Concurrent Logic and Interconnect Delay Estimation of MOS Circuits by Mixed Algebraic and Boolean Symbolic Analysis*

Sambuddha Bhattacharya and C-J. Richard Shi
Department of Electrical Engineering, University of Washington
Seattle, WA 98195-2500, USA
{sbb,cjshi}@ee.washington.edu

ABSTRACT: *Accurate estimation of delay in logic-stages and interconnects is of utmost importance in digital VLSI design. Conventional delay estimation techniques are numeric in terms of design parameters for both logic-stages and interconnect trees driven by them. In this paper, we present a symbolic method of computing delay in logic stages followed by interconnect trees. For each stage, our method provides a single analytic delay expression that is symbolic in terms of all input logic assignments as well as transistor and interconnect parameters. The method has been implemented and validated on modern digital VLSI technologies.*

1. Introduction

Efficient and accurate delay estimation for logic stages is fundamental to many VLSI automations like timing analysis and transistor sizing. For MOS circuits, delay computation is traditionally performed on the channel connected regions (CCR) that consist of conducting transistors connected to each other through their drains and sources. Varying input patterns at the gates of the transistors result in changing CCRs causing different signal delays in the logic stage.

Recently, delay estimation by symbolically representing input logic patterns has gained attention. [4] presents a BDD based method of estimating the signal delay in logic stages by replacing the transistors with their equivalent on-resistances and capacitances. A similar scheme of delay estimation based on the more general MTBDDs is presented in [3].

However, these methods suffer several deficiencies. First, these methods symbolically enumerate only the different input logic patterns at the gates of transistors. Changes in the transistors' sizes cannot be handled in these schemes without recreating the BDD/MTBDD structures as their delay calculation is inherently numeric. Second, as these methods are based on RC tree methods, loops of transistors cannot be handled. Third, severe inaccuracies result due to the Elmore model [2] and empirical handling of input rise/fall times. Finally, these methods cannot model the delay due to the interconnect parasitic networks driven by logic gates, which is becoming more important in deep submicron digital MOS design.

This paper introduces a novel approach that symbolically represents the delay of a logic-stage not only for all possible input logic patterns but also for all possible transistor parameters (sizes). We use a technique called multi-terminal determinant decision diagram (*MTDDD*), introduced recently for symbolic circuit analysis [5]. We extend MTDDD for efficient handling of Boolean conditions along with regular algebraic equations.

We use a delay estimate based on higher order circuit moments [6] which renders greater accuracy. Our method extends directly from the modified nodal analysis (MNA) circuit equations and therefore can correctly handle any circuit topology. We use a pre-computed lookup table for the equivalent resistors of conducting transistors and incorporate the slope of the input signal at the gate of the transistor into the table.

We further extend our method to the integrated estimation of signal delay in a logic-stage followed by the interconnect tree driven by the stage. Thus, the interconnect parameters are also symbolic variables along with transistor parameters and input logic patterns.

The paper is organized as follows. The formulation of delay estimation in terms of Boolean-ized moment equations is introduced in Section 2. Section 3 illustrates the computation of circuit moments using MTDDDs. Experimental results are presented in Section 4.

2. Formulation with Boolean-ized Moments

2.1 Background

A MOS transistor is modeled by a voltage controlled on-resistor between the drain and the source and three grounded capacitors at the drain, source and gate nodes. Interconnects are modeled as RC trees. The modified nodal analysis (MNA) based formulation of circuit equations [8] can be written as

$$\mathbf{C}\dot{\mathbf{x}} = \mathbf{G}\mathbf{x} + \mathbf{b}\mathbf{u} \qquad \mathbf{x}|_{t=0} = \mathbf{x}_0 \tag{1}$$

where $\mathbf{x}(t) \in \Re^n$ is a vector composed of node voltages and necessary branch currents, $\mathbf{G}$ is the modified conductance matrix, $\mathbf{C}$ is the capacitance and inductance matrix, and $\mathbf{u}$ is due to the system's input. The transfer function of such a linear network at any node can be expressed in terms of circuit moments as

$$H(s) = m_0 + m_1 s + m_2 s^2 + \ldots + m_n s^n + \ldots \tag{2}$$

The circuit moments then can be derived from (1) in a recursive form [6] where $\mathbf{m}_i$ is a vector composed of the i^{th} moments.

$$\mathbf{m}_0 = \mathbf{x}_h(0) \qquad \mathbf{G}\mathbf{m}_{k+1} = \mathbf{C}\mathbf{m}_k \tag{3}$$

The transfer function (2) is then matched to a lower order function, such as the one in (4), using Pade approximation [6].

$$\hat{H}(s) = \hat{m}_0 + \hat{m}_1 s + \hat{m}_2 s^2 \tag{4}$$

The propagation delay for the transfer function (4) can be estimated in terms of the 1st and 2nd moments. The delay metric we use in this work is adopted from [1] and is given as

* Supported by DARPA under Grant No. 66001-01-1-8920.

0-7803-7761-3/03/$17.00 ©2003 IEEE

$$t_{pd} = \ln(2)\frac{m_1^2}{\sqrt{m_2}} \quad (5)$$

2.2 Moment Equations and MTDDDs

Consider the pull-up circuit of an OAI21 and its equivalent RC network of Fig.1. Transistor M_i with Boolean X at its gate in the original circuit is replaced by a resistor R_{Mi} and a Boolean switch $\overline{X}$ (as the transistor is a PMOS). Node 2 in the equivalent circuit has capacitance contribution due to the transistors M_2 and M_3 of the original circuit. Similarly, the capacitance at node 3 is due to the load capacitance C_L and the contribution of the transistors M_1 and M_3.

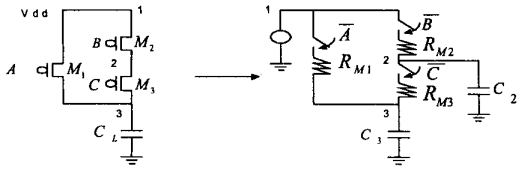

Figure 1: Pullup section of OAI21 and equivalent circuit.

The equation set (6) represents the recursive moment equations (3) for the equivalent RC network of Fig.1 when all the switches are closed. The k^{th} moment at the i^{th} node is represented as $m_k(i)$, whereas the k^{th} moment for current I_P is represented as $m_k(I_P)$. I_4 represents the current in the voltage source, and any other I_P represents the current in R_{MP}. The 3 equations on top represent KCL-type moment equations. The last 4 rows represent the branch equations.

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & -1 & 0 \\ -1 & 0 & 1 & R_{M1} & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & R_{M2} & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & R_{M3} & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} m_k(1) \\ m_k(2) \\ m_k(3) \\ m_k(I_1) \\ m_k(I_2) \\ m_k(I_3) \\ m_k(I_4) \end{bmatrix} = \begin{bmatrix} 0 \\ C_2 m_{k-1}(2) \\ C_3 m_{k-1}(3) \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (6)$$

Solving the equation set (6) is the same as repeatedly solving a set of linear equations in the following matrix form:

$$\mathbf{T}\mathbf{x} = \mathbf{b} \quad (7)$$

Similar to symbolic circuit analysis, our procedure for symbolic computation of moments is based on Cramer's rule for solving sets of linear equations. Then, the i[th] element of **x** is obtained as

$$x_i = \sum_j \mathbf{b}_j \det(\mathbf{T}_{ij}) \Big/ \det(\mathbf{T}) \quad (8)$$

where $det(\mathbf{T})$ is the determinant of the matrix **T**, and $det(\mathbf{T}_{ij})$ is the determinant of the matrix **T** after removing row i and column j, or the cofactor of the matrix **T** with respect to the element at (i,j). Therefore, the key task is the representation of the determinants and the cofactors of a semi-symbolic matrix.

For this purpose, we utilize an efficient technique called *Multi-Terminal Determinant Decision Diagram (MTDDD)*, introduced recently in the context of symbolic circuit analysis [5]. An MTDDD is an ordered, rooted, directed acyclic graph. As illustrated in Fig.2, it consists of some symbolic vertices and a set of terminal vertices, which can be the *0-terminal* vertex, the *1-terminal* vertex and some numeric terminal vertices with non-zero values. A symbolic vertex V, is characterized by a label *(V.label)*, and two edges, namely *1-edge* (solid line) and *0-edge* (dotted line) pointing, respectively, to its *1-child (V.1-child)* and *0-child (V.0-child)*. Thus, a vertex V represents a *semi-symbolic expression V.expr* defined recursively as follows:

If (V is 0-terminal), then V.expr=0
if (V is 1-terminal), then V.expr=1
if (V is numeric terminal), then V.expr=V.value
if (V is a symbolic vertex), then
V.expr=V.label(V.1-child).expr + (V.0-child).expr*

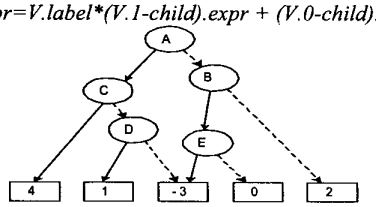

Figure 2: An MTDDD representing 4AC + AD -3A -3BE + 2.

To see how an MTDDD can be used to represent the determinant and cofactors of a circuit matrix, we consider the Laplace expansion of a matrix determinant with respect to a particular element, $t_{i,j}$, at row i and column j. Then, the matrix determinant $det(\mathbf{T})$ can be represented as follows:

$$\det(\mathbf{T}) = (-1)^{(i+j)} t_{i,j} \det(\mathbf{T}_{ij}) + \det(\mathbf{T}_{\bar{ij}}) \quad (9)$$

where, $det(\mathbf{T}_{ij})$ is the cofactor of the matrix **T** with respect to the element $t_{i,j}$ and $\det(\mathbf{T}_{\bar{ij}})$ is the remainder of the matrix **T** with respect to the element $t_{i,j}$, which is defined as the determinant of the matrix **T** after setting $t_{i,j}$ to 0. Clearly, if we can represent this expansion by a vertex with label $(-1)^{(i+j)} t_{i,j}$, the cofactor as the 1-child and the remainder as the 0-child, and then recursively expand the cofactor and the remainder, we can obtain an MTDDD. During this process, all the subgraphs can be shared.

2.3 Boolean-ized Moment Equations

Equation (6) is valid for one input logic pattern. For an n-input logic cell, the 2^n different input pattern sets result in 2^n sets of moment equations like (6). Naturally, creating MTDDDs for each such moment equation set is inefficient. This motivates the need for a single set of moment equations valid for all input patterns. This is accomplished by incorporating the Boolean input variables into the moment equations.

For the circuit of Fig.1, we recognize that the current in a branch is zero when the switch in the branch is open. That leads to the Boolean-ized branch I-V equation shown in Fig.3.

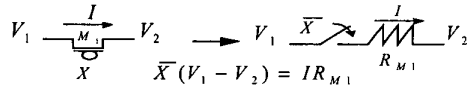

Figure 3. Boolean-ized MOSFET branch equation.

An isolated node results in a zero column in the $\boldsymbol{G}$ matrix of (6) rendering it singular. A node is isolated if there are no sensitizing paths from the voltage source to that node. In other words, the presence of currents in branches connected to a node and isolation of that node are mutually exclusive

events. The current I_2 in branch R_{M2} of Fig.1 exists if $\overline{B}$ is "1". Similarly, the condition for existence of current I_3 is

$$F_{I3} = \overline{A}\,\overline{C} + \overline{B}\,\overline{C} \quad (10)$$

The condition that node 2 is isolated is then given by

$$F_2 = !(\overline{A}\,\overline{C} + \overline{B}) \quad (11)$$

If node 2 is isolated, it does not affect any other node or branch that forms the CCR. So replacing the element *(2,2)* in the matrix by F_2 in (11) prevents the singularity in the matrix, and we forcibly set node 2 to 0v if it is isolated. Also, all the elements in the matrix $\boldsymbol{G}$ corresponding to the currents are replaced by respective Boolean functions. The complete set of changes to be incorporated into the $\boldsymbol{G}$ matrix of equation (6) is shown in Table 1.

Table 1. Complete set of changes in matrix G of (6).

Loc	Old	New	Loc	Old	New
(1,4)	1	$\overline{A}$	(3.6)	-1	$-(\overline{A}\,\overline{C} + \overline{B}\,\overline{C})$
(1,5)	1	$\overline{B}$	(4,1)	-1	$-\overline{A}$
(2,2)	0	$!(\overline{A}\,\overline{C} + \overline{B})$	(4,3)	1	$\overline{A}$
(2,5)	-1	$-\overline{B}$	(5,1)	-1	$-\overline{B}$
(2,6)	1	$(\overline{A}\,\overline{C} + \overline{B}\,\overline{C})$	(5,2)	1	$\overline{B}$
(3,3)	0	$!(\overline{A} + \overline{B}\,\overline{C})$	(6,2)	-1	$-\overline{C}$
(3,4)	-1	$-\overline{A}$	(6,3)	1	$\overline{C}$

3. MTDDDs for Boolean-ized Moments

The Boolean-ized moment equations thus obtained are now converted to MTDDD structures according to the algorithm presented in Fig. 4. The MTDDD operations *cofactor, union, multiply* and *getvertex* described in [5][7], are modified to incorporate the simplification due to the presence of Boolean and numeric elements. The algebraic and Boolean symbols occupy places closer to the root while the numeric elements are pushed down to the leaf terminals. The procedure *getvertex(top,D_1,D_2)* generates an MTDDD vertex for the element *top* with the sub-graphs rooted at D_1 and D_2 as its 1-child and 0-child respectively. The procedure *cofactor(G-{j,k})* returns an MTDDD vertex representing the cofactor of the matrix with respect to the element at *{j,k}*. The *multiply(top,P)* operation returns an MTDDD vertex corresponding to the multiplication of the MTDDD *P* with the numeric element *top*. The MTDDD obtained is similar to the MTDDD *P* except that the terminal vertices are the product of the terminal values of *P* with the value of *top*.

```
Create_mtddd
    List_boolean_conditions
    for (i = 1 to MAX_MOMENT_ORDER)
        moment[i,j] = NULL
        for (j: capacitive node)
            for (k: capacitive node)
                P = cofactor(G - {j,k})
                if (C(k) is SYMBOLIC)
                    Q = getvertex(C(k),P,0-terminal)
                else
                    Q = multiply(C(k),P)
                moment[i,j]=union(moment[i,j],Q)
```

Figure 4: Algorithm for setting up MTDDD for moments.

Procedure *List_boolean_conditions* in *Create_mtddd* lists the Boolean functions in the modified $\boldsymbol{G}$ matrix. This is different from identification of sensitization conditions for cell level networks as loops of sensitized paths can be obtained for such transistor level networks. An efficient tree-link based implementation for identifying the Boolean conditions is presented in Fig. 5. First, the Boolean conditions along a spanning tree of the switched-resistor network are enumerated. Sensitization through the link branches are considered next and the corresponding Boolean function at each node and branch of the switched-resistor network are updated. For each switch-resistor link branch, an update in the sensitization function at either of its two incident nodes **i,j** triggers further enumeration. In case of an update in the function at node **i**, the branches in the tree connected to the node **j** are updated with the new function at node **i**. The Boolean functions are compactly stored as MTDDD trees.

```
List_boolean_conditions
    create_spanning_tree
    create_list_of_links
    List_boolean_along_tree
    for (; ;)
        foreach branch in list_of_links
            L = branch . get_node1_function()
            R = branch . get_node2_function()
            update_tree (node1, R, branch)
            update_tree (node2, L, branch)
        if (no_update_in_one_pass)
            break

update_tree (node N, function F, branch B )
    if (loop_formed_by_node N)
        return
    G = update_branch_function ( B , F)
    foreach ( tree branch X at node N)
        nextnode = X. get_2nd_node()
        if ( ! X . updated() )
            update_tree ( nextnode, G, X)
    return
```

Figure 5: Algorithm for enumerating Boolean sensitization.

The progression of the Boolean enumeration algorithm for the example circuit graph of Fig. 6 is presented in Table 2. In Fig. 6, the solid and the dashed lines represent the tree and link branches, respectively. Each branch has a Boolean variable associated with it. Node 1 is the start node and node 3 the sink. First, a traversal through the spanning tree from the source to the sink sets up the Boolean conditions at every node (Table 2, column 2). The remaining columns represent the additional Boolean functions at the nodes due to each link branch.

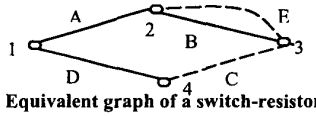

Figure 6: Equivalent graph of a switch-resistor circuit.

Table 2. Progression of Boolean enumeration algorithm.

Node #	Tree Setup	Due to C	Due to E	Due to C
2	A	DCB	DCE	-
3	AB	DC	AE	-
4	D	ABC	-	AEC

Once the MTDDDs for moments are set up, the evaluation of moments involves simple traversals through the

MTDDD tree. The sharing of sub-graphs in the MTDDD trees enables efficient calculation of moments, as sub-graphs evaluated once need not be traversed again during the entire moment calculation process. The generation of expression for the moment at any node in the circuit requires a traversal through the MTDDD tree.

Figure 7 shows a nand2 cell, its equivalent circuit and a part of the corresponding MTDDD for the first moment at node 1.

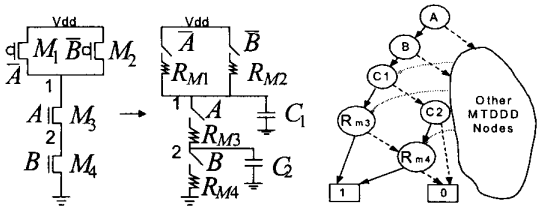

Figure 7: A nand2 gate, its equivalent circuit and MTDDD.

4. Experimental Results

The proposed symbolic procedure has been implemented in the program SAMBA (Symbolic Analysis Mixing Boolean and Algebra). SAMBA generates a single completely symbolic expression for delay under any input pattern. For the nand2 cell of Fig. 7, the 1st moment at the output node under any input logic pattern is given as

$$m_1(1) = n / d$$

$$d = (1-A)BR_{M2}R_{M3} + A(1-B)R_{M1}R_{M4} + ABR_{M1}R_{M2} + (1-A)(1-B)(R_{M1}R_{M3}R_{M4} + R_{M2}R_{M3}R_{M4})$$

$$n = (1-A)BR_{M1}R_{M2}R_{M3}C_1 + (1-A)(1-B)R_{M1}R_{M2}R_{M3}R_{M4}C_1 + A(1-B)(R_{M1}R_{M2}R_{M4}C_1 + R_{M1}R_{M2}R_{M4}C_2) + AB(R_{M1}R_{M2}R_{M4}C_1 + R_{M1}R_{M2}R_{M4}C_2 + R_{M1}R_{M2}R_{M3}C_1)$$

Semi-symbolic expressions with most transistor parameters as numeric and only a few as symbolic can also be obtained.

The computation of moments involves library lookup for the equivalent resistance of transistors. Our library generation scheme [9] uses a novel technique that incorporates effects of circuit topology, input signal slope and events at the input in addition to transistor sizes and loads for the resistance computation and results in excellent accuracy.

We apply our method of delay computation on a standard cell library in TSMC 0.18 micron technology. Experiments are conducted with different p-transistor and n-transistor sizes, loads, input patterns and input signal slopes. The maximum error is less than 5% when the estimated delays are compared to HSPICE. We present the results for a 6-input AOI321 cell and an RC interconnect tree driven by it in Fig. 8. The estimated delay in SAMBA is compared with HSPICE under parameter variation. The results show excellent match with HSPICE values.

5. Conclusion

A symbolic approach for the accurate estimation of delay in a logic stage followed by interconnect tree is presented. Our approach leverages MTDDDs to enable the

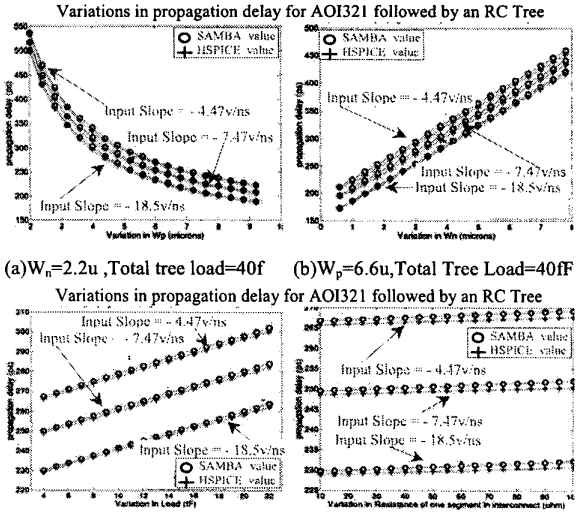

Figure 8: Rise delay variation in SAMBA and HSPICE in AOI321 cell followed by an RC Tree. The sensitized CCR consists of 3 PMOS in series and 3 NMOS. (a) Wp variation (b) Wn variation, (c) Load variation at interconnect sink (d) interconnect Resistance variation. Test results shown for 3 different input signal slopes.

efficient and compact symbolic representation of delay. Under this scheme, the equivalent resistor and capacitor of transistors, resistors and capacitors of interconnect tree and input logic assignments are all symbolic. For each stage, we provide a single delay expression that holds for all possible input assignments as well as all possible sizes of the transistors. This technique has potential application in behavioral simulation, timing analysis, variational analysis and concurrent transistor and interconnect sizing.

6. References

[1]. C. J. Alpert, A. Devgan and C. V. Kashyap, "RC delay metrics for performance optimization", *IEEE Trans. Computer-Aided Design*, vol. 20, pp. 571-582, May, 2001.

[2]. W. C. Elmore, "The transient response of damped linear networks", *J. Appl. Phys.*, vol. 19, pp. 55-63, Jan. 1948.

[3]. C. B. McDonald and R. E. Bryant, "Computing logic-stage delays using circuit simulation and symbolic elmore analysis", *Proc. 38th IEEE/ACM Design Automation Conf., Las Vegas, NV*, June 2001, pp. 283-288.

[4]. M. P. Desai and Y. T. Yen, "A systematic technique for verifying critical path delays in a 300MHz alpha CPU design using circuit simulation", *Proc. 33rd IEEE/ACM Design Automation Conf., Las Vegas, NV, June 1993, pp. 125-130.*

[5]. T. Pi and C-J. R. Shi, "Multi-terminal determinant decision diagrams: A new approach to semi-symbolic analysis of analog integrated circuits", *Proc. 37th IEEE/ACM Design Automation Conf., Los Angeles, CA*, June 2000, pp. 19-22.

[6]. L. T. Pillage and R. A. Rohrer, "Asymptotic waveform evaluation for timing analysis", *IEEE Trans. Computer-Aided Design*, vol. 9, pp. 352-366, Apr. 1990

[7]. C-J. R. Shi and X-D. Tan, "Canonical symbolic analysis of large analog circuits with determinant decision diagrams", *IEEE Trans. Computer-Aided Design*, vol. 19, pp. 1-18, Jan. 2000.

[8]. J. Vlach and K. Singhal, *Computer methods for circuit analysis & design*, New York: Van Nostrand Reinhold, 1983.

[9]. S. Bhattacharya and C-J. R. Shi, "A Table Lookup Method for Effective Resistance Estimation for Digital VLSI Delay Calculation", Department of Electrical Engineering, University of Washington, Technical Report TR-12, Oct. 2002.