Delay Analysis for Current Mode Threshold Logic Gate Designs

Chandra Babu Dara, Themistoklis Haniotakis, *Member, IEEE*, and Spyros Tragoudas

***Abstract*— Current mode is a popular CMOS-based implementation of threshold logic functions, where the gate delay depends on the sensor size. This paper presents a new implementation of current mode threshold functions for improved gate delay and switching energy. An analytical method is also proposed in order to identify quickly the sensor size that minimizes the gate delay. Simulation results on different gates implemented using the optimum sensor size indicate that the proposed current mode implementation method outperforms consistently the existing implementations in delay as well as switching energy.**

***Index Terms*— Current mode, operating speed, sensor sizing, threshold logic gates (TLGs).**

I. Introduction

EXPONENTIAL savings in the performance of digital circuits due to parameter scaling have disappeared [4]. Alternative technologies, such as threshold logic gates (TLGs), among others, can extend parallel processing capabilities [4]–[8]. A TLG is an N-input device that calculates the weighted sum of inputs [3]. Current mode, monostable–bistable transition logic element, neuron MOS, and single electron technology are a few examples for the design of TLGs [6], [9], [10]. Some of these methodologies are CMOS-based and the synthesis of efficient TLG-based circuits becomes feasible [12], [15].

Logical processing in TLGs is more sophisticated than the traditional Boolean gates, and TLGs can implement complex logic functions [2]. In a TLG, weights are the principal elements that define the functionality of a gate.

A basic TLG consists of N-inputs, a weight value for each input, and a threshold weight. The sum of the input weights is compared with the threshold weight. If it is greater than the threshold weight, then the digital output of TLG is logic high, and if it is less it will be logic zero [3]. In the CMOS-based implementation considered in this paper, when the sum of the input weights is equal to the threshold weight, then the gate is in undefined state. Weights are selected so that this case is avoided. The equation representing the output of a TLG is given as

$$f = \begin{cases} 1 & \text{if } \sum_i w_i \cdot x_i > w_T \\ 0 & \text{if } \sum_i w_i \cdot x_i < w_T \\ N/A & \text{if } \sum_i w_i \cdot x_i = w_T \end{cases} \quad (1)$$

where w_i is the weight of the ith input, x_i is the input applied to the ith input, and w_T is the threshold weight for the function f of a TLG. The input weights can be either positive or negative but the threshold weight is always positive. In this paper, an N-input function with P positive weights is denoted as $\{w_1, \ldots, w_P : w_T, w_{P+1}, \ldots, w_N\}$.

Example 1: Consider a function $f = x_1 + x_2 + x_3$ with weight configuration $(w_1, w_2, w_3 : w_T)$, where w_1, w_2, and w_3 correspond to the weights of the inputs x_1, x_2, and x_3, respectively, and w_T is the threshold weight. A possible weight configuration is $\{w_1, w_2, w_3 : w_T\} = \{4, 4, 4 : 3\}$, where all the input weights are positive. When applying the input pattern $\{x_1, x_2, x_3\} = \{0, 0, 1\}$, the weighted sum of inputs is $4 \cdot 0 + 4 \cdot 0 + 4 \cdot 1 > 3$, and, according to (1), $f = 1$. See also Fig. 1. Function f is denoted as $\{4, 4, 4 : 3\}$. □

This paper considers implementations of threshold logic functions using current mode. This is a popular CMOS-based approach. All current mode implementation methods considered in this paper consist of two parts: the differential part and the sensor part. The number of transistors in the sensor part is constant and does not depend on the implemented function. The number of transistors in the differential part depends on the sum of input weights and the threshold weight.

There exist two approaches for implementing current mode TLGs: the current mode TLG (CMTLG) [1] and the Differential current mode logic (DCML) [11]. Section II reviews these two approaches.

Section III presents a new implementation, which we call the dual clock current mode logic (DCCML), which results in both speed and switching energy [power-delay product (PDP)] improvements over the approaches in [1] and [11]. They consist of two parts: the differential part and the sensor part. All the pMOS transistors in the sensor part have the same size S, which we call the sensor size. The sensor size impacts the performance of all the three current mode implementations for any threshold logic function. It is a very time-consuming task to obtain the optimum sensor size through iterative SPICE simulations, one simulation for a different sensor size.

Section IV presents the second contribution of this paper, which is an analytical approach to determine quickly and

Manuscript received January 17, 2016; revised April 19, 2016 and June 28, 2016; accepted August 4, 2016. This work was supported by the NSF I/UCRC for Embedded Systems at SIUC under Grant NSF IIP 1230757, Grant NSF IIP 1432026, and Grant NSF IIP 1361847.

C. B. Dara is with Broadcom Ltd., Sunnyvale, CA 94086 USA (e-mail: dchandra@siu.edu).

T. Haniotakis and S. Tragoudas are with the Electrical and Computer Engineering Department, Southern Illinois University Carbondale, Carbondale, IL 62901 USA (e-mail: haniotak@siu.edu; spyros@engr.siu.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2016.2608953

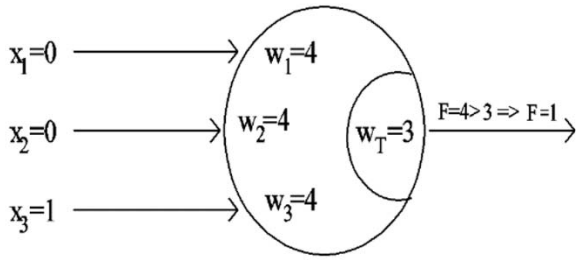

Fig. 1. Functionality of a TLG for a given weight configuration and input pattern.

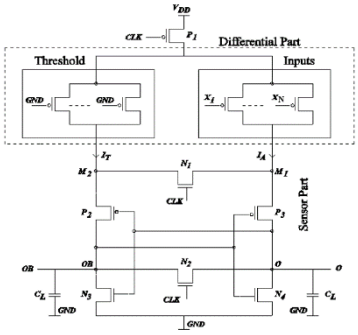

Fig. 2. Current mode TLG [1].

accurately the appropriate sensor size S for a given function under any existing current mode approach, such as those in [1] and [11] and the proposed implementation in Section III. Section V presents simulation results that demonstrate the accuracy of the optimum sensor identification method in Section IV. It also presents results that show that the current mode approach in Section III consistently outperforms those in [1] and [11] on delay as well as switching energy. Finally, Section VI concludes.

II. CMTLG and DCML Implementations of a Threshold Logic Function

This section presents two current mode designs in [1] (called CMTLG) and in [11] (called DCML). The block diagram of the CMTLG is shown in Fig. 2. It consists of the differential part and the sensor part. The differential part is subdivided into two parts: the threshold part and the (positive) inputs part. The inputs part has pMOS transistors that implement the positive input weights. The threshold part has pMOS transistors that implement the threshold weight and the negative input weights. Typically, a weight of value x is implemented by connecting x minimum size pMOS transistors in parallel. (Alternatively, it can be implemented by a single pMOS transistor whose width is x times the minimum size.) In both the parts, all the pMOS transistors are connected in parallel. The total current flowing through the threshold part is denoted by I_T. The total current passing through the inputs part is denoted by I_A. For each applied input pattern, pMOS active (ON) transistors correspond to input weights for inputs that are assigned a logic value 1. The pMOS transistors that implement the threshold weight are always active (ON).

The nodes connecting the differential part and the sensor part on the input side and the threshold side are M_1 and M_2,

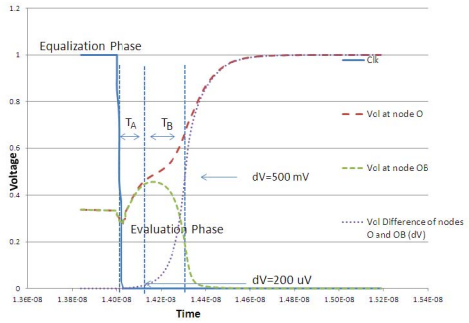

Fig. 3. Output voltages and their difference in the two clock phases for CMTLG.

respectively. The sensor part has three pMOS transistors P_1, P_2, P_3, and four nMOS transistors N_1, N_2, N_3, and N_4 as shown in Fig. 2. If the size of the sensor is S, then all the pMOS transistors in the sensor part have S μm size and all the nMOS transistors in the sensor part have a size smaller than S μm.

The operation of the CMTLG is divided into two phases [1]: the equalization phase and the evaluation phase. These phases are explained with the help of Figs. 2 and 3. When the applied clock (clk) to the CMTLG is high, then the circuit is in the equalization phase. When clk is low, then the circuit is in the evaluation phase [1]. In the equalization phase, transistors N_1 and N_2 are ON, nodes M_1 and M_2 have the same voltage because of transistor N_1, and nodes O and OB have the same voltage because of transistor N_2 (see also Fig. 2). In the evaluation phase, transistors N_1 and N_2 are OFF, and if the threshold current is less than the active current, then the voltage at node O rises faster than that at node OB [1]. If during the evaluation phase the threshold current exceeds the active current, then the voltage at node OB rises faster than that at node O [1].

Fig. 3 shows the two phases of clock, the voltage at the output nodes O and OB, and the voltage difference between the output nodes O and OB (dV). The delay of a CMTLG can be divided into two phases: the activation time and the boosting time. The first phase is the time taken by CMTLG to develop a small voltage difference (200 μV) across the output nodes O and OB [1]. In this phase, the difference between I_A and I_T leads to a gradually increasing voltage difference between the nodes M_1 and M_2. The time taken by the CMTLG to develop an initial voltage difference between the nodes O and OB is called the activation time T_A. The activation time depends mainly on the differential part. The second phase is the time taken by the sensor part (the back-to-back connected inverters) to boost the initial voltage difference to a logic state at the output nodes. This time is referred to as the boosting time T_B. The boosting time depends mainly on the sensor part.

An alternative differential clock threshold logic implementation is presented in [11], and it is referred to as the differential current mode logic (DCML) approach. Its block diagram is shown in Fig. 4. It is also divided into the differential part and the sensor part. The currents through the threshold part and the inputs part are also denoted by I_T and I_A, respectively. The sensor part consists of four pMOS transistors,

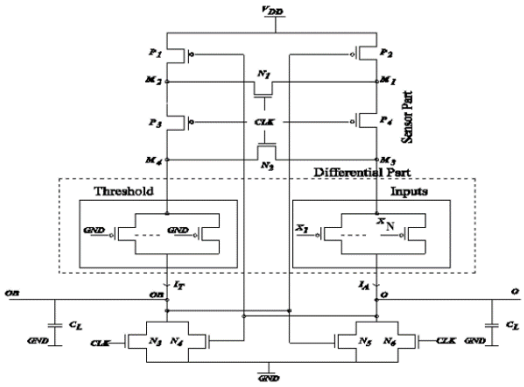

Fig. 4. Block diagram of differential current mode logic.

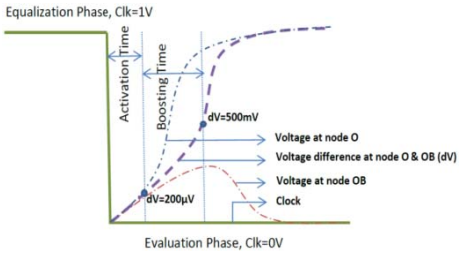

Fig. 5. Output voltages and their difference in the two clock phases for DCML.

labeled P_1–P_4, and six nMOS transistors, labeled N_1–N_6. The load capacitance C_L is applied to both the output nodes O and OB.

The applied clock is divided into two phases: when the clock is high the TLG is in the equalization phase and when it is low it operates on the evaluation phase. In the equalization phase, nMOS transistors N_1, N_2, N_3, and N_6 are active. Transistor N_1 equalizes the voltage at nodes M_1 and M_2. Similarly, transistor N_2 equalizes the voltage at nodes M_3 and M_4. In the equalization phase, transistors N_6 and N_3 are active and there exists a discharge path for nodes O and OB of Fig. 4. If there is a voltage difference at nodes O and OB, during the evaluation phase, then the sensor part will identify the voltage difference and it will boost the voltage at the output nodes O and OB to a desired voltage. When the active current I_A is greater than the threshold current I_T, then the voltage at the output node O rises faster than the voltage at node OB. As a result, high voltage is obtained at node O and low voltage is obtained at node OB. When I_T is greater than I_A, then the voltage at OB rises faster than the voltage at O and low voltage results at OB.

Fig. 5 shows the two phases of the clock, the voltage at nodes O and OB, and the voltage difference between O and OB (dV). The delay of DCML is divided into the activation time T_A and the boosting time T_B.

III. Low Power and High-Speed Dual-Clock-Based Current Mode TL Implementation

A new TLG implementation is proposed. It is called DCCML. As the name indicates, two clocks are used to achieve low power consumption and high speed.

The block diagram DCCML is shown in Fig. 6. As in previous approaches, the DCCML is divided into two basic blocks: the differential block and the sensor block. The differential block is further divided into four blocks: the positive threshold, the negative inputs, the negative threshold, and the positive inputs. All the transistors in the differential block are equal-sized pMOS transistors and are connected in parallel, as shown in Fig. 6. The sensor block consists of six pMOS transistors $P_1 \cdots P_6$ and three nMOS transistors N_1, N_2, and N_3. The gates of transistors P_1 and N_1 are connected to Clk_1 and the gates of transistors P_2, P_5, and P_6 are connected to Clk_2. Transistor N_1 acts as an equalizing transistor and it equalizes the voltage at nodes OP and OPB. Transistors P_5 and P_6 isolate the differential block from the sensor block.

The transistors in the positive threshold and negative threshold are always active. Transistors in the positive and negative inputs blocks are active depending upon the input pattern applied. The input pattern applied for the positive inputs block is denoted by $\{x_1, x_2, \ldots, x_I$. Let N denote the number of inputs, and I denote the number of positive inputs. Then the number of negative inputs is N–I. The input pattern applied for the negative inputs block is denoted by $\{x_{I+1}, x_{I+2}, \ldots, x_N$.

Consider a function *f,* with a possible weight configuration $\{w_1, w_2 : w_T, w_3, w_4\}$={2, 2:3, −1, −1}. In the given weight configuration, we have two positive weights $w_1$andw_2 and two negative weights w_3 and w_4. Weights w_1 and w_2 are implemented in the positive inputs section and weights w_3 and w_4 are implemented in the negative inputs section. The threshold weight w_T is implemented in the positive threshold section.

The current through the four blocks (positive threshold, negative inputs, negative threshold, and positive inputs) are denoted by I_{PT}, I_{NI}, I_{NT}, and I_{PI}, respectively. The currents through transistors P_5 and P_6 are denoted by I_P^5 and I_P^6. Here, $I_P^5 = I_{PT} + I_{NI}$ and $I_P^6 = I_{NT} + I_{PI}$. Nodes OP and OPB are the output nodes. The load capacitance is denoted by C_L.

The operation is divided into three phases: the equalization phase, the pre-evaluation phase, and the final-evaluation phase. When clocks Clk_1 and Clk_2 are high, then the circuit is in the equalization phase. When clocks Clk_1 and Clk_2 are low, then the circuit is in the pre-evaluation phase. When Clk_1 is low and Clk_2 is high, then the circuit is in the final-evaluation phase. See also Fig. 7.

It is noted that when the two clocks are not completely aligned the operation of the gate is not effected. The possible cases of misalignment are: 1) the falling edge of Clk_2 comes before the falling edge of Clk_1 and 2) the falling edge of Clk_2 comes after the falling edge of Clk_1. In the first case, the current from the differential part is equalized because of transistor N_1 and the evaluation phase starts after the falling edge of Clk_1. In the second case, there will be no current from the differential part as Clk_2 is not active yet. Hence, the pre-evaluation phase starts after the falling edge of Clk_2. The implementation avoids a very early arrival of Clk_1. In that case, a nonstable signal might result in erroneous output.

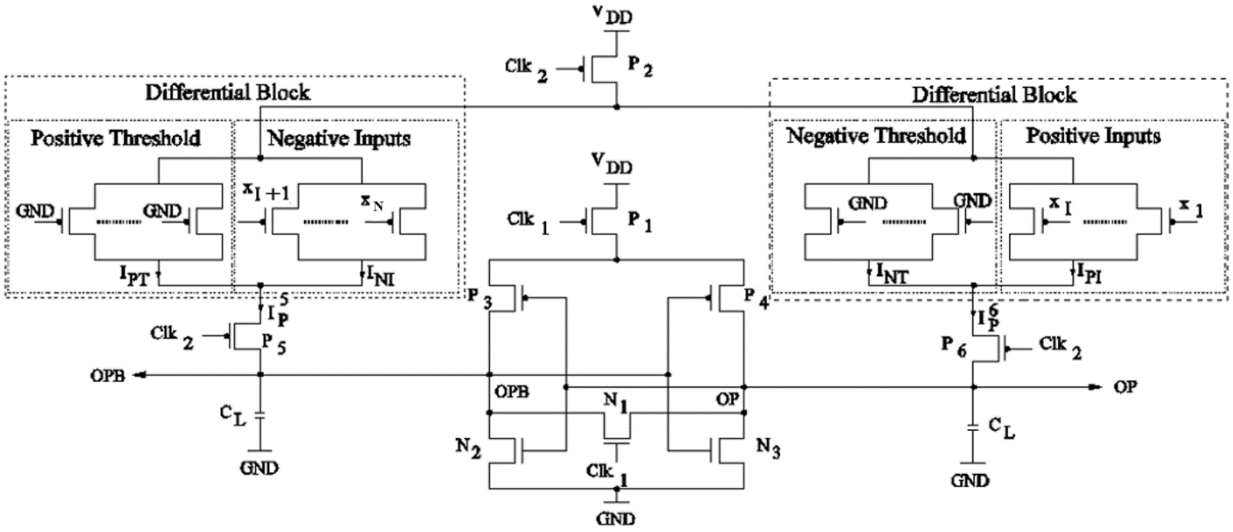

Fig. 6. Block diagram of DCCML TLG.

If the current I_P^6 through the pMOS transistor P_6 is greater than the current I_P^5 through the pMOS transistor P_5, then the voltage at the output node OP rises faster than the output node OPB. As a result, high voltage is obtained at output node OP and low voltage occurs at output node OPB. Otherwise, the voltage at the output node OPB rises faster than the output node OP. As a result, high voltage is obtained at the output node OPB and low voltage is obtained at node OP.

In DCCML, the pMOS transistors P_1, P_2, P_5, P_6 and the pMOS transistors in the differential block are used to provide the initial voltage at the output nodes OP and OPB. Using Clk_2, we restrict the current flow from the differential block to the sensor block, once initial voltage difference is established at the nodes OP and OPB; in this way we stop the current flowing from the differential block to the sensor block. Using Clk_2, we are able to minimize power consumption in the circuit. Transistors P_5 and P_6 are also used to isolate high capacitance circuit block (the differential block) at the output nodes. Hence, in the final evaluation phase the sensor block drives the load capacitance as well as the capacitance from a single transistor P_5 or P_6. Delay is reduced because the duration of the final evaluation phase is small. The voltage at the output nodes OP and OPB and the voltage difference (dV) at the output nodes OP and OPB are shown in Fig. 8 for the three clock phases.

In particular, the delay of the DCCML is divided into two time phases: the activation time and the boosting time. The activation time is the time taken by the circuit to develop an initial voltage difference at the output nodes OP and OPB. The boosting time is the time taken by the DCCML to bring the initial voltage to the correct voltage at the output nodes OP and OPB.

In the pre-evaluation phase, both the differential part and the sensor part are active, and therefore the activation time is not affected. In the final evaluation phase, the differential part is kept inactive using *Clk2*. Therefore, the effect of internal capacitance due to the differential part is isolated. Hence, it takes very little time to boost the outputs to the final value. The power is also reduced due to the isolation of the differential part.

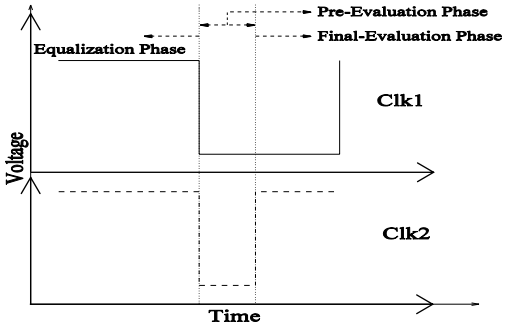

Fig. 7. Clocks in DCCML.

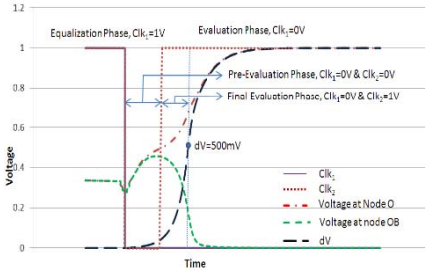

Fig. 8. Voltage at output nodes *OP* and *OPB* and *dV* during the three clock phases.

IV. Delay Minimization by an Appropriate Sensor Size Selection

This section presents an analytical formula to compute the sensor size that minimizes the gate delay. Let N denote the number of inputs, N the sum of all positive input weights, and T the sum of the threshold weight and negative input weights.

Our analysis assumes that all the input weights are connected in parallel, and that each weight w_i can be implemented by w_i unit width pMOS transistors connected in parallel. This is an accurate assumption. We have implemented TLG weights using a smaller number of wider pMOS transistors connected in parallel and SPICE simulations showed no difference in the performance of the TLG. This is further explained in the example below.

Example 2: Consider a threshold function where N, the sum of positive input weights, is 11. Let also T, the sum of the threshold weight and negative input weights, be 4. In this function, we have $(N, T) = (11, 4)$. Gates {11:4}, {6, 5: 4}, {5, 5, 1: 4}, {5, 4, 1, 1: 4}, {4, 4, 1, 1, 1: 4}, and {1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1: 4} were implemented in the 45-nm technology. SPICE simulation shows an identical delay of 297 ps. □

In the following, we will not differentiate among functions for which the sum of all positive input weights is N, and the sum of the negative input weight and threshold weight is T. Since all these threshold functions exhibit the same delay, these functions will be denoted by the pair (N, T). The remaining focus is on how to determine the optimum sensor that minimizes the delay of any (N, T) function.

The proposed method considers that the TLG operates under an input pattern that exhibits the worst case propagation delay, and then focuses on deriving an analytical model that expresses TLG delay in terms of the sensor size S in that setting. In a first step, we identify the pattern that gives the highest delay for the function. In a second step, we consider this worst case scenario, and the delay will be expressed as a function of the sensor size S. Then, we operate on that function in order to optimize the sensor size S.

In the first step, it is shown that when $T+1$ inputs are active then the TLG exhibits its worst delay. Let $N_A = \sum_i w_i$, such that $x_i = 1$. Such inputs i are called active, and the respective pMOS transistors are also called active. Assume that the initial current flowing through an active minimum-sized pMOS is I_p. Then the current flowing through the threshold side of the TLG is $T \cdot I_p$, and the current flowing through the input side for N_A inputs being active is $N_A \cdot I_p$. To obtain the worst case delay for logic 1 at the output node O, the current difference $I_A - I_T$ should be minimum.

For logic 0, this current difference should also be minimum. Since transistors on the threshold side are always ON, the maximum delay for a rising transition of the output is obtained when we have $T+1$ active transistors. Likewise, $T-1$ active transistors tend to obtain the worst case delay for a falling transition at the output. However, it is known that the worst case delay occurs for rising output transition [1]. Hence, a worst case delay pattern is one that gives the least current difference at nodes M_1 and M_2. The following is an example where SPICE simulations confirm this analysis.

Example 3: Consider a CMTLG implementation of a function with $T = 4$, $N = 11$, and sensor size $S = 10$. The input pattern that has $T+1$ number of active inputs gives the worst delay. Hence, the highest delay encountered is $N_A = 5$. Fig. 9 shows the delay of the TLG using SPICE in 45-nm technology. When N_A varies in range [5], [11], the output transition

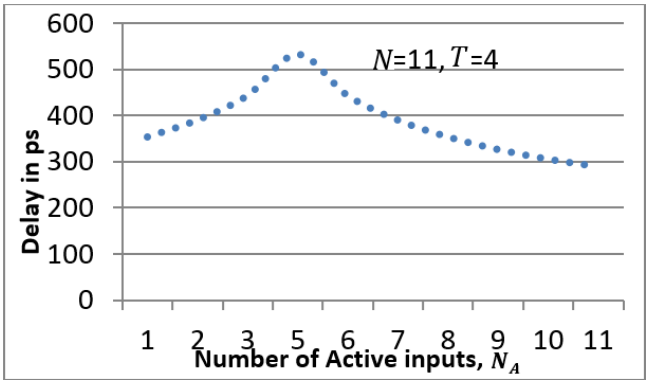

Fig. 9. CMTLG delay with $\mathbf{N = 11}$ and $\mathbf{T = 4}$ as $\mathbf{N_A}$ varies.

is rising, and the highest rising delay occurs when $N_A = 5$. When N_A is in the range (0, 5) the transition at the output is falling and in that case the delay is less. □

Similar behavior has been observed for different values of T and N. Furthermore, extensive SPICE simulations have confirmed that the worst case delay of DCML gates is obtained when $N_A = T + 1$ and also occurs when the output is rising.

In the second step of the proposed method, it is shown how to obtain an analytical expression that approximates the time delay T_D as a function of the sensor size S, given N and T. The delay time T_D is divided into two phases: the activation time T_A and the boosting time T_B. The tradeoff among the two phases is analyzed by varying the sensor size S and keeping all the other parameters N, T, and N_A constant.

During the activation time, the major current component is the current from the differential part. From the schematics in Figs. 2 and 4, due to the voltage difference at nodes O and OB, we conclude that $|I_A - I_T|$ is proportional to $|N_A - T|$. The time requirement for the activation time will be inversely proportional to the current. The time will be proportional to a charge that depends on two components: the voltage difference that is required at the end of the activation phase and the capacitance that the differential part is driving. This capacitance is the difference in the differential capacitance $N \cdot C_i^{'} - T \cdot C_T^{'}$, where $C_i^{'}$ and $C_T^{'}$ are the unit capacitances of the input part and the threshold part, the sensor capacitance, which is $S \cdot C_S^{'}$ where $C_S^{'}$ is unit capacitances of the sensor part, and the output capacitance. The overall time required for activation will be proportional to $(N \cdot C_i^{'} - T \cdot C_T^{'} + S \cdot C_S^{'} + C_L/|N_A - T|)$. Term $(N \cdot C_i^{'} - T \cdot C_T^{'} + C_L/|N_A - T|)$ is invariant to the sensor size and term $(S \cdot C_S^{'}/|N_A - T|)$ is proportional to S.

During the boosting time, the delay depends on the current provided by the sensor. This current will be proportional to the sensor size. The capacitance to be charged will be the same as in the activation time. (The voltage will be different, which does not depend on the sensor size S.) Hence, the boosting time will be proportional to $(N \cdot C_i^{'} - T \cdot C_T^{'} + S \cdot C_S^{'} + C_L/S)$. The numerator is an approximation to the overall capacitance connected to the outputs O and OB. The boosting time consists of $(S \cdot C_S^{'}/S) = C_S^{'}$, which is invariable to the size of the sensor and $(N \cdot C_i^{'} - T \cdot C_T^{'} + C_L/S)$, which is inversely proportional to the sensor size.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

We conclude that the gate delay consists of three components T_0, T_1, and T_2 defined below. Component T_0 is invariant to S and that is the sum of the invariant components of the activation time and the invariable components of the boosting time, i.e., $T_0 = C_S' + (N \cdot C_i' - T \cdot C_T' + C_L/|N_A - T|)$. Component T_1 is proportional to the sensor size S and occurs during the activation time, i.e., $T_1 = C_S' \cdot |(1/N_A - T)|$. Finally, component T_2, which is inversely proportional to the sensor size, occurs during the boosting time and is equal to $N \cdot C_i' - T \cdot C_T' + C_L$. Concluding, the overall time T_D is estimated as

$$T_D = T_0 + T_1 \cdot S + T_2 \cdot \frac{1}{S}$$

when

$$T_0 = a \cdot (C_S' + \frac{b \cdot (N \cdot C_i' - T \cdot C_T') + C_L}{|N_A - T| \cdot c})$$
$$T_1 = d \cdot (C_S' \cdot \left|\frac{1}{N_A - T}\right| \cdot \frac{1}{c})$$
$$T_2 = d \cdot (b \cdot (N \cdot C_i' - T \cdot C_T') + C_L).$$

By applying regression analysis on SPICE simulations, T_D is rewritten as

$$T_D = T_0 + \varepsilon_0 + T_1 \cdot S + \varepsilon_1 + T_2 \cdot \frac{1}{S} + \varepsilon_2 \tag{2}$$

with $\varepsilon_0 \in (0, 5)$, $\varepsilon_1 \in (0, 20)$, and $\varepsilon_2 \in (0, 5)$. All the ranges are expressed in picoseconds. Here *a, b, c,* and *d* are constants and their values are $a = 1\text{e} - 9$, $b = 1$ for CMTLG, DCML and $b = 0.1$ for DCCML, $c = 1\text{e} - 11$, and $d = 3.86\text{e} - 2$. Equation (3) gives the gate delay for different sensor sizes for fixed values of N, T, N_A, and C_L.

The final step of the proposed method operates on (3) in order to derive sensor size S_{opt}, which gives the minimum gate delay. Sensor size S_{opt} is derived by applying the first derivative on (3) and equating it to zero in order to find the minimum value of T_D. We have that

$$S_{\text{opt}} = \sqrt{\left(\frac{(b \cdot (N \cdot C_i' - T \cdot C_T') + C_L)}{C_S' \cdot \left|\frac{1}{N_A - T}\right| \cdot \frac{1}{c}}\right)}. \tag{3}$$

The remainder of this section presents the corollaries obtained by (3).

Corollary 1: The delay T_D decreases with an increase in S, reaches an optimum value for some consecutive values of S, and then increases as S increases.

The actual values of minimum S depend on N and T.

Corollary 2: For a sensor size that is smaller than the optimum sensor size S_{opt}, the activation time T_A is low and the boosting time T_B is high.

The activation time is less because it has less capacitance and the output can drive this small capacitance faster to develop an initial voltage difference. In order to boost the initial voltage difference, the back-to-back connected inverters must be small. Hence, the boosting time is high.

Corollary 3: For a sensor size that is larger than the optimum sensor size S_{opt}, the activation time T_A is high and the boosting time T_B is low.

TABLE I

FUNCTIONS THAT CORRESPOND TO DIFFERENT VALUES OF N AND T PAIRS

N,T	**Input Weights**
8,5	{4,2,2:5}
12,9	{4,4,2,2:9}
10,4	{5,3,2:4}
9,7	{3,3,3:7}
32,4	{3,3,5,5,5,5,6:4}
12,7	{6,2,2,2:7}
18,9	{8,5,5:6,3}
22,6	{4,3,7,8:6}
12,4	{3,3,6:4}
16,15	{2,2,2,2,2,2,2,2:15}
47,6	{3,2,2,8,8,8,8,8:6}
66,7	{2,2,2,2,8,8,8,8,8,9,9:7}
80,1	{2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2, 2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2:1}
42,3	{2,2,4,4,4,4,4,4,4,5,5:3}
77,5	{2,2,2,6,6,6,6,6,7,7,7,7,7:5}

The activation time is high because it may have a large capacitance and the output is slow to develop an initial voltage difference. Large back-to-back connected inverters will boost the initial voltage difference quickly.

Corollary 4: T_D decreases as S approaches S_{opt} and then increases as S grows larger than S_{opt}.

The corollary is justified because the total delay T_D of TLG is the sum of T_A and T_B.

V. SIMULATION RESULTS

This section presents the simulation results that demonstrate the impact of the sensor size optimization method in Section IV and the DCCML approach in Section III. The evaluation in this section was conducted on randomly selected threshold logic functions described by the respective (*N, T*) values, where N denotes the sum of the positive input weights and T denotes the sum of the threshold weight and negative input weights. Table I lists some of the functions. The first column in Table I provides the N and T values. The second column in Table I lists the positive input weights, followed by ":", followed by the threshold weight, and then by the negative input weights. As an example, function (18, 9) in the seventh row has three positive input weights and a negative input weight.

All the functions were implemented in 45-nm CMOS technology with DCCML as well as the approaches in [1] and [11]. The clock voltages were set to 1 V for high voltage and 0 V for low voltage, and the load capacitance C_L was 30fF. The channel length was set to the minimum value for all transistors. All the input pMOS transistors and threshold pMOS transistors had a240 − nm width size, which is the minimum transistor width. Weights of value higher than one were implemented by connecting the unit-sized pMOS transistors in parallel.

The first part of this section presents the detailed simulations that determine the impact of the proposed analytical method

TABLE II

S_{opt} AND MINIMUM DELAY (IN PS) WITH ITERATIVE SPICE, THE ESTIMATE OF [1], AND THE PROPOSED APPROACH IN SECTION IV. ALL (N, T) FUNCTIONS HAVE BEEN IMPLEMENTED USING THE CMTLG APPROACH IN [1]

N,T values	Iterative SPICE		Estimate in [1]		Proposed (Section IV)			
	S_{opt}	Delay (ps)	S_{opt}	Delay (ps)	S_{opt}	Delay with calculated S_{opt}	% delay overhead compared to iterative SPICE	% improvement in delay over the estimate in [1]
8,5	15.0	364.2	4	510.1	15.4	364.8	0.16	39.83
12,9	16.7	365.5	6	-	15.4	366.4	0.24	-
10,4	16.1	366.8	5	479.3	16.3	366.9	0.02	30.63
9,7	16.0	362.7	4.5	437.9	15.2	368.1	1.48	18.96
32,4	18.5	396	16	401.3	21.3	397.9	0.47	0.85
12,7	16.5	365.7	6	393.1	16.0	366.2	0.13	7.34
18,9	17.2	370.3	9	391.6	17.0	372.1	0.48	5.24
22,6	17.7	381.1	11	476.8	18.7	382.3	0.31	24.71
12,4	16.2	336.2	6	521.8	16.8	337.6	0.41	54.56
16,15	17.0	381.3	8	503.6	14.9	383.7	0.62	31.24
47,6	22.5	418.8	28.5	433.5	23.7	419.7	0.21	3.28
66,7	27.5	442.2	33	455.3	26.8	442.6	0.09	2.86
80,1	32.0	471.2	40	490.1	29.8	475.1	0.82	3.15
42,3	20.1	411	21	412.8	23.4	412.6	0.38	0.04
77,5	28.7	465.6	38.5	490.5	28.8	465.7	0.02	5.32

in Section IV, which calculates the optimum sensor size S_{opt} using the N and T values and then a SPICE simulation determines the delay for the S_{opt} value. Our simulation results considered all the (N, T) functions in Table I, and each function was implemented with the CMTLG method in [1], the DCML method in [11], and the proposed approach in Section III.

In order to determine the impact of our proposed method in Section IV, we implemented a brute-force time-consuming approach to calculate S_{opt}. Given an (N, T) function, this approach selects 100 different sensor sizes within a range of values that we speculate that the optimum size exists, calculates the gate delay by a SPICE simulation, and eventually returns the sensor size S_{opt} that results in the minimum gate delay. We call this approach the iterative SPICE method. We note that iterative SPICE is a very time-consuming method due, in part, to the large number of steps that we employ at each simulation: its execution time is 100 times slower than the proposed method in Section IV, as the time required for each SPICE simulation is approximately 1 s, and this method requires approximately 2 min to execute per function. In contrast, our method in Section IV executes in approximately 1 s per function, which is practically the execution time for a SPICE simulation.

Tables II–IV present the detailed results that demonstrate the superiority of the S_{opt} identification method in Section IV for all the three implementation methods. In particular, Table II presents the results when each (N, T) function is implemented using the CMTLG approach in [1], Table III presents the results when each function is implemented using the CMTLG approach in [11], and Table IV when the functions were implemented using the approach in Section IV.

Table II focuses on the CMTLG implementation [1]. For this implementation, we also compared the impact of the proposed optimum size identification method in Section IV against another approach. Namely, Bobba and Hajj [1]

TABLE III

S_{opt} AND MINIMUM DELAY (IN PS) WITH ITERATIVE SPICE AND THE PROPOSED APPROACH IN SECTION IV. ALL (N, T) FUNCTIONS HAVE BEEN IMPLEMENTED USING THE DCML APPROACH IN [1]

N,T values	Iterative SPICE		Proposed (Section IV)		
	S_{opt}	Delay (ps)	S_{opt}	Delay with calculated S_{opt}	% delay overhead compared to iterative SPICE
8,5	10.5	269.3	11.5	270.1	0.29
12,9	12.5	271.1	11.5	272.3	0.44
10,4	11.0	273.0	12.1	273.5	0.18
9,7	11.7	265.6	11.3	265.9	0.11
32,4	14.0	296.0	15.9	298.1	0.70
12,7	12.0	272.0	12.1	272.1	0.03
18,9	13.3	275.6	12.7	276.8	0.43
22,6	13.5	383.4	14.0	384.1	0.18
12,4	11.5	275.6	12.5	275.9	0.10
16,15	13.0	272.0	11.1	274.0	0.73
47,6	16.2	314.8	17.7	316.8	0.63
66,7	18.7	333.1	20.0	335.9	0.84
80,1	20.0	357.6	22.3	359.2	0.44
42,3	15.0	308.7	17.5	311.6	0.93
77,5	19.5	351.3	21.5	353.9	0.74

TABLE IV

S_{opt} AND MINIMUM DELAY (IN PS) WITH ITERATIVE SPICE AND THE PROPOSED APPROACH IN SECTION IV. ALL (N, T) FUNCTIONS HAVE BEEN IMPLEMENTED USING THE APPROACH IN SECTION III

N,T	Iterative SPICE		Proposed (Section IV)		
	S_{opt}	Delay (ps)	S_{opt}	Delay with calculated S_{opt}	% delay overhead compared to exhaustive SPICE
8,5	21.2	192.7	22.4	192.8	0.05
12,9	22.2	193.1	22.4	193.2	0.02
10,4	22.0	194.6	22.6	194.7	0.05
9,7	22.1	189.8	22.4	190.1	0.15
32,4	22.7	207.8	23.5	208.3	0.24
12,7	22.0	193.7	22.5	193.9	0.07
18,9	22.4	196.8	22.7	196.9	0.04
22,6	22.5	201.2	23.0	202.3	0.53
12,4	22.0	195.7	22.7	195.8	0.05
16,15	22.3	193.1	22.4	193.7	0.31
47,6	23.2	216.6	24.1	216.9	0.12
66,7	23.7	229.6	24.8	231.5	0.82
80,1	25.0	246.9	25.6	248.1	0.48
42,3	23.0	212.3	24.0	213.6	0.61
77,5	23.9	244.2	25.3	247.1	1.15

estimated the optimum sensor size for a given value of N in a CMTLG implementation to be approximately $(N/2)\mu$m. Notice that this method did not take into consideration the value of T. More significantly, the presented results show that this approximation sometimes fails to implement the function correctly. The following outline the results listed in Table II. Column 1 lists the (N, T) functions of Table I. Column 2 lists the minimum sensor size that was calculated for each function using the iterative SPICE approach, and the best observed gate delay (in ps) by iterative SPICE is listed in column 3. Column 4 lists the approximation of S_{opt} as listed in [1], and the corresponding gate delay (obtained by SPICE) is shown in column 5 of Table II. Column 6 gives the optimum sensor size using the proposed method in Section IV, and column 7 lists the gate delay (in ps) obtained by a SPICE simulation. For the second function where (N, T) = (12, 9), the approximation method in [1] resulted in an incorrect implementation, and this is denoted by "−" in Table II.

The results in columns 2 and 6 in Table II show that the sensor sizes by the proposed method and iterative SPICE are very close for all the listed functions implemented with CMTLG. In contrast, the sensor size estimate in column 4 is often very different. Columns 3 and 7 show that for each function, the gate delay by the time-consuming iterative SPICE is almost identical to that returned by the proposed method. In particular, column 8 shows that for each implemented function with CMTLG, the gate delay by our method was no more than 0.4% of the delay for the sensor size computed by the time-consuming iterative SPICE method. In contrast, column 9 of Table II shows that the gate delays by our method were much less (often over 25% more) than the delays where the sensor size was determined by [1].

Table III considers function implementations using the DCML method in [11]. In Table III, we compare the sensor identification method of Section IV with the time-consuming iterative SPICE method. Column 1 lists the (N, T) functions of Table I, column 2 lists the minimum sensor size that was calculated for each function using iterative SPICE, and column 3 the best observed gate delay (in ps) by iterative SPICE. Column 4 gives the optimum sensor size using the proposed method in Section IV, and column 5 lists the respective gate delay (in ps). As was the case for CMTLG implementations, the results in columns 2 and 4 show that the sensor size by the proposed method is very close to the best sensor sizes by iterative SPICE for all the listed functions implemented with DCML. For each function implemented with DCML, columns 3 and 5 show that the gate delay where the sensor size was determined by the time-consuming iterative SPICE is almost identical to the gate delay where the sensor size was determined by the proposed method in Section IV. More specifically, column 6 shows that the gate delay by our method was no more than 0.93% of the gate delay by iterative SPICE.

Table IV elaborates on function implementations using the proposed DCCML method in Section III, and we show the impact of the sensor identification method of Section IV compared with iterative SPICE. Column 1 lists again all the (N, T) functions in Table I. Column 2 presents the minimum sensor size for each function using iterative SPICE, and column 3 the best observed gate delay (in ps) by iterative SPICE. Column 4 gives the optimum sensor size obtained with the proposed method in Section IV, and column 5 lists the respective gate delay (in ps). As was the case for CMTLG and DCML implementations, the results in columns 2 and 4 show that the sensor sizes by the proposed method and iterative SPICE are close for all the DCCML implementations. Columns 3 and 5 show that when the sensor size was calculated by the time-consuming iterative SPICE, it is always very similar to the delay when the sensor size was quickly calculated by the proposed method in Section IV. In particular, column 6 shows that for all the functions the DCCML delay with the method in Section IV was never more than 1.15% of the gate delay by iterative SPICE.

TABLE V

GATE DELAY AND SWITCHING ENERGY WITH THE CMTLG, DCML, DCCML (PROPOSED), AND STANDARD CMOS IMPLEMENTATIONS FOR ALL (N, T) FUNCTIONS IN TABLE I

N,T	CMTLG		DCML		DCCML (proposed)		Standard CMOS	
	Delay (ps)	PDP (fJ)	Delay (ps)	PDP (fJ)	Delay (ps)	PDP (fJ)	Delay (ps)	PDP (fJ)
8,5	364.2	7.54	269.3	7.86	192.7	4.52	219.68	1.13
12,9	365.5	8.36	271.1	8.33	193.1	4.83	230.14	1.24
10,4	366.8	7.94	273.0	8.08	194.6	4.79	215.61	1.75
9,7	362.7	8.16	265.6	8.02	189.8	4.69	208.92	1.71
32,4	396.0	10.11	296.0	9.80	207.8	5.42	255.16	12.84
12,7	365.7	8.23	272.0	8.27	193.7	4.81	237.49	2.33
18,9	370.3	8.85	275.6	8.62	196.8	4.98	245.89	2.16
22,6	381.1	9.19	383.4	12.30	201.2	5.13	235.48	8.45
12,4	336.2	7.44	275.6	8.28	195.7	4.81	215.61	1.75
16,15	381.3	8.99	272.0	8.45	193.1	4.88	241.18	2.48
47,6	418.8	12.32	314.8	11.84	216.6	5.80	255.16	13.46
66,7	442.2	15.49	333.1	14.19	229.6	6.50	268.01	7.46
80,1	471.2	17.83	357.6	16.12	246.9	7.13	291.28	42.11
42,3	411.0	10.94	308.7	10.85	212.3	5.71	263.93	16.37
77,5	465.6	16.97	351.3	15.13	244.2	7.01	265.63	15.13

The results in Tables II–IV clearly indicate that independent of the current mode implementation method, the proposed analytical method in Section IV calculates a sensor size for which the gate has practically optimum delay. Therefore, (3) in Section IV should be used to quickly calculate the transistor size for any (N, T) function under any current mode implementation.

In the second part of this section, the current mode implementation in Section III is compared with the TLG current mode implementations in [1] and [11] as well as the traditional CMOS designs with respect to gate delay (in ps) as well as PDP, also referred to as switching energy (in fJ). The functions in Table I were considered and the load capacitance considered for all these functions was set to 30 fF.

In all the CMTLG methods the sensor size for each function was computed using (3). Optimized non-TLG CMOS implementations were obtained by Synopsys Design Compiler using two-NAND, three-NAND, and inverter gates. Similar to the designed TLGs, minimum sized transistors are considered as 240 nm.

Table V summarizes this comparative study for the 15 different threshold logic functions in Table I. Table V is divided into nine columns. Column 1 lists the N and T values of the functions. For each function, columns 2 and 3 show the gate delay and PDP for the CMTLG implementation [1]. Columns 4 and 5 show the gate delay and the PDP for the DCML implementation [11]. Columns 6 and 7 show the gate delay and PDP for the proposed DCCML implementation in Section III. Finally, columns 8 and 9 show the gate delay and PDP obtained using standard CMOS implementation.

Observe that the proposed DCCML implementation outperformed DCML and CMTLG for all the functions, and for some functions the DCCML delay and switching energy were drastically reduced. As an example, for function (22, 6) both the delay and switching energy were almost half of those obtained by the other two implementations. In addition, columns 6 and 8 show that the DCCML delay was always less than in standard CMOS implementations. The results also show that the PDP is significantly less for the larger functions that have more than five inputs. It is important to observe that the benefits in both the delay and PDP increase as the functions become more complex. As an example, for function $(N, T) = (80, 1)$ the delay of our TL design is reduced by 45 ps, whereas the PDP of the CMOS implementation is six times more than the proposed. The results in Table V clearly indicate that the proposed DCCML method can be used to design high-speed and switching energy efficient functions.

VI. Conclusion

An analytical method has been proposed to identify quickly the transistor size in the sensor component of a current mode implementation that ensures very low gate delay (very close to the minimum), independent of the current mode method used to implement the threshold logic function. A new current mode implementation method was also proposed that outperforms existing implementations both in gate delay as well as energy.

Acknowledgment

Part of this work was accomplished while at the Electrical and Computer Engineering Department, Southern Illinois University Carbondale.

References

[1] S. Bobba and I. N. Hajj, "Current-mode threshold logic gates," in *Proc. IEEE ICCD*, Sep. 2000, pp. 235–240.

[2] T. Ogawa, T. Hirose, T. Asai, and Y. Amemiya, "Threshold-logic devices consisting of subthreshold CMOS circuits," *IEICE Trans. Fundam. Electron., Commun. Comput. Sci.*, vol. E92-A, no. 2, pp. 436–442, 2009.

[3] S. Muroga, *Threshold Logic and Its Applications*. New York, NY, USA: Wiley, 1971.

[4] W. Prost *et al.*, "Manufacturability and robust design of nanoelectronic logic circuits based on resonant tunnelling diodes," *Int. J. Circuit Theory Appl.*, vol. 28, no. 6, pp. 537–552, Nov./Dec. 2000.

[5] S. Leshner, K. Berezowski, X. Yao, G. Chalivendra, S. Patel, and S. Vrudhula, "A low power, high performance threshold logic-based standard cell multiplier in 65 nm CMOS," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI*, Lixouri, Greece, Jul. 2010, pp. 210–215.

[6] M. Sharad, D. Fan, and K. Roy. (2013). "Ultra-low energy, high-performance dynamic resistive threshold logic." [Online]. Available: http://arxiv.org/abs/1308.4672

[7] P. Celinski, J. F. López, S. Al-Sarawi, and D. Abbott, "Low power, high speed, charge recycling CMOS threshold logic gate," *Electron. Lett.*, vol. 37, no. 17, pp. 1067–1069, Aug. 2001.

[8] S. Leshner and S. Vrudhula, "Threshold logic element having low leakage power and high performance," WO Patent 2009 102 948, Aug. 20, 2009.

[9] T. Shibata and T. Ohmi, "A functional MOS transistor featuring gate-level weighted sum and threshold operations," *IEEE Trans. Electron Devices*, vol. 39, no. 6, pp. 1444–1455, Jun. 1992.

[10] V. Beiu, J. M. Quintana, and M. J. Avedillo, "VLSI implementations of threshold logic—A comprehensive survey," *IEEE Trans. Neural Netw.*, vol. 14, no. 5, pp. 1217–1243, Sep. 2003.

[11] T. Gowda, S. Leshner, S. Vrudhula, and S. Kim, "Threshold logic gene regulatory networks," in *Proc. IEEE Int. Workshop GENSIPS*, Jun. 2007, pp. 1–4.

[12] A. K. Palaniswamy and S. Tragoudas, "A scalable threshold logic synthesis method using ZBDDs," in *Proc. 22nd Great Lakes Symp. VLSI*, 2012, pp. 307–310.

[13] C. B. Dara, T. Haniotakis, and S. Tragoudas, "Delay analysis for an N-input current mode threshold logic gate," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI)*, Aug. 2012, pp. 344–349.

[14] A. K. Palaniswamy, T. Haniotakis, and S. Tragoudas, "ATPG for delay defects in current mode threshold logic circuits," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. PP, no. 99, pp. 1–1.

[15] A. Neutzling, J. M. Matos, A. Mishchenko, R. Ribas, and A. I. Reis, "Threshold logic synthesis based on cut pruning," in *Proc. ICCAD*, Nov. 2015, pp. 494–499.

Chandra Babu Dara received the M.S. and Ph.D. degrees in electrical and computer engineering from Southern Illinois University Carbondale, Carbondale, IL, USA, in 2009 and 2015, respectively. His Ph.D. dissertation is in the area of design of high performance threshold logic gates.

He is currently a Staff Engineer with Broadcom Limited, San Jose, CA, USA. His current research interests include low power very large scale integration design, threshold logic gates, memristor based designs, and current mode TLGs.

Mr. Dara has been a Reviewer for the IEEE journals and conferences and a member of various program committees.

Themistoklis Haniotakis (M'98) received the B.S. degree in physics and the Ph.D. degree in informatics from the University of Athens, Athens, Greece. His Ph.D. thesis is in the area of self checking circuits.

He was a Faculty Member with the University of Patras, Patras, Greece, and is currently a Faculty Member with the Department of Electrical and Computer Engineering, Southern Illinois University at Carbondale, Carbondale, IL, USA. He has authored 20 journal and over 50 conference publications. His current research interests include very large scale integration (VLSI) design, fault-tolerant computing, VLSI testing and design for testability, and RF IC design and test.

Mr. Haniotakis was a recipient of the Best Paper Award (ISQED). He has been a Reviewer for the IEEE journals and conferences and a member of various program committees.

Spyros Tragoudas received the B.Sc. degree from the University of Patras, Patras, Greece, in 1986, and the M.Sc. and Ph.D. degrees from the University of Texas at Dallas, Richardson, TX, USA, in 1988 and 1991, respectively.

He is currently a Professor, the Chair of the Department of Electrical and Computer Engineering, and the Director of the National Science Foundation with the Industry University Cooperative Research Center on Embedded Systems, Southern Illinois University at Carbondale (SIUC), Carbondale, IL, USA. He was a Faculty Member with the Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ, USA, and the Department of Computer Science, SIUC. He has authored over 200 papers in journals and peer-reviewed conference proceedings. His current research interests include very large scale integration design and test automation and embedded systems.

Dr. Tragoudas was the Program Chair of DFTS in 2009 and the General Chair of DFTS in 2010. He has served and currently serves on the Editorial Board of several journals and the Technical Program Committees of many conferences. He was a recipient of three Outstanding Paper Awards for research in very large scale integration testing.