Buffer Minimization in Pass Transistor Logic

Hai Zhou and Adnan Aziz

***Abstract*—With shrinking feature sizes and increasing transistor counts on chips, demands for higher speed and lower power make it necessary to look for alternative design styles that offer better performance than static complementary metal–oxide–semiconductors. Among them, pass transistor logic (PTL) is of great promise. Since delay in a transistor chain is quadratically proportional to the number of transistors and a signal may degenerate passing through a transistor, buffers are necessary to guarantee performance and restore signal strength in PTL circuits. In this paper, we first analyze effects of buffer insertion on a circuit and give a sufficient and necessary condition for safe buffer insertion. Then, a buffer minimization problem is formulated. Although it is NP-hard in general, it can be solved linearly when buffers are required on multifan-out nodes. We also consider the case when buffers are inverters, where phase assignment needs to be done with buffer insertion.**

***Index Terms*—Buffers, pass transistor logic performance.**

I. Introduction

At present, the technical limits of widely used existing circuit families, such as static complementary metal–oxide–semiconductors (CMOSs), threaten to hold back the development of microprocessors that can operate at speeds significantly above 1 GHz. For example, at such high speeds, it is predicted that the power dissipation of high-end integrated circuits fabricated using traditional static CMOS logic will exceed the practical limits of ceramic packages. This makes it necessary to look for alternative design styles which can offer better performance characteristics than static CMOS. Among them, pass transistor logic (PTL) is of great promise.

Traditionally, hand-crafted PTL has been successfully used to implement digital systems which are smaller, faster, and more energy efficient than static CMOS implementations for the same designs [5], [6], [8], [10], [13]. However, lack of a systematic methodology restricts the use of pass transistors in industry circuits. Recently, there have been a number of attempts developing synthesis tools targeting PTL. A top-down design flow for PTL called lean integration with pass transistors (LEAP) was proposed in [12]. There, designs are first converted into monolithic binary decision diagram (BDD) representations and then mapped to cells of a pass transistor library consisting of three function cells and four inverters with various drive capabilities. The approach in [2] also utilizes the natural efficient mapping between BDD and PTL. However, to control the BDD size, it uses decomposed BDD instead of monolithic BDD. Furthermore, instead of mapping to pass transistor cells, it converts BDD directly into a pass transistor netlist.

Since delay in a transistor chain is quadratically proportional to the number of transistors and a signal may degenerate when pass through a transistor (e.g., one through n-MOS or zero through p-MOS), buffers are necessary to guarantee performance and restore signals. In [12], buffers are inserted in a straightforward level-by-level fashion. This definitely guarantees the performance of the designs, but may incur a larger number of buffers than necessary. In [2], the height of each decomposed BDD is upper-bounded and a buffer is inserted at the root of each decomposed BDD. As we can see, since the height bound is only three or four, it overrestricts the flexibility of BDD synthesis and does not guarantee a minimum number of buffers either.

Manuscript received November 2, 2000. This paper was recommended by Associate Editor D. Hill.

H. Zhou is with the Advanced Technology Group, Synopsys, Inc., Mountain View, CA 94043 USA.

A. Aziz is with the Department of Electrical and Computer Engineering, University of Texas, Austin, TX 78712 USA.

Publisher Item Identifier S 0278-0070(01)03079-2.

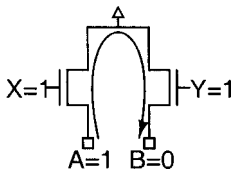

Fig. 1. Steady path from one to zero.

In this paper, we first demonstrate that buffer insertion on some positions can introduce sneak paths in a PTL circuit. We analyze how a sneak path is introduced and give the condition a node must satisfy to have a safe buffer insertion. We show that checking this condition is NP-complete. It has already been established that PTL circuits derived from BDD are sneak-path-free—this is one of the primary benefits of using BDD based approaches to PTL synthesis. Our results show that buffering can introduce sneak paths; we also prove that, in PTL circuits derived from BDD, safe buffer insertion positions can be easily computed. This reinforces the use of BDD based PTL synthesis.

Given a PTL circuit and a set of buffer insertion candidate positions, we need to consider the following buffer minimization problem, that is, insert a minimum number of buffers among these candidate positions such that the maximum number of consecutive transistors is upper-bounded by a given constant. We show that this problem is in general NP-hard. In reality, buffers are usually required on multifan-out nodes for drive strength. In this case, the problem can be solved in linear time.

Usually, buffers are implemented by inverters as they are superior in area, timing, and power. Inverter insertion changes a signal to its complement. Furthermore, in the original circuit, a complemented signal may be needed when passed from one stage to the next stage. Therefore, besides separating long paths into short paths, correct phases must also be decided upon for the whole circuit. We define this as an inverting buffer minimization problem. We deal with this problem in two phases: a noninverting buffer insertion and a global inverter minimization.

The rest of the paper is organized as follows. In Section II, effects of buffer insertion on a circuit are analyzed and conditions for safe buffer insertion are given. The buffer minimization problem when using noninverting buffers is solved in Section III and the corresponding inverting buffer minimization problem is dealt in Section IV. Experimental results on MCNC logic synthesis and optimization benchmarks are given in Section V. Section VI concludes the paper.

II. Conditions of Safe Buffer Insertion

A major difference between PTL and other logic styles (such as static CMOS logic, dynamic CMOS logic) is that in other logic styles, at most one kind of value (zero or one) can appear in a transit path (i.e., a sequence of drain/source connected transistors); but in PTL, a path may transmit either a zero or one at a different time. This may introduce a connection between one and zero in PTL circuits. As illustrated in Fig. 1, when $A = X = Y = 1$ and $B = 0$, there will be a steady current flow from A to B. We call the steady connection between zero and one a *sneak path*. A sneak path in a circuit wastes power and the resulting heat may even lead to catastrophic failure. Therefore, sneak paths should be avoided.

Definition 1: A PTL circuit without sneak paths is called a *safe* circuit.

A sneak path only happens when transit paths are wired together. Conventionally, this is called a *wired*-OR. We say that an output of a

TABLE I
TRUTH TABLE OF WIRED-OR

	0	1	Z
0	0	X	0
1	X	1	1
Z	0	1	Z

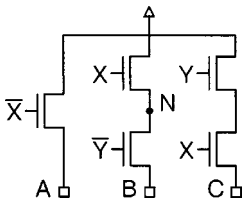

Fig. 2. Buffer inserted at N makes the circuit unsafe.

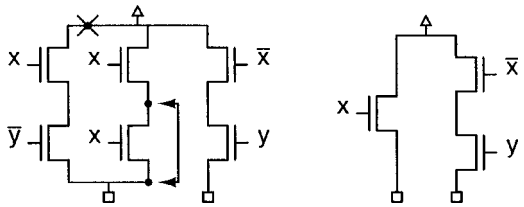

Fig. 3. (a) Redundant circuit and (b) its equivalent circuit.

pass transistor is in a *floating* state when its channel is turned off by the control signal. Representing a floating state by a Z and a sneak path by an X, the truth table of a wired-OR can be presented as shown in Table I.

Given a safe PTL circuit, buffers are inserted on transit paths to speed up and enhance pass signals, but can we guarantee a safe circuit after any insertion? The answer is no.

The reason is that in reality, buffers are not "ideal." That is, besides transferring a weak one or zero to a strong one or zero, it can not transfer a Z to a Z. Instead, it transfers a Z to either zero or one. Therefore, a buffer may transform a safe wired-OR of zero (or one) with Z to a unsafe wired-OR of zero with one. Fig. 2 shows a safe circuit, which will become unsafe when a buffer is inserted at node N.

In order to characterize positions where buffer insertions are safe, we need the following definition.

Definition 2: A safe PTL circuit is *redundant* if bridging a transistor (that is, connecting its source and drain) or floating a node (that is, cutting off all paths to that node) gives a safe circuit implementing the same function.

To illustrate this concept, a redundant circuit and the circuit after bridging a transistor and floating a node are shown in Fig. 3.

Based on the definition, we have the following lemma.

Lemma 1: Given an irredundant and safe circuit, buffer insertion before wired-OR or between two consecutive pass transistors is unsafe.

Proof: Suppose there is a buffer inserted before a wire-OR but the circuit is still safe. This means that the input of the buffer can not be floating or, when it is floating, other branches are also floating. Since the original circuit is safe, this also means that the signal after the wired-OR is totally decided by the signal at the input of the buffer. Therefore, the original circuit can be simplified by floating all other branches, which is a contradiction.

Similarly, suppose there is a buffer inserted on a node between two consecutive pass transistors, but the circuit is still safe. When the pass transistor before the node is off, there can not be any path from the node to any other nonfloating node. Therefore, bridging the pass transistor before the node will not change any signal at any nonfloating node. This contradicts the irredundancy of the original circuit. ■

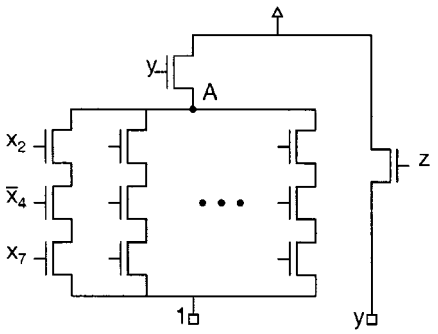

Fig. 4. Transforming 3-SAT to a circuit.

Equivalently, the lemma says that in an irredundant and safe circuit, safe buffering positions must be positions after wired-OR. However, the reverse is not true. That is, inserting buffers after some wired-OR may produce sneak paths. In fact, if under some input vectors a node after a wired-OR is floating, but there is a path from it to a nonfloating node, inserting a buffer on the node is still not safe. Therefore, we have the following theorem.

Theorem 1: In an irredundant and safe circuit, the safe buffer insertion positions are the nodes after wired-OR such that when it is floating there is no path from it to any nonfloating node.

Unfortunately, the above condition is not easy to check.

Theorem 2: Given an irredundant and safe circuit, decide whether a node is not safe for buffer insertion is NP-complete.

Proof: Given an irredundant and safe circuit, by guessing an assignment on control variables, the floating of a node and the existence of a path to a nonfloating node under this assignment can be checked in polynomial time. This means the problem is in NP.

We show the problem is NP-hard by transforming 3-SAT problem [3] to it. Given an instance of 3-SAT with n clauses, each having three literals, we can construct a circuit illustrated in Fig. 4 as follows. For each clause, we have from one to node A a path with three pass transistors. The control signals of these pass transistors are the complements of literals in the clause. That is, if a clause is composed of $\overline{x}_2$, x_4 and $\overline{x}_7$, then the control signals of the path will be x_2, $\overline{x}_4$, x_7. Under this construction, it is easy to see that a path is off if and only if the corresponding clause is satisfied. Therefore, the instance of 3-SAT is satisfiable if and only if A could be floating or, in other words, inserting a buffer on A is not safe. ■

However, for PTL circuits synthesized from BDD, this is not a problem.

Lemma 2: In a PTL circuit synthesized from BDD, a buffer can be safely inserted after any wired-OR.

Proof: Since each node in a BDD represents an everywhere defined Boolean function, each wire-OR in the circuit is always evaluated to a fixed voltage. Based on Theorem 1, a buffer can be safely inserted on this position. ■

III. BUFFER MINIMIZATION

A. General Buffer Minimization

In terms of circuit theory, a pass transistor has both capacitance and resistance. Hence, the delay of a transistor chain is quadratically proportional to the number of transistors [7]. It is known from today's static CMOS logic that transistor chains longer than three or four can be unacceptably slow [9]. Therefore, the main purpose of buffer insertion in a PTL circuit is to make sure that the maximum number of consecutive pass transistors is not greater than a given upper bound. We

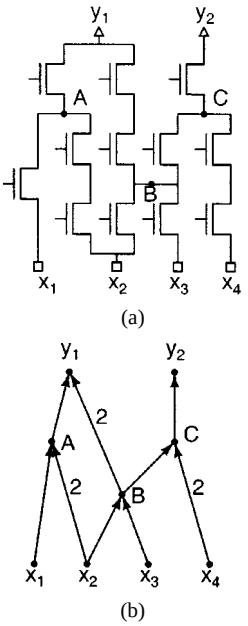

Fig. 5. Modeling buffer minimization problem by a DAG. (a) A PTL circuit. (b) Its DAG model.

hope to use as few buffers as possible to achieve this objective. Therefore, we need to solve the following problem.

Problem 1 (Buffer Minimization): Given a PTL circuit, a set of buffer insertion candidate positions, and a constant K, insert a minimum number of buffers such that the maximum number of consecutive transistors is not greater than K.

Given a buffer minimization problem, we can model it by an edge-weighted directed acyclic graph (DAG) $G = (V, E)$ as follows. The vertices V represent the buffer insertion candidate positions, primary inputs, and primary outputs. If there are transistor chains from vertex u to vertex v without going through other vertex, we let $(u, v) \in E$ and use the maximum length of all these chains as its weight. For a problem whose circuit and candidate positions are shown in Fig. 5(a), its corresponding DAG is shown in Fig. 5(b).

If there is an edge in the DAG whose weight is greater than K, that means there is a chain in the original circuit whose length is greater than K, but no buffer can be inserted within the chain. Therefore, there is no feasible solution to the buffer minimization problem. Since this can be simply checked, we assume this is not the case in the sequel. Using the DAG presentation, the decision version of the buffer minimization problem can be formulated as follows.

Problem 2 (Buffer Minimization-Decision): Given an edge-weighted DAG $G = (V, E)$ and two constants K and B, is there $V' \subseteq V$ such that $|V'| \leq B$ and after splitting each $v \in V'$ (that is, substituting v by two nodes v_1, v_2 and connecting in-coming edges to v_1 and out-going edges to v_2), there is no path whose length is greater than K?

Based on this formulation, we can show the following result.

Theorem 3: The buffer minimization problem is NP-hard.

Proof: This can be shown by a reduction from the problem of induced subgraph with property Π [3], which is defined as follows.

INSTANCE: Graph $G = (V, E)$, positive integer $M \leq |V|$.

QUESTION: Is there a subset $V' \subseteq V$ with $|V'| \geq M$ such that the subgraph induced by V' has property Π?

The problem is NP-hard for any property Π that holds for arbitrarily large graphs does not hold for all graphs and is "hereditary," that is, holds for all induced subgraphs of G whenever it holds for G. It is still NP-hard when G is restricted to a DAG.

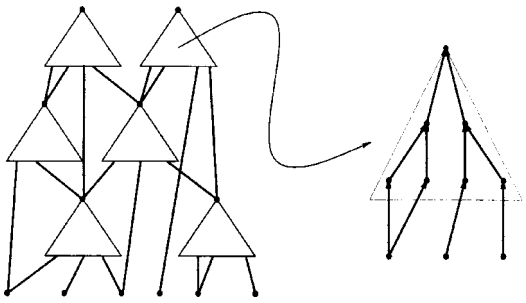

Fig. 6. Buffers on multifan-out nodes separate a DAG into leaf DAGs.

To prove Theorem 3, we define Π to be the following property: the longest path length is upper-bounded by a given constant N. It can be easily checked that the property fulfills the above conditions.

Given such an instance with a DAG $G = (V, E)$, constant M, and our property Π, an instance of the buffer minimization problem can be constructed as follows. Copy G and add two vertices s and t. Add edges from s to all vertices with zero in-degrees in G and edges from all vertices with zero out-degrees in G to t. Call this graph G'. Set each edge length to be one and let $B = |V| - M$, $K = N + 2$. We will show that an instance of induced subgraph with property Π has a yes answer if and only if the instance of the buffer minimization problem has a yes answer.

Suppose the answer to the instance of buffer minimization is yes, that is, there is a subset $V' \subseteq V \cup \{s, t\}$ with $|V'| \leq B = |V| - M$ such that, after splitting vertices in V', the longest path length is not greater than $K = N + 2$. Denote the graph after splitting by G''. The paths in G'' start and end with only vertices in $V' \cup \{s, t\}$. Therefore, deleting these vertices makes the longest path length not greater than N, but deleting them also gives a subgraph of G induced on $V - V'$. Since $|V - V'| \geq |V| - |V'| \geq M$, this means the answer to the problem of induced subgraph with property Π is yes. The other direction can be similarly proved. ■

B. Drive Capability Consideration

Besides the quadratic dependence of delay on the transistor chain length, a signal may degenerate when passing through transistors. Its strength then may not be strong enough to drive multiple fan-outs. Therefore, it is usually required to put a buffer on a multifan-out node. If the fan-out number is large, this buffer needs to be sized accordingly.

As shown in Fig. 6, the requirement of buffers on multifan-out nodes will split a DAG into separate *leaf-DAG*s. Here, a *leaf-DAG* is a DAG where only vertices with zero in-degree can have multiple out-going edges. In this situation, the buffer minimization problem becomes the following problem.

Problem 3 (Buffer Minimization for Leaf-DAG): Given an edge-weighted leaf-DAG and a constant integer K, insert a minimum number of buffers such that there is no path whose length is greater than K.

Contrary to the buffer minimization problem for general DAG, the above problem can be solved by an algorithm described in Fig. 7. It is a greedy algorithm and runs in linear time. The correctness is given by the following theorem.

Theorem 4: The greedy labeling algorithm gives an optimal solution to the buffer minimization problem for a leaf-DAG.

Proof: Since no buffer will be inserted on the leaves, buffer insertion on a leaf-DAG is equal to buffer insertion on a tree when leaves are duplicated. In any path whose length is greater than K, there must be a buffer. Since the number of buffers needed on a graph is not smaller than the number needed on its subgraph, inserting the buffer on the

```
Algorithm Greedy Labeling
  for each leaf v
    l(v) = 0;
  for each internal node v bottom-up {
    l(v) = max_{(u,v)∈E}(l(u) + w(u,v));
    if (l(v) + w(v, parent(v)) > K) {
      insert a buffer on v;
      l(v) = 0;
    }
  }
```

Fig. 7. Greedy labeling algorithm.

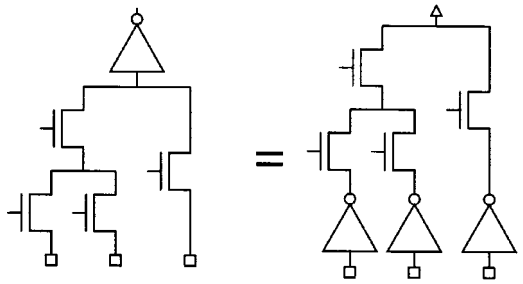

Fig. 8. Inverter can be moved freely in a PTL circuit.

highest level will result in a subgraph comparing with inserting on other nodes, which, therefore, gives an optimal solution. ■

IV. Inverting Buffers

n reality, buffers are usually implemented by inverters as they have better area, timing, and power characteristics. However, insertion of an inverter is more difficult as it changes a signal to its complement. Furthermore, in the original circuit, there may already be inverters on transit paths [2]. Due to the nature of PTL, we can get a complemented signal with the same circuit by complementing all the pass signals of the circuit, as shown in Fig. 8. This gives us the freedom to redistribute inverters around the circuit.

We assume that either a signal or its complement can be chosen to pass at a primary input and that a primary output can be implemented with arbitrary polarity. This is reasonable because we must have two polarities for control signals and some of primary outputs are used as control signals.

It follows from these assumptions that an inverter minimization problem can be formulated as follows.

Problem 4 (Inverter Minimization with Phase Assignment): Given a PTL circuit, a set of buffer insertion candidate positions, and a constant K, insert a minimum number of inverters in the circuit and decide a phase assignments at primary inputs such that each primary output implements the original function or its complement and the maximum number of consecutive transistors is not greater than K.

We deal with this problem in two phases. First, ignore all existing inverters and solve the buffer minimization problem on the circuit. At a position where a buffer is needed, two consecutive inverters are inserted. One of them is fixed at the position and is called a *hard* inverter; the other together with the preexisting ones can be moved around and are called *soft* inverters. An even number of consecutive soft inverters are canceled. An inverter at a primary input or primary output can be deleted by changing the polarity of the input or output. Therefore, the following problem needs to be solved in the second phase.

Problem 5: Given a PTL circuit with inverters and a set of buffer insertion candidate positions, leave a minimum number of inverters only at candidate positions and decide phase assignments at primary inputs such that each primary output implements the original function or its complement.

Generally speaking, the first phase can not be solved efficiently, but if buffers are required on multifan-out nodes, it can be solved in linear time.

Consider the second phase. Similarly, it can be modeled by a DAG. The vertices represent the buffer insertion candidate positions and the primary inputs and outputs. If there is a chain from u to v without going through any other vertex, we have an edge (u, v). If all such chains have odd (even) numbers of inverters, there is one (zero) inverter on the edge. If some have odd numbers but some have even numbers, then we have two edges between u and v, one with no inverter and one with one inverter.

Treating each internal vertex as a Boolean function and assuming it allows simultaneous polarity change on fan-ins and fan-outs, the above problem is actually the *global phase assignment* problem known in the literature. It was shown to be NP-complete by a transformation from MAX-CUT [11]. Heuristic algorithms were given in [4] and [11].

V. Experimental Results

We implement the two-phase algorithm described in the previous section for the inverting buffer minimization problem. It is required that each multifan-out node needs at least one buffer. Experiments are done on MCNC logic synthesis and optimization benchmarks.

Based on discussions in Section II, BDD is used for PTL synthesis. We use a simple threshold based BDD decomposition heuristic [1]; for simplicity, we do not optimize BDD sizes. Since our algorithm does not base on any specific feature of BDDs, it can be used on PTL circuits synthesized from other BDD methods or even non-BDD methods if insertion candidate positions are given.

Based on a related work [7], we choose three as an upper bound on the length of any transistor chain in PTL circuits. For comparison, an algorithm which inserts buffers level by level is also implemented. That is, in topological order, a buffer is inserted wherever a node fans out to more than two nodes or there are already three or more consecutive transistors. Wherever a node needs to provide two phases, an additional inverter is introduced. For our algorithm and the level-by-level approach, the numbers of needed buffers and inverters and the maximum delays are reported in Table II. Transistor parameters are based on a 0.5-μm technology [14]. Delays are computed by Elmore delay with the assumptions of no wire delay and the worst case down-stream capacitance [7].

VI. Summary

In this paper, we give the condition for safe buffer insertion in PTL circuits and deal with the problem of inserting the minimum number of buffers such that the number of consecutive transistors is upper bounded. We show that this problem is NP-hard in general and can be solved in linear time if a buffer is always needed at a multifan-out node. Although we ignored the true timing on electrical basis and used a simple-minded pass transistor path length for delay control, the problem defines a fundamental issue in buffering PTL circuits and was also considered in [2] and [12]. Built on the greedy algorithm for leaf-DAGs, our approach for inverting buffer minimization is an effective and efficient heuristic algorithm. However, on noncritical paths, a tight bound on the number of consecutive transistors may be relaxed and a buffer driving multiple fan-outs may need to be sized accordingly. Therefore, future work may include noncritical path area recovering and buffer sizing.

TABLE II
EXPERIMENTAL RESULTS

	Our algorithm				Level-by-level			
circuit	#buf	delay(ps)	#inv	delay(ps)	#buf	delay(ps)	#inv	delay(ps)
C432	922	370.75	1396	412.75	930	381	1760	339
C2670	378	145.75	465	151.75	442	146.2	740	146
C3540	2006	482.5	2752	506.5	2300	481.5	3913	406.2
C5315	2468	404.5	3273	458.75	2832	418	4478	412
C6288	8515	815.75	10568	843.25	9485	798	14349	762
C7552	3187	637.75	4415	710.75	3393	632.5	5262	636.8
s1	470	150.75	609	162.75	553	183	986	177
s27	3	28.25	5	31.25	5	38.5	11	42.8
s208	26	115	36	136	29	109	60	92.5
s298	38	52	55	55	54	63.8	86	56.8
s344	151	87	195	93	176	110.2	329	119.8
s349	151	87	195	93	176	110.2	329	119.8
s382	43	116.25	54	128.25	60	120.5	95	116
s1196	475	165.75	624	180.75	565	193	923	175
s1238	475	165.75	624	180.75	565	193	923	175
s1423	2563	444.75	3630	474.75	2737	713	4769	730.8
s1488	210	122.75	271	131.75	240	171	397	159.2
s1494	210	122.75	271	131.75	240	171	397	159.2

REFERENCES

[1] R. K. Brayton *et al.*, "VIS: A system for verification and synthesis," in *Proc. Conf. Computer-Aided Verification*, July 1996, pp. 428–432.

[2] P. Buch, A. Narayan, A. R. Newton, and A. Sangiovanni-Vincentelli, "Logic synthesis for large pass transistor circuits," in *Proc. IEEE/ACM Int. Conf. Computer Aided Design*, Nov. 1997, pp. 663–670.

[3] M. S. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco, CA: Freeman, 1979.

[4] A. Jain and R. E. Bryant, "Inverter minimization in multi-level logic networks," in *Proc. IEEE/ACM Int. Conf. Computer Aided Design*, Nov. 1993, pp. 462–465.

[5] T. Kuroda and T. Sakurai, "Overview of low-power ULSI circuit techniques," *IEICE Trans. Electron.*, vol. E78-C, no. 4, pp. 334–343, Apr. 1995.

[6] F. S. Lai and W. Hwang, "Differential cascade voltage switch with pass-gate (DCVSPG) logic tree for high performance CMOS digital systems," in *Proc. Int. Symp. VLSI Technology, Systems, and Applications*, May 1993, pp. 358–362.

[7] I.-M. Liu, T.-H. Liu, H. Zhou, and A. Aziz, "Simultaneous PTL buffer insertion and sizing for minimizing elmore delay," in *Proc. Int. Workshop Logic Synthesis*, May 1998, pp. 162–168.

[8] A. Parameswar, H. Hara, and T. Sakurai, "A high speed, low power, swing restored pass transistor logic based multiply and accumulate circuit for multimedia applications," in *Proc. Custom Integrated Circuits Conf.*, May 1994, pp. 278–281.

[9] J. Rabaey, *Digital Integrated Circuits*. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1996.

[10] T. S. Sheungm and K. Asada, "Regenerative pass transistor logic: A circuit technique for high speed digital design," *IEICE Trans. Electron.*, vol. E79-C, no. 9, pp. 1274–1283, Sept. 1996.

[11] A. Wang, "Algorithms for multilevel logic optimization," Ph.D. dissertation, Univ. California, Berkeley, CA, 1989.

[12] K. Yano, Y. Sasaki, K. Rikino, and K. Seki, "Top-down pass transistor logic design," *IEEE J. Solid-State Circuits*, vol. 31, pp. 792–803, June 1996.

[13] K. Yano, T. Yamnaka, T. Nishida, M. Satio, K. Shimohigashi, and A. Shimizu, "A 3.8-ns CMOS 16× 16-b multiplier using complementary pass transistor logic," *IEEE J. Solid-State Circuits*, vol. 25, pp. 388–395, Apr. 1990.

[14] *MCNC Designers' Manual*, MCNC, Research Triangle Park, NC.