees listed in Table I, the chosen objective function minimizes the average of the delays from the root of the tree to each of the leaf nodes of the RC tree. This objective function tends to minimize the delay through the trunk of the RC tree.

The results of the downhill simplex optimization method on uniform repeater insertion in an RC tree are summarized in Section V. The downhill simplex optimization produces a repeater implementation between 10% and 20% faster (with respect to the total path delay) than the application of the locally optimal repeater insertion methodology. In addition, the accuracy of the system of inserted repeaters implemented by the downhill simplex method is generally within 10% of SPICE. The RC tree shown in Fig. 1 is also shown in Fig. 8 after the global insertion algorithm has been performed. Note the decrease in circuit area (i.e., the total number of repeaters) and an approximately 20% decrease in path delay as compared to the circuit implemented by the local repeater insertion methodology as shown in Fig. 6.

V. Effectiveness, Accuracy, and Applications of Repeater Insertion Methodologies

A comparison of the local and global repeater insertion methodologies is presented in this section. The effectiveness of these repeater insertion algorithms are compared to both a classical cascaded buffer system and a completely passive RC tree (no buffers or repeaters). The system of inserted repeaters within the RC tree is also compared to SPICE to quantify the accuracy of the timing model. In addition, the global optimization is compared to an exhaustive search solution for a small example RC tree. Circuit applications of the local and global optimization algorithms are also discussed.

A. Accuracy and Effectiveness

The path delay t_{PD} from the root node to the end of each branch for three different trees is listed in Table I. The depth and impedance characteristics of each branch of these three trees are listed in the first three columns. The topology of each tree is characterized by the branch naming convention and indentation in the first column. In the fourth column, the path delay t_{passive} from the source to the end of each branch is listed. The RC impedances within the passive RC tree are modeled as a $\pi 3$ distributed load. In the fifth column, the cascaded buffer delay t_{buffer} from the tree source to each branch is listed. The cascaded buffer system is a series of optimally tapered buffers placed at the input of each branch, so as to drive the capacitive load of each branch (without considering the interconnect resistance) [18]. This delay assumes the cascaded buffer system uses a tapering factor of three [15], [17], [18].

The next three superior columns shown in Table I list similar information for the local branch repeater insertion methodology described in Section III and the downhill simplex method described in Section IV. For the local optimization, the predicted path delay is shown in column six, and the SPICE simulation and the associated error for the repeater insertion implementation are shown in columns seven and eight, respectively. The number and size of the repeaters are shown in columns nine and ten. Note that the maximum deviation of the analytic result from SPICE is 10% with a typical error of 5% or less.

An analysis of two branches listed in Table I is shown in Fig. 9. The delay of branches 1 and 8 from the first tree of Table I are shown for the four techniques (passive, tapered buffers, local repeater insertion, and global repeater insertion) of driving RC interconnect. A significant performance improvement is achieved with repeaters rather than buffers for those highly resistive branches, as exemplified by the relative performance improvements for branch 1 (1 KΩ) versus branch 8 (300 Ω). The area in terms of the total width consumed by inserting repeaters and buffers into the RC tree is shown at the bottom of the figure.

The signal waveforms at the final branch output of the locally optimized repeater system and the optimally tapered buffer system are shown in Fig. 10. The performance improvement of the repeater system over the tapered buffer system for this example RC tree is in the range of 25% to 33%. The buffer system does not drive the highly resistive lines effectively, hence longer than expected propagation delays and slower rise times are generated, particularly for highly resistive branches such as branch 6.

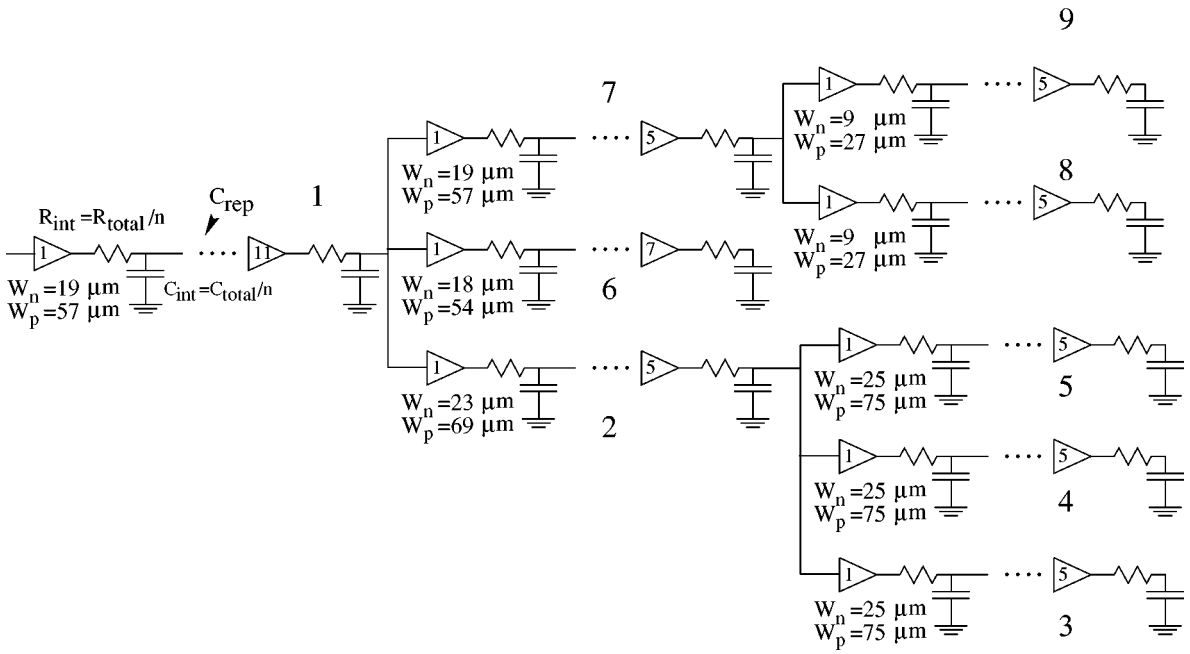

Fig. 6. RC tree shown in Fig. 1 synthesized by the local branch repeater insertion system. The transistor widths are shown below the first repeater of each branch, and the number of repeaters per branch is shown inside the last repeater of each branch.

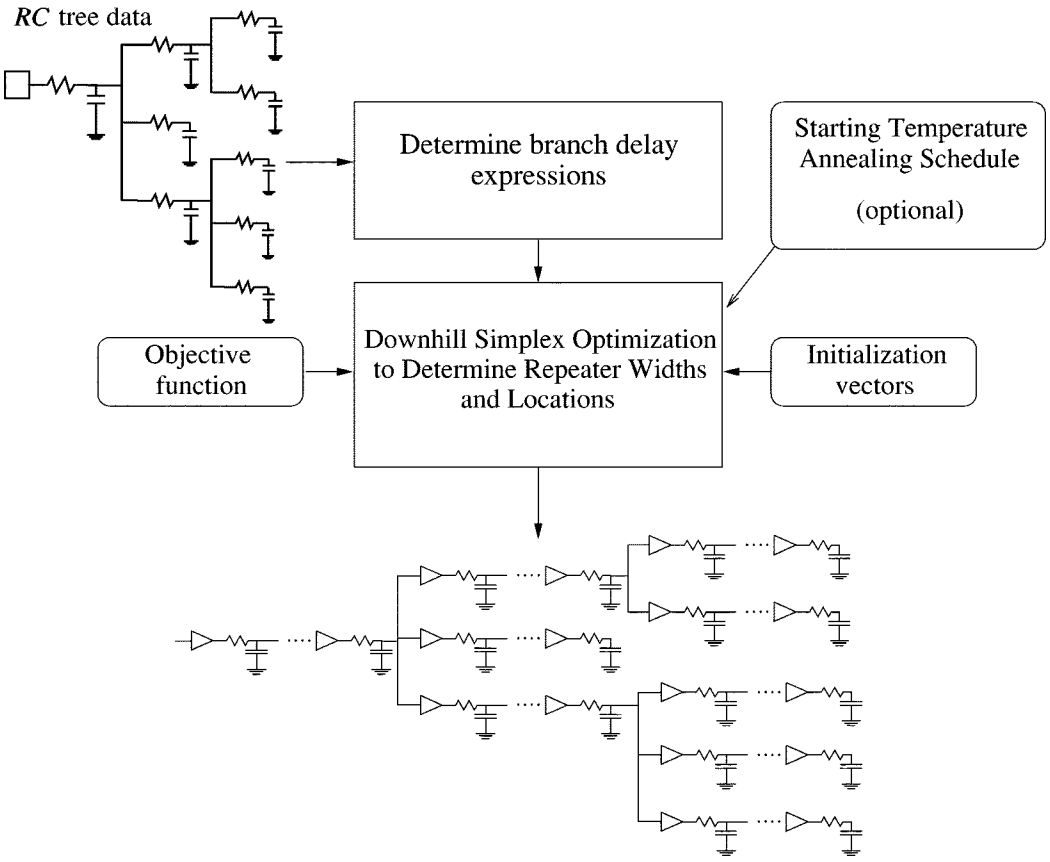

Fig. 7. Methodology for globally optimal repeater insertion.

For the downhill simplex method, similar information is shown in columns 11 through 15 in Table I. A comparison of SPICE simulations of the downhill simplex method exhibits branch delay improvements of up to 25% over the application of the local optimization method. Performance improvements derived from using the downhill simplex method over the local branch optimization algorithm are guaranteed, if one of the points of the initial simplex is the final result of the local optimization method. This improvement can be attributed to the

TABLE I

THE SIZE AND NUMBER OF REPEATERS AS DETERMINED BY THE LOCAL AND GLOBAL OPTIMIZATION ALGORITHMS FOR THREE DIFFERENT RC TREE TOPOLOGIES. (THE PROPAGATION DELAY IS IN NANOSECONDS, # IS THE NUMBER OF REPEATERS IN A BRANCH, SIZE IS THE GEOMETRIC WIDTH OF THE N-CHANNEL DEVICE OF THE UNIFORM REPEATER FOR THAT BRANCH, AND THE P-CHANNEL TO N-CHANNEL RATIO IS 3 : 1)

Branch	R	C	$t_{passive}$ Passive	t_{buffer} Buffers	Local Optimization					Downhill Simplex				
					t_{PD} Analytical	t_{PD} SPICE	Error	#	Size μm	t_{PD} Analytical	t_{PD} SPICE	Error	#	Size μm
1	1 KΩ	1 pF	9.0	1.95	1.05	1.16	9%	11	19	.88	.93	5%	7	12.7
2	400Ω	.05 pF	9.7	1.73	1.51	1.54	2%	5	23	1.14	1.15	1%	2	5.2
3	200Ω	.5 pF	9.75	2.41	1.77	1.70	4%	5	25	1.53	1.61	5%	4	5.7
4	200Ω	.5 pF	9.75	2.41	1.77	1.70	4%	5	25	1.51	1.57	4%	2	6.0
5	200Ω	.5 pF	9.75	2.41	1.77	1.70	4%	5	25	1.51	1.58	4%	2	5.9
6	700Ω	1 pF	9.45	2.98	1.71	1.67	2%	7	18	1.67	1.76	5%	6	7.3
7	500Ω	.5 pF	9.28	2.46	1.51	1.48	2%	5	19	1.35	1.45	7%	3	7.2
8	300Ω	.1 pF	9.3	2.57	1.70	1.67	2%	5	9	1.48	1.53	3%	2	4.1
9	300Ω	.1 pF	9.3	2.57	1.70	1.67	2%	5	9	1.52	1.59	4%	2	2.6
1	700Ω	.8 pF	7.95	1.36	.91	1.02	10%	9	24	.72	.85	15%	7	16.4
2	100Ω	.5 pF	7.98	1.70	1.12	1.09	3%	5	35	.98	1.08	9%	2	8.4
3	200Ω	.7 pF	8.18	1.86	1.27	1.23	3%	5	36	1.05	1.14	8%	3	17.1
4	700Ω	.6 pF	8.44	3.02	1.77	1.66	7%	5	15	1.56	1.61	3%	5	9.6
5	100Ω	.1 pF	8.19	2.14	1.42	1.40	1%	5	16	1.13	1.20	6%	2	6.4
6	1KΩ	1.6 pF	9.70	3.87	2.02	1.97	2%	11	20	1.79	1.80	1%	11	17.2
7	300Ω	.5 pF	9.79	3.72	2.33	2.22	5%	5	20	2.10	2.11	0%	2	11.3
8	600Ω	.1 pF	9.74	3.39	2.25	2.16	4%	5	6	1.94	1.93	1%	2	5.6
1	200Ω	5 pF	5.73	2.14	.85	.88	3%	9	79	.82	.86	5%	8	75.4
2	1 KΩ	1 pF	9.34	3.62	1.87	1.90	2%	9	19	1.72	1.77	3%	8	15.4
3	400Ω	.8 pF	9.54	3.95	2.30	2.17	6%	5	22	2.19	2.22	1%	5	10.9
4	1.5 KΩ	.1 pF	9.43	3.45	2.18	2.08	5%	5	4	1.9	1.95	3%	3	3.1
5	1.5 KΩ	.1 pF	9.43	3.45	2.18	2.08	5%	5	4	1.9	1.95	3%	3	3.1
6	400Ω	.8 pF	9.54	3.95	2.30	2.17	6%	5	22	2.19	2.22	1%	5	10.8
7	2 KΩ	.5 pF	7.45	3.61	1.79	1.82	2%	9	9	1.69	1.72	2%	8	7.5
8	800Ω	.2 pF	7.55	3.57	2.10	2.05	2%	5	8	1.99	2.01	1%	4	4.7
9	800Ω	.2 pF	7.55	3.57	2.10	2.05	2%	5	8	1.99	1.96	2%	3	5.1
10	2 KΩ	.5 pF	7.45	3.61	1.79	1.82	2%	9	9	1.70	1.74	3%	8	7.2
11	800Ω	.2 pF	7.55	3.57	2.10	2.05	2%	5	8	1.99	1.97	1%	3	5.4
12	800Ω	.2 pF	7.55	3.57	2.10	2.05	2%	5	8	1.99	1.97	1%	3	5.3
13	1 KΩ	1 pF	9.34	3.62	1.87	1.90	2%	9	19	1.75	1.77	1%	8	14.8
14	400Ω	.8 pF	9.54	3.95	2.30	2.17	6%	5	22	2.21	2.21	0%	4	11.2
15	1.5 KΩ	.1 pF	9.43	3.45	2.18	2.08	5%	5	4	2.02	1.96	3%	3	3.0
16	1.5 KΩ	.1 pF	9.43	3.45	2.18	2.08	5%	5	4	2.03	1.96	3%	3	3.0
17	400Ω	.8 pF	9.54	3.95	2.30	2.17	6%	5	22	2.19	2.42	10%	4	10.4

TABLE II

REPEATER INSERTION AS DETERMINED BY THE DOWNHILL SIMPLEX METHOD AND AN EXHAUSTIVE SEARCH FOR THE RC TREE SHOWN IN Fig. 11

Branch	R	C	Downhill Simplex			Exhaustive		
			t_{PD} (ns) Analytical	# of Repeaters	Size μm	t_{PD} (ns) Analytical	# of Repeaters	Size μm
1	1 KΩ	1 pF	.88	8	15.9	.88	8	16.0
2	700Ω	1 pF	1.55	7	13.5	1.55	7	13.5
3	500Ω	.5 pF	1.27	4	10.8	1.27	4	10.5

reduction in the size of the repeaters, which reduces the load capacitance at the branching nodes. Hence, not only is the delay decreased by globally optimizing the system, but the total area (and power) required by the repeater system is reduced when the downhill simplex method is applied, as compared to the local branch repeater insertion algorithm. Note that, on occasion, branches with similar impedance characteristics and parents can have different repeater implementations. This behavior is explained by the simplex solution falling into a nearby minimum, creating a slightly different repeater implementation. The run time of various implemented algorithms is discussed below.

A comparison of the downhill simplex method to an exhaustive search has been performed. The RC tree used for comparison is shown in Fig. 11 and is a three-branch section of the tree, illustrated in Fig. 1. A relatively small tree is used for comparison, due to the number of possible repeater implementations. A tree with b branches has $(n \times w)^b$ different possible implementations, where n is the number of repeaters that can be implemented within each branch and w is the number of possible discrete sizes of each of the uniformly sized repeaters. The number of possible implementations therefore can be enormous, thus the comparison to an exhaustively evaluated solution has been restricted to a tree with three branches. In the exhaustive search, the number of

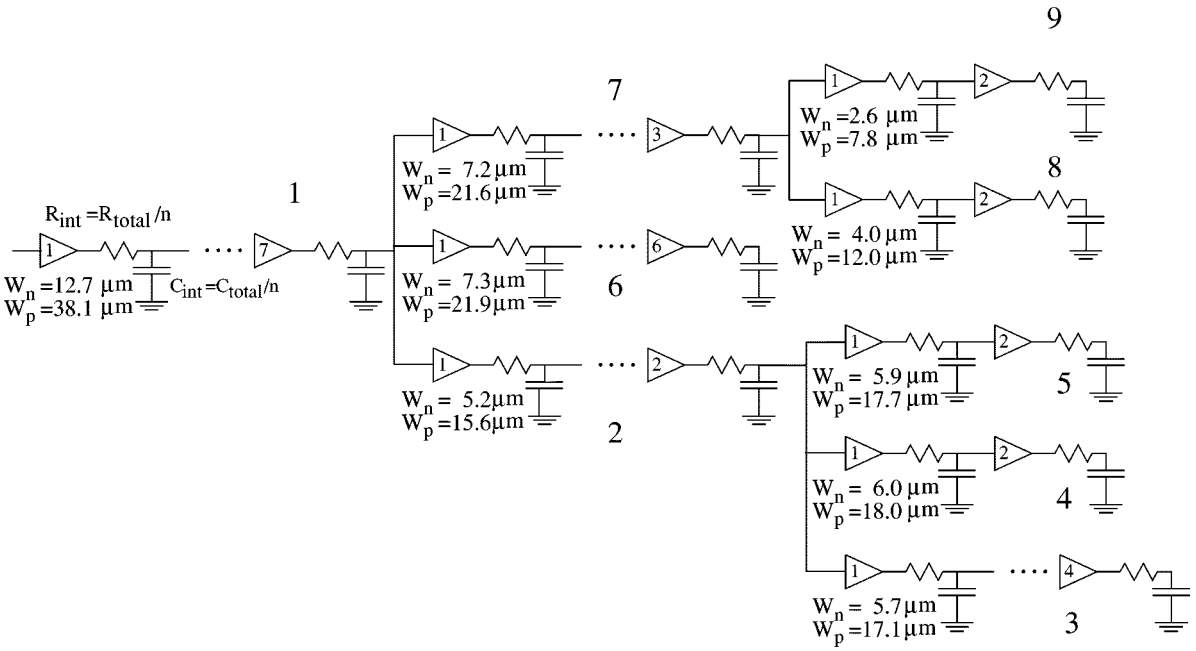

Fig. 8. RC tree shown in Fig. 1 synthesized by the global repeater insertion system. The transistor widths are shown below the first repeater of each branch, and the number of repeaters per branch is shown inside the last repeater of each branch.

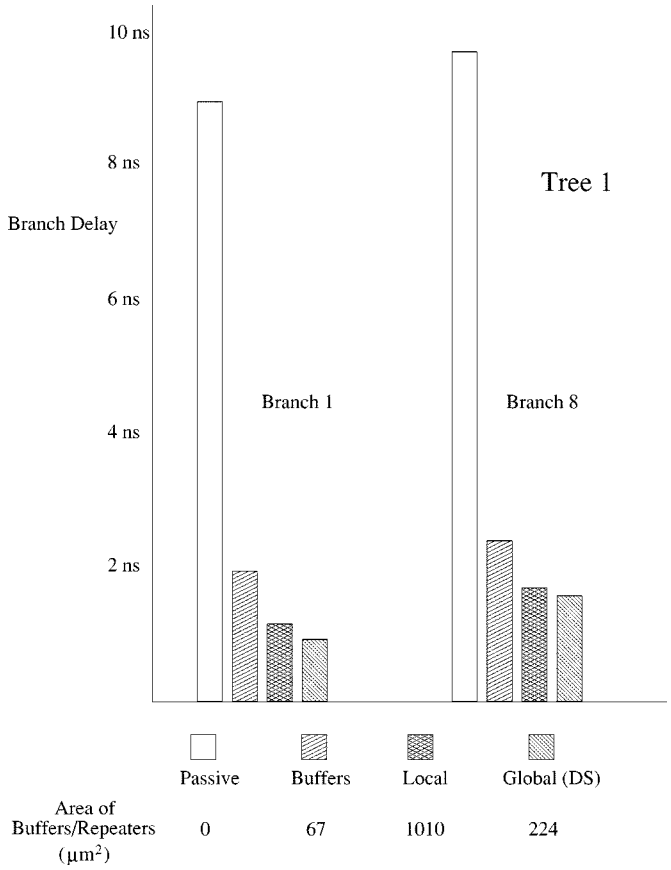

Fig. 9. Delay of branches 1 and 8 from the first tree in Table I. The total area cost of the inserted repeaters or buffers is shown at the bottom which is the sum of the total transistor widths.

repeaters in each branch ranges from 1 to 10, and the repeater size in each branch ranges from 1.0 to 25.0 μm in increments of 0.5 μm. While the local optimization algorithm provides a computationally fast solution to the repeater insertion problem, the resulting circuit implementation is less power, area, and speed efficient than applying global optimization techniques.

The results of the exhaustive search and application of the downhill simplex method on globally inserting repeaters into the circuit shown in Fig. 11 are listed in Table II. The position of each branch within the tree and the RC characteristics of the branch are described in columns one through three. The results of applying repeater insertion based on the objective function for the global optimization are shown in columns four through six. The same results for the exhaustive search are shown in the last three columns. The objective function minimizes the average root-to-leaf delay. In this comparison, the results derived from the exhaustive search match almost exactly the results derived from the heuristic search given the restrictions of the repeater size applied during the exhaustive search.

B. Applications

As mentioned previously, achieving a specific target delay may be the desired goal rather than minimizing the path delay. The downhill simplex algorithm can be used to determine a repeater insertion implementation for targeting a specific final leaf node delay. The objective function for this case minimizes the sum of the squares of the difference between the analytically determined delay and the desired target delay. Alternatively, targeting individual branch delays may be desirable. In this case, the local optimization algorithm is preferable, because the individual branch delays cannot be controlled within the global optimization algorithms. However, the optimization criteria may be significantly more complex than targeting a global delay depending upon the number of internal branches as compared to the number of leaves. In order to ensure the polarity of a set of repeaters within a branch, a two-pass optimization is performed. A first pass optimization is performed to determine the original repeater insertion. The objective criteria is then modified to limit the optimization to yield the desired polarity, and the optimization is performed a second time.

A comparison of run times or order of operations is important. In order to minimize the final branch delay using the local optimization method, a tree with n total branches results in n 3×3 matrices, resulting

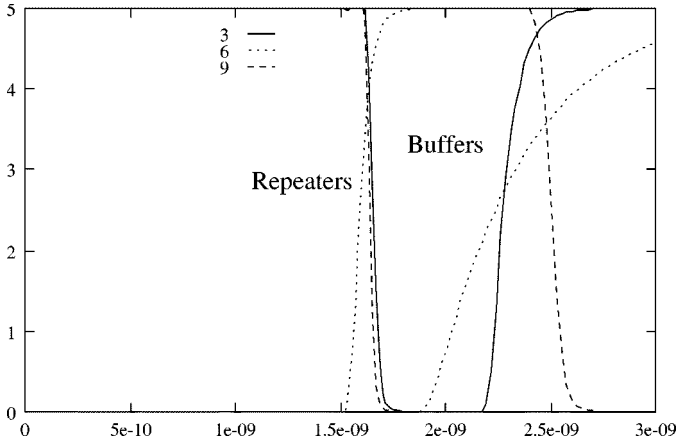

Fig. 10. Delay from the input of the RC tree to specific leaves of the tree based on the repeater insertion system as compared to applying optimally tapered buffers. Numbers indicate the leaf nodes as labeled in Fig. 6.

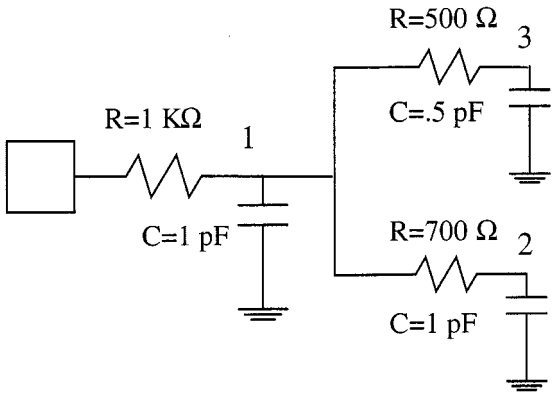

Fig. 11. Section of the RC tree shown in Fig. 1 used to compare the global optimization algorithm versus an exhaustive search.

in a complexity of $O(n)$. For the downhill simplex method for global optimization, an $n \times n$ matrix is required, resulting in a complexity of $O(n^2)$. However, a limit on the rate of convergence of the simplex can be set to reduce the computational run time.

VI. Conclusion

A design system for determining the optimal number and size of uniform repeaters to insert into an RC tree has been described. An accurate timing model based on a short-channel $I - V$ model, which considers the shape of the signal waveform is used within this system to achieve a more accurate and efficient repeater implementation. Analytical estimates of the total propagation delay of example RC trees with inserted repeaters agree within 10% of SPICE. A local optimization method and a global optimization method have been presented.

Depending upon the application, either delay targeting or delay minimization of the interconnect in RC trees may be appropriate goals. Both of these goals can be accomplished by the repeater insertion methods presented in this paper. The global repeater insertion algorithm is applied to minimize the total path delay or to satisfy specific root-to-leaf delays, while the local repeater insertion algorithm is applied to satisfy a specific branch delay objective.

Delay improvements of 25% to 60% over a typical cascaded buffer insertion methodology are achieved by inserting repeaters. Finally, the global repeater insertion methodology reduces the propagation delay, circuit area, and power dissipation as compared to the local optimization method. Thus, an integrated design system is presented in this paper for effectively and accurately inserting repeaters into RC trees.

Acknowledgment

The authors would like to thank Dr. S. Nassif of IBM of Austin Research Labs for his advice and technical assistance on the global optimization methodology.

References

[1] H. B. Bakoglu, *Circuits, Interconnections, and Packaging for VLSI*. Reading, MA: Addison-Wesley, 1990.
[2] C. Y. Wu and M. Shiau, "Accurate speed improvement techniques for RC line and tree interconnections in CMOS VLSI," in *Proc. IEEE Int. Symp. Circuits and Systems*, May 1990, pp. 2.1648–2.1651.
[3] M. Nekili and Y. Savaria, "Optimal methods of driving interconnections in VLSI circuits," in *Proc. IEEE Int. Symp. Circuits and Systems*, May 1992, pp. 21–23.
[4] C. Y. Wu and M. Shiau, "Delay models and speed improvement techniques for RC tree interconnections among small-geometry CMOS inverters," *IEEE J. Solid-State Circuits*, vol. 25, pp. 1247–1256, Oct. 1990.
[5] H. Shichman and D. A. Hodges, "Modeling and simulation of insulated-gate field-effect transistor switching circuits," *IEEE J. Solid-State Circuits*, vol. SC-3, pp. 285–289, Sept. 1968.
[6] M. Nekili and Y. Savaria, "Parallel regeneration of interconnections in VLSI & ULSI circuits," in *Proc. IEEE Int. Symp. Circuits and Systems*, May 1993, pp. 2023–2026.
[7] S. Dhar and M. A. Franklin, "Optimum buffer circuits for driving long uniform lines," *IEEE J. Solid-State Circuits*, vol. 26, pp. 32–40, Jan. 1991.
[8] C. Tretz and C. Zukowski, "CMOS transistor sizing for minimization of energy-delay product," in *Proc. IEEE Great Lakes Symp. VLSI*, Mar. 1996, pp. 168–173.
[9] C. Zukowski and C. Tretz, "Transistor sizing in CMOS logic chains to minimize energy-delay product," in *Proc. Workshop Academic Electronics in New York State*, June 1996, pp. 221–226.
[10] C. J. Alpert and A. Devgan, "Wire segmenting for improved buffer insertion," in *Proc. IEEE/ACM Design Automation Conf.*, June 1997, pp. 588–593.
[11] L. P. P. P. van Ginneken, "Buffer placement in distributed RC-tree networks for minimal elmore delay," in *Proc. IEEE Int. Symp. Circuits and Systems*, May 1990, pp. 865–868.
[12] J. Lillis, C.-K. Cheng, and T.-T. Y. Lin, "Optimal wire sizing and buffer insertion for low power and a generalized delay model," *IEEE J. Solid-State Circuits*, vol. 31, pp. 437–446, Mar. 1996.
[13] V. Adler and E. G. Friedman, "Delay and power expressions for a CMOS inverter driving a resistive-capacitive load," *Analog Integrated Circuits for Signal Processing*, vol. 14, no. 1/2, pp. 29–40, Sept. 1997.
[14] ——, "Repeater design to reduce delay and power in resistive interconnect," *IEEE Trans. Circuits Syst. II*, vol. 45, pp. 607–616, May 1998.
[15] R. C. Jaeger, "Comments on "An optimized output stage for MOS integrated circuits"," *IEEE J. Solid State Circuits*, vol. SC-10, pp. 185–186, June 1975.
[16] C. Mead and L. Conway, *Introduction to VLSI Systems*. Reading, MA: Addison-Wesley, 1980.
[17] B. S. Cherkauer and E. G. Friedman, "A unified design methodology for CMOS tapered buffers," *IEEE Trans. VLSI Syst.*, vol. 3, pp. 99–111, Mar. 1995.
[18] ——, "Design of tapered buffers with local interconnect capacitance," *IEEE J. Solid-State Circuits*, vol. 30, pp. 151–155, Feb. 1995.
[19] T. Sakurai and A. R. Newton, "Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas," *IEEE J. Solid-State Circuits*, vol. 25, pp. 584–594, Apr. 1990.
[20] V. Adler and E. G. Friedman, "Repeater design to reduce delay and power in resistive interconnect," in *Proc. IEEE Int. Symp. Circuits and Systems*, June 1997, pp. 2148–2151.
[21] ——, "Repeater insertion to reduce delay and power in RC tree structures," in *Proc. Asilomar Conf. Signals, Systems, and Computers*, Nov. 1997, pp. 749–752.
[22] W. C. Elmore, "The transient response of damped linear networks with particular regard to wideband amplifiers," *J. Appl. Phys.*, vol. 19, no. 1, pp. 55–63, Jan. 1948.
[23] J. Rubinstein, P. Penfield, and M. A. Horowitz, "Signal delay in RC tree networks," *IEEE Trans. Computer-Aided Design*, vol. CAD-2, pp. 202–211, Mar. 1983.

[24] J. A. Nelder and R. Mead, "A simplex method for function minimization," *Comput. J.*, vol. 7, no. 4, pp. 308–313, Jan. 1965.
[25] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C, The Art of Scientific Computing*. Cambridge, U.K.: Cambridge Univ. Press, 1988.
[26] E. G. Friedman, *Clock Distribution Networks in VLSI Circuits and Systems*. Piscataway, NJ: IEEE Press, 1995.

A Systolic Architecture for High-Performance Scaled Residue to Binary Conversion

G. C. Cardarilli, M. Re, R. Lojacono, and G. Ferri

***Abstract*—The scaled Chinese remainder theorem (CRT) is a very useful tool in residue arithmetic. Its properties can be exploited for the simplification and speeding-up of the conversion process. The main drawback presented by this methodology, when it is used for the output conversion, is the need of long wordlength look-up tables (LUTs) storing the correspondence among the modular numbers and the corresponding scaled terms of the CRT. This fact limits the maximum speed obtainable by this approach. In this brief, a new method for the computation of the scaled terms is presented. It has been implemented by using very small wordlength LUTs and simple arithmetic operators. The only proviso is that the moduli must be odd. The obtained architecture is very fast and due to the local interconnections is suitable for an efficient VLSI implementation.**

***Index Terms*—CRT, RNS, scaled output conversion.**

I. Introduction

There are a lot of applications (military, avionics, and telecommunication), in which digital high-speed architectures are used for real-time processing of large bandwidth signals [1], [2]. At present, the analog-to-digital converter technology allows high-resolution direct intermediate frequency (IF) sampling, and the actual trend will allow RF sampling in the near future. In this scenario, that implies the use of very fast processors, residue number system (RNS) constitutes an interesting method for obtaining high-speed and large dynamic range hardware structures. Its main advantage is the possibility to decompose large integer numbers in a set of N residue numbers $\{r_1, r_2, \cdots, r_N\}$ corresponding to the results of modulo operations, with respect to a set of pairwise coprime moduli $\{m_1, m_2, \cdots, m_N\}$. Considering the set $\{r_1, r_2, \cdots, r_N\}$, there exists only a number $X < m_1 m_2 \cdots m_N = M$ satisfying the congruencies $r_i = \langle X \rangle_{m_i}$, for $i = 1, 2, \cdots, N$. In RNS, an operation on large numbers is replaced by N-parallel and carry free operations on small numbers represented with $\lceil \log_2 m_i \rceil$ bits [3], [4]. For the most common dynamic ranges, $\lceil \log_2 m_i \rceil$ ranges from three to six.

The obtained results are finally converted into a conventional weighted representation. Usually this conversion is implemented by using the Chinese remainder theorem (CRT) algorithm [3], [4]. An important drawback of the conventional CRT is the requirement of a $\bmod M$ operation, where M is a very large number. This drawback can be overcome, by introducing a scaling factor. Consequently, the scaled CRT become a very useful technique in many RNS-implemented DSP-algorithms, that often require operations, such as sign detection, division, and overflow handling. Different scaled CRT-implementations have been presented in the literature [4]–[8], [10], [11]. The precision of the scaled output depends on the application and ranges from low, in the case of approximate sign detection [6], to very high precision, for exact sign detection and scaled conversion [7], [10], [11]. The works presented in the literature can be also classified, with respect to the used scaling factor. In [5], [7], and [10], a scaling factor of the form $P = 2^d/M$ has been used. In [8] and [11], a more general scaling factor has been used. In particular, in [8], $P \in [1, M]$, but in order to avoid the "*catastrophic error band*", a reduced dynamic range must be used. In the scaling factor, P must be such that $P > N$, being N the number of moduli used in the RNS representation.

Moreover, closed form solutions for the error computation in approximate CRT decoding of residue numbers have been presented in [12].

The brief is organized, as follows. A review of the scaled CRT is presented in Section II, while in Section III we introduce the algorithm for the recursive expansion of the terms (x_i/m_i). Moreover, we show how the new algorithm can be applied to the scaled CRT. In Section IV, an efficient hardware architecture implementing this algorithm is presented.

II. The Scaled CRT

By choosing $P = M$, the scaled CRT is defined as

$$\frac{X}{M} = \left\langle \sum_{i=1}^{N} \frac{\langle \hat{m}_i^{-1} r_i \rangle_{m_i}}{m_i} \right\rangle_1 = \left\langle \sum_{i=1}^{N} \frac{x_i}{m_i} \right\rangle_1 \tag{1}$$

where (x_i/m_i) are rational numbers, $\hat{m}_i = (M/m_i)$ and $\hat{m}_i^{-1}$ is the inverse of $\hat{m}_i$ defined as $\langle \hat{m}_i^{-1} \hat{m}_i \rangle_{m_i} = 1$. The scaling by M modifies the output range of the resulting CRT, that is now bounded in the interval $0 \le (X/M) < 1$. This introduces a modular operation, that discards all the bits of the integer part of (1). This modular operation is represented with the symbol $\langle \cdots \rangle_1$. By using this technique, the $\bmod M$ operation is substituted by the $\bmod 1$ operator, corresponding to the extraction of the fractional part of the term $\sum_{i=1}^{N} (x_i/m_i)$.

Unfortunately, the simplification of the modulo extraction implies the use of a set of look-up tables (LUTs) to store the fractional terms of (1).

In [7], to avoid any reconstruction error (i.e., the output is well ordered and the sign is exactly decoded) each term $(2 \cdot x_i/m_i)$ of the summation has been represented by using a LUT of m_i words of $\lceil \log_2(NM) \rceil + 1$ bits. It is important to point out that, while the number of the memory cells depends only on the modulus size (normally it is very small), their wordlengths grow with the output dynamic range and the number of moduli. This reduces the speed of the final RNS to binary converter limiting the advantages of using the RNS approach (see [9]).

Our work is aimed to increase the speed of the output converter, in order to equalize its performance with that of the inner processing structure. As a consequence, for example, if m_N is the biggest modulus, the maximum converter delay must be comparable with that of a $\bmod m_n$ adder, for any output range useful in practice.

Manuscript received April 29, 1998; revised June 5, 2000. This work was supported in part by MURST, under the national project "Co-design Methods of Low-Power Integrated Microelectronic Systems."

G. C. Cardarilli, M. Re, and R. Lojacono are with the Department of Electronic Engineering University of Rome "Tor Vergata," Italy (e-mail: g.cardarilli@ieee.org; marco.re@ieee.org; roberto.lojacono@uniroma2.it).
G. Ferri is with the Department of Electrical Engineering, University of L'Aquila, Italy (e-mail: ferri@ing.univaq.it).

Publisher Item Identifier S 1057-7122(00)08340-9.