A Generalized Approach to Implement Efficient CMOS-Based Threshold Logic Functions

Seyed Nima Mozaffari, *Student Member, IEEE*, Spyros Tragoudas, *Senior Member, IEEE*, Themistoklis Haniotakis, *Member, IEEE*

***Abstract*— An integer linear programming-based framework to identify the current-mode threshold logic functions is presented. The approach minimizes the transistor count and benefits from a generalized definition of threshold logic functions. It is shown that the threshold logic functions can be implemented in CMOS-based current mode logic with reduced transistor count when the input weights are not restricted to be integers. A novel implementation of rational weights is proposed. Process variations, transistor aging, and circuit parasitics are taken into consideration. Experimental results show that many more functions can be implemented with predetermined hardware overhead, and the hardware requirement of a large percentage of the existing threshold functions is reduced when comparing with the traditional CMOS-based threshold logic implementation.**

***Index Terms*— Threshold logic gate, synthesis, threshold function, current mode design, weight assignment.**

I. Introduction

THRESHOLD Logic Gate (TG) is a promising candidate for the future digital circuits. Recent synthesis approaches show that TG-based circuits exhibit less delay, power dissipation, and silicon area [1]–[5]. A Boolean function that can be implemented as a single TG is called Threshold Logic Function (TF). In a TF, there is an integer weight for each input. When the input is set to binary value 1 then the weight is active. An input pattern evaluates the function to logic one only when the sum of the active weights is greater than (or equal) to a predetermined integer weight value called the threshold weight [10]–[12]. Otherwise, it evaluates the function to logic 0. In this paper, such a TF function is called a 1^{st}-order TF and will be denoted as a 1-TF. A n-input 1-TF $f(x_1, x_2, \ldots, x_n)$ is formally defined as [11]:

$$f(x_1, x_2, \cdots, x_n) = \begin{cases} 1 & if \ \sum_{i=1}^{n} w_i \cdot x_i \geq w_T \\ 0 & otherwise, \end{cases} \tag{1}$$

Manuscript received June 8, 2017; revised September 11, 2017 and October 18, 2017; accepted October 23, 2017. Date of publication November 22, 2017; date of current version February 15, 2018. This work was supported by NSF I/UCRC for Embedded Systems at SIUC under Grant NSF IIP 1432026 and Grant NSF IIP 1361847. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. This paper was recommended by Associate Editor F. Pareschi. *(Corresponding author: Seyed Nima Mozaffari.)*

The authors are with Department of Electrical and Computer Engineering, Southern Illinois University at Carbondale, Carbondale, IL 62901 USA (e-mail: nima.mozaffari@siu.edu; spyros@siu.edu; haniotak@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/TCSI.2017.2768563

where x_i, $i = 1, \ldots, n$, are binary input variables, w_i is the weight corresponding to the i^{th} input, and w_T is the threshold weight (threshold value). The 1-TF $f(x_1, x_2, \ldots, x_n)$ is also uniquely identified by the set of threshold and input integer weights $[w_1, w_2, \ldots, w_n; w_T]$.

1-TF implementations consist of three components: two differential networks (input networks) and a sensor (sense amplifier) [4]–[10]. One input network implements positive input weights and negative threshold weight and the other one implements the negative input weights and the positive threshold weight. The sensor evaluates the output by comparing and amplifying the difference between the sum of active weights of the two input networks.

In CMOS-based TGs, the power dissipation depends primarily on two factors: the transistor count of the input networks which is the total number of unit size transistors that implements weights and the sensor size which is proportional to the transistor count of input networks [5]. Input networks implement the weights with NMOS (or PMOS) transistors connected in parallel. Let X denote the width of a minimum size transistor which implements the unit weight. Each transistor implements a weight w and its width is $w \cdot X$, and the gate of the transistor is connected to an input. (The area of a transistor with width $w \cdot X$ is practically the same as w minimum size transistor.) The gate of the NMOS transistor for the threshold is connected to the power supply (it is active for all input patterns). The length of all transistors is the same and is determined by the used technology. The less transistor count in input networks, the lower the power dissipation in the differential network of a TG is. Furthermore, a reduced transistor count subsequently reduces the sensor size, which, in turn, decreases further the power dissipation [5].

This paper proposes a new method to reduce the transistor count of the input networks by introducing non-integer weights. This method is applicable to all existing TG implementations. In order to demonstrate the impact of non-integer weights on TGs in terms of area, power dissipation, and delay, this paper focuses on the current-mode TGs (CTGs) implementation which is a popular PMOS- and NMOS-based implementation. (See [5], [6], among others.)

A small fraction of binary functions are 1-TFs [11], and this limits the impact of TGs in digital circuit synthesis. Thus, our focus shifts on identifying TFs that are not 1-TF. A higher order TF, also called k^{th}-order TF (k-TF) in this paper, was introduced in [13]. For each input pattern, a weight in a k-TF can be activated by a group of k active inputs, $1 \leq k \leq n$. Such a weight is called a k-weight.

Let x_{i_1}, $i_1 = 1, \ldots, n$, be binary input variables, w_T denote the threshold weight, and integer weight $w_{i_1,i_2,\ldots,i_m}$ denote the m-weight, $1 \le m \le k$, that is activated by the group of m inputs $i_1, i_2, \ldots,$ and i_m. The k-TF formulation for an n-input function is [13]:

$$f(x_1, x_2, \cdots, x_n) = \begin{cases} 1 \quad if \displaystyle\sum_{i_1=1}^{n} w_{i_1} x_{i_1} + \ldots + \sum_{i_1=1}^{n-m+1} \sum_{i_2=i_1+1}^{n-m} \cdots \sum_{i_m=i_{m-1}+1}^{n} \\ \quad w_{i_1,i_2,\ldots,i_m} x_{i_1} x_{i_2} \ldots x_{i_m} + \ldots + \displaystyle\sum_{i_1=1}^{n-k+1} \sum_{i_2=i_1+1}^{n-k} \\ \quad \ldots \displaystyle\sum_{i_k=i_{k-1}+1}^{n} w_{i_1,i_2,\ldots,i_k} x_{i_1} x_{i_2} \ldots x_{i_k} \ge w_T \\ 0 \quad otherwise, \end{cases} \tag{2}$$

Let us consider the recently proposed synthesis approach in [4] where a flip-flop at the output of the circuit together with a portion of the predecessor combinational logic is replaced by a single TG. Since more functions can be identified as a single k-TG by Eq. (2), each output cone will more likely be implemented with fewer gates and one k-TG as the final gate.

This paper presents an Integer Linear Programming (ILP) formulation to assign efficiently weights to a k-TF so that the CTG implementation has low transistor count, and, subsequently, low power and delay. Weight variations due to CMOS-aging, circuit parasitics, and process variations are also taken into consideration.

It will be shown that many more TFs can be implemented as CTG without increased transistor count when compared to CTGs using traditional 1-TF definition. Moreover, the cost of certain CTGs that are 1-TF is reduced when considering the k-TF definition.

This paper is organized as follows. Section II provides preliminaries on 1-TFs, including a scalable integer linear programming (ILP) based method to identify 1-TF and assign weights. Section III proposes an ILP capable of reducing the transistor count using rational weight values. A new method to identify and implement higher order TFs is presented in Section IV. An efficient approach to implement higher order TFs using rational weights is presented in Section V. Section VI provides experimental results. Section V concludes the paper and outlines future work that includes fast synthesis of complex circuit specifications and beyond CMOS k-TF implementations. We will build upon existing 1-TF based synthesis methods [7], [14]–[24] and 1-TF resistive-based implementations [3], [27]–[31].

II. Preliminaries on the Algorithmic Infrastructure

This section provides with definitions and properties of 1-TF implemented using integer weights. It is also outlines some algorithmic aspects for scalable identification of threshold logic functions.

Positive (Negative) Function: Assume that function f is expressed in a disjunctive form. Function f is positive (negative) in variable x_i if the variable $\bar{x}_i$ (x_i) does not appear in the expression of f. Function f is a positive (negative) function if it is positive (negative) in all variables [11].

Unate Function (UF): Assume function f is expressed in a disjunctive form. f is unate in variable x_i if f is either positive or negative in variable x_i [11]. A function is a UF if it is unate in all variables. In other words, a function f is a UF, if and only if, for each variable x_i, $f_{x_i}(f_{\bar{x}_i}) \supseteq f_{\bar{x}_i}(f_{x_i})$.

Non-unate functions are called Binate Functions (BFs). All 1-TFs are UF [11]. However, higher order TFs may be BF [13].

Modified Chow's Parameters [11]: For an n-input function, the Modified Chow's parameter (m_i) of variable x_i, $1 \le i \le n$, is defined as the difference between the number of fully specified product terms in the onset of the function that have $x_i = 1$, and the fully specified product terms in the onset of that have $x_i = 0$.

Let $\vec{m}_f = (m_1, \ldots, m_n)$ be the set of Modified Chow's parameters of all n variables for an n-input UF f. Function f is positive (negative) in variable x_i, if and only if, $m_i > 0$ ($m_i < 0$). Therefore, function f is a positive (negative) UF if all members of set $\vec{m}_f$ are non-zero and positive (negative) [11].

Negation Property [11]: The negation of any variable results into a new set of weights. Let $f(x_1, x_2, \ldots, x_i, \ldots, x_n)$ be $[w_1, w_2, \ldots, w_i, \ldots, w_n; w_T]$. The negation of variable x_i, ($x_i \rightarrow \bar{x}_i$), changes the weight configuration to $[w_1, w_2, \ldots, -w_i, \ldots, w_n; w_T - w_i]$.

In order to determine if a function is a 1-TF, we form an ILP constraint per input pattern using the right-hand side of (1) according to the binary evaluation of the function for the pattern [11]. However, the ILP requires more variables in order to implement TFs with low hardware overhead. The objective function in the ILP must minimize the absolute value of the weights, some of which may be negative. Negative weights can be assigned using three variables for each weight. (Details are omitted for brevity.) However, such an approach introduces unnecessarily many variables and impacts ILP scalability. The following present an improved ILP where only one variable per weight is needed.

Let f be the examined non-positive UF. First, we find the Modified Chow's parameter for every variable. Every variable x_i that corresponds to negative weight w_i has a negative Modified Chow's parameter. Such variables must be complemented. This implies that the weight w_i will be activated when $x_i = 0$. The new set of input variables ensures that all weights are positive.

Next, we form an ILP for function f with the new set of variables, one variable per weight and an additional variable for the threshold weight. There is one constraint per input pattern to satisfy the functionality, and one constraint per variable that bind the range of that variable. The ILP objective minimizes the sum of the variables. The weight configuration is assigned from the ILP solution and the negation property. The following example illustrates the approach.

Example 1: Consider the three-input function $F_1 = x_2 + x_1 x_3'$. In F_1, $\vec{m}_{F_1} = (1, 3, -1)$. Parameter m_3 is negative, and hence variable x_3 must be complemented before forming the ILP. Table I lists the inequalities extracted from the truth value of F_1 using Eq. (1). The first three columns

TABLE I
THE LINEAR INEQUALITIES FOR $F_1 = x_2 + x_1 x_3'$ WITH ACTIVATION SIGNALS x_1, x_2, AND x_3'

Truth Table			Inequalities
	Input Value $(x_1x_2x_3)$	F_1	
P_0	000	0	$w_3 < w_T$
P_1	001	0	$0 < w_T$
P_2	010	1	$w_2 + w_3 \geq w_T$
P_3	011	1	$w_2 \geq w_T$
P_4	100	1	$w_1 + w_3 \geq w_T$
P_5	101	0	$w_1 < w_T$
P_6	110	1	$w_1 + w_2 + w_3 \geq w_T$
P_7	111	1	$w_1 + w_2 \geq w_T$
Minimize :			$w_T + \sum_{i=1}^{3} w_i$

show the truth table of function F_1. Column four shows the linear inequalities. Weights w_1 and w_2 are activated when $x_1 = 1$ and $x_2 = 1$, while weight w_3 is activated when $x_3 = 0$. The last row shows the objective function that minimizes quantity $w_T + \sum_{i=1}^{3} w_i$. For the set of constraints in Table I, $[w_1, w_2, w_3; w_T] = [1, 2, 1; 2]$ is an optimum solution. Hence, F_1 is a 1-TF and its weight configuration, using the negation property, is $w_{F_1} = [w_1, w_2, w_3; w_T] = [1, 2, -1; 1]$. □

The performance of the CMOS transistor which implements a weight is impacted by circuit parasitics, process variations, and aging. Transistor aging is caused by Bias Temperature Instability (BTI), dielectric breakdown, and hot career injections [24]–[26] that shift the threshold voltage and decrease the current through the transistors [32], [33]. This is called weight aging. Process variations impact transistor width and length, and therefore they may modify the designed weight. However, all CTG weights will be shifted by the same factor and in the same direction, and therefore weight assignment is not that sensitive to process variations. Finally, parasitics are evaluated accurately with post-layout simulations.

Let value C denote the maximum weight deviation due to the above factors. It is obtained with SPICE simulations, as explained on the predetermined technology as explained in Section VI. Let $|w|$ denote the absolute value of weight w. The pattern dependent inequalities of the ILP are rewritten as $\sum_i (w_i - C \cdot |w_i|) \cdot x_i > w_T + C \cdot |w_T|$ when function evaluates to 1, and as $\sum_i (w_i + C \cdot |w_i|) \cdot x_i < w_T - C \cdot |w_T|$ for the remaining input patterns. The above may only change the total sum of weights. For example, the weight configuration in Example 1 considering $C = 5\%$ becomes $w = [w_1, w_2, w_3; w_T] = [2, 4, -2; 1]$. This weight assignment results in a reliable CTG implementation.

III. EFFICIENT DESIGN OF FIRST-ORDER THRESHOLD FUNCTIONS BASED ON RATIONAL WEIGHTS

The previous section described an ILP formulation that implements 1-weight by assigning an integer value w to each input variable and the variable for the threshold. This section shows how to form an ILP capable of implementing weights that are fractions w/l, where w and l are integers, and is an extension of the preliminary results presented in [34] for the special case where $l = 2$.

In contrast to the ILP method in the previous section, each weight value assigned by the new ILP is different from the implemented weight. The novelty is that the ILP assigns an integer value w to each variable so that the value of the respective weight when evaluating the TF (as in Equation 1) is w/l for some predetermined integer l. The flexibility of implementing rational weights results into significant reduction in the transistor count of CTGs with subsequent reduction in power and delay.

In CTGs that have been identified as 1-TF, each integer weight is implemented by a component that is connected to the components for the other weights, including the threshold. Each input weight component is controlled by an input variable. Such weights and components are called 1^{st}-order weights (or 1-weight) and 1^{st}-order components, respectively.

Let X denote the width of a minimum size transistor and I the active current through this transistor. The CTG implementation in [6] implements an integer 1-weight with value w using either a single NMOS (or PMOS) transistor with width $w \cdot X$ or w minimum size transistors which are connected in parallel. The active current through this component is $w \cdot I$. A transistor (weight) is active when its corresponding input is set to 1 (or 0 in case of using PMOS).

The proposed approach implements a rational 1-weight using multiple minimum size transistors connected in series. A 1-weight with value w/l is implemented with l NMOS (or PMOS) transistors which are connected in series. The transistor gates are connected to each other and are controlled by the corresponding input. The width of each transistor is $w \cdot X$ and the active current through them is $\frac{w}{l} \cdot I$. The transistor count of this component is l times the transistor count of a 1^{st}-order component that implements an integer 1-weight with value w.

Fig. 1 shows the 1^{st}-order components that implement rational 1-weights with value $1/j$, for $1 \leq j \leq l$, given a predetermined integer l. The figure also shows the active current through them. The active current decreases when j increases. It is true that adding a transistor increases the capacitance of that component, and this increases the power dissipation. However, when a TF is implemented by rational components of Fig. 1, the total transistor count of the gate reduces, and this reduces significantly the power dissipation of the gate.

Let x_i, $i_1 = 1, \ldots, n$, be binary input variables, j, w_i^j, and w_T^j integer values. w_i^j the 1-weight component with value w_i^j/j corresponding to the i^{th} input, $1 \leq j \leq l$, and w_T^j the threshold weight component with value w_T^j/j. In the proposed method, the ILP assigns l different integer weights w_i^j for each input i, $1 \leq j \leq l$, but the value of each w_i^j is $\frac{1}{j} w_i^j$ when the TF is evaluated. When the TF is evaluated, the total value for input variable x_i is $\sum_{j=1}^{l} \frac{1}{j} w_i^j$. Likewise, the ILP assigns l different integer weights w_T^j for the threshold, $1 \leq j \leq l$. However, the value of each w_T^j is $\frac{1}{j} w_T^j$ when the TF is evaluated. The total value for the threshold weight w_T is $\sum_{j=1}^{l} \frac{1}{j} w_T^j$ when the TF is evaluated.Based on the above,

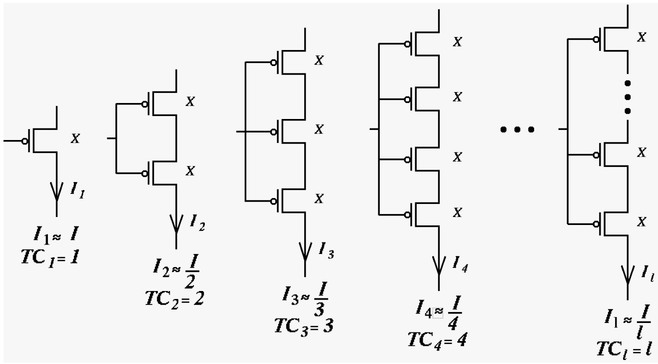


Fig. 1. 1^{st}-order components that implement rational 1-weights with value $1/j$, for $1 \le j \le l$.

an n-input 1-TF $f(x_1, x_2, \ldots, x_n)$ is defined as

$$f(x_1, x_2, \cdots, x_n) = \begin{cases} 1 \quad if \ \sum_{i=1}^{n}((\sum_{j=1}^{l}\frac{1}{j}w_i^j)\cdot x_i) \ge \sum_{j=1}^{l}\frac{1}{j}w_T^j \\ 0 \quad otherwise, \end{cases} \tag{3}$$

Although a 1-weight component with value w/l requires a significant number of transistors, the flexibility of allowing rational weights allows the ILP to assign values to the respective variables so that the sum of the values assigned is much lower than the sum of values for the ILP that is restricted to integer weights. That way, the transistor count for the CTG may be reduced.

Before we elaborate on the details in forming the ILP of a function, we show that we have identified 1-TFs whose transistor count is reduced when considering rational weights due to the flexibility in selecting the appropriate weight values. The following shows that a 1-TF with integer weights can be implemented with less number of transistors when considering rational weights when $l = 2$, i.e., non-integer weights that are multiples of 0.5. It also shows that the transistor count reduces further when $l = 4$, i.e., non-integer weights that are multiples of 0.25.

Example 2: Consider the 5-variable function $F_2 = x_2x_5' + x_1'x_4' + x_1'x_5' + x_1'x_4'x_5'$. It is a TF with optimum integer weight configuration $w = [w_1, w_2, w_3, w_4, w_5; w_T] = [-7, 9, 2, -5, -12; -4]$ using the ILP in Section II and considering $C = 5\%$. The transistor count for implementing input weights is 39. However, when $l = 2$ (weights are multiple of 0.5), this function can be implemented as $w' = \left[w_1^1, w_2^1, w_3^1, w_4^1, w_4^2, w_5^1, w_5^2; w_T^1\right] = [-4, 5, 1, -2, -1, -6, -1; -2]$. The total transistor count reduces to 24. Moreover, when $l = 4$, weight set changes to $w'' = \left[w_1^1, w_1^4, w_2^1, w_2^4, w_3^2, w_4^1, w_4^4, w_5^1; w_T^1\right] = [-2, 1, 2, 1, 1, -1, -1, -3; -1]$, and the total transistor count reduces to 23.

Fig. 2 shows the CTG implementation of F_2 with integer weights [6] and the proposed method when $l = 4$. In Fig. 2, X denotes the size of a minimum width transistor to implement a unit integer 1-weight. □

The proposed ILP formulation is an extension of the one in [11] that was explained in Section II. It has $2^n + l(n + 1)$ constraints with $l(n + 1)$ unknown variables. There is a constraint per input pattern to satisfy the functionality, and $l(n+1)$ constraints that determine the range of each variable. The weight at input i and the threshold weight are $\sum_{j=1}^{l}\frac{1}{j}w_i^j$ and $\sum_{j=1}^{l}\frac{1}{j}w_T^j$, respectively, for some integer value j and predetermined l.

TABLE II

THE TRUTH TABLE AND THE ILP CONSTRAINTS FOR UF F_3. ($0 < w_T^1 + 0.5 \cdot w_T^2 \Longleftrightarrow 0 < 2 \cdot w_T^1 + w_T^2$)

Truth Table			Inequalities
Input Pattern ($x_1x_2x_3$)		F_3	
P_0	000	0	$0 < 2w_T^1 + w_T^2$
P_1	001	0	$2w_3^1 + w_3^2 < 2w_T^1 + w_T^2$
P_2	010	0	$2w_2^1 + w_2^2 < 2w_T^1 + w_T^2$
P_3	011	1	$2w_2^1 + w_2^2 + 2w_3^1 + w_3^2 > 2w_T^1 + w_T^2$
P_4	100	1	$2w_1^1 + w_1^2 > 2w_T^1 + w_T^2$
P_5	101	1	$2w_1^1 + w_1^2 + 2w_3^1 + w_3^2 > 2w_T^1 + w_T^2$
P_6	110	1	$2w_1^1 + w_1^2 + 2w_2^1 + w_2^2 > 2w_T^1 + w_T^2$
P_7	111	1	$2w_1^1 + w_1^2 + 2w_2^1 + w_2^2 + 2w_3^1 + w_3^2 > 2w_T^1 + w_T^2$
$\forall\, w_x^y \in \{w_T^1, w_1^1, w_2^1, w_3^1\}$: $0 \le w_x^y \le 10$			
$\forall\, w_x^y \in \{w_T^2, w_1^2, w_2^2, w_3^2\}$: $w_x^y \in \{0, 1\}$			
minimize:			$w_T^1 + 2 \cdot w_T^2 + \sum_{i=1}^{3}(w_i^1 + 2 \cdot w_T^2)$

In addition, the ILP must minimize the transistor count considering that any rational 1-weight with value w/l requires l times more transistors than any integer 1-weight with value w. Therefore, the ILP must minimize quantity

$$\sum_{j=1}^{l} j w_T^j + \sum_{i=1}^{n}\left(\sum_{j=1}^{l} j w_i^j\right) \tag{4}$$

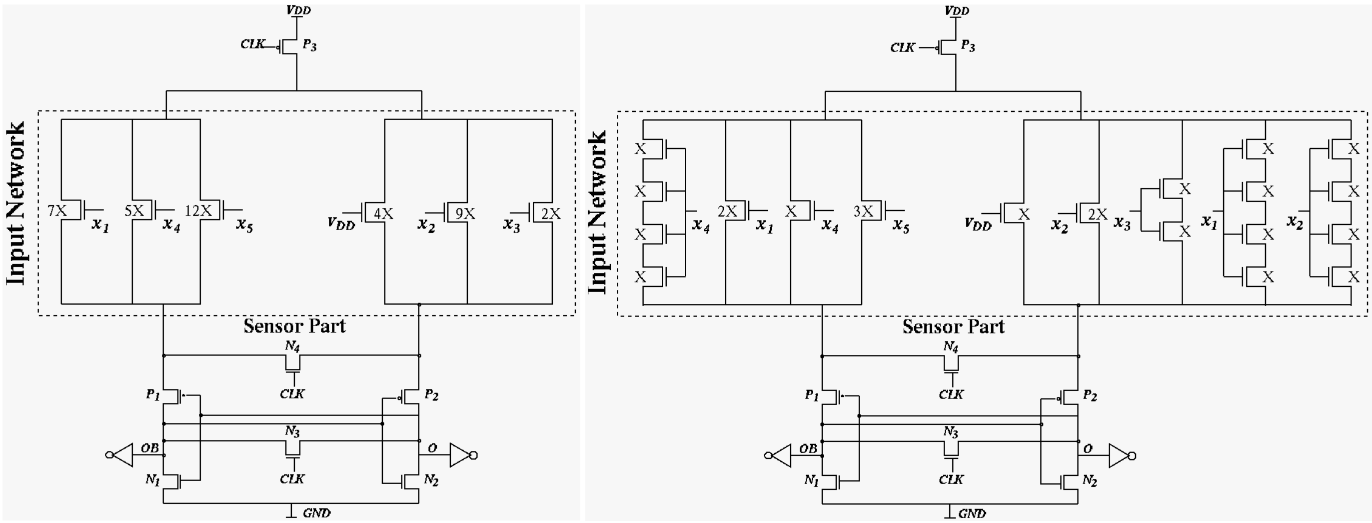
The example below illustrates the ILP-based approach to identify a 1-TF and assign optimum weights that are multiples of 0.5 ($l = 2$).

Example 3: Consider UF $F_3 = x_1 + x_2x_3$. Table II lists the ILP inequalities of F_3 based on the proposed ILP framework and 1-TF formulation in (3) considering $l = 2$. The last row shows the objective function of ILP-solver introduced in (4). For the set of constraints listed in Table II, $w = \left[w_1^1, w_2^1, w_3^1, w_4^1; w_T^1, w_T^2\right] = [2, 1, 1; 1, 1]$ is an optimum solution for F_3. □

For non-positive UF f, in order to avoid implementing the negative weights, we form the above mentioned ILP by using the method explained in Section II.

IV. HIGHER-ORDER IMPLEMENTATION OF THRESHOLD FUNCTIONS USING INTEGER WEIGHTS

A small fraction of binary functions are 1-TFs [3] and can be implemented as a single TG. In order to identify more threshold functions and increase the impact of TFs in digital synthesis, we consider the generalized k-TF definition described in Equation 2. Preliminary results for the special case where $k = 2$ were presented in [35]. This section shows TF implementations using integer weights. The next section



(a) (b)

Fig. 2. The CTG implementation for function F_2 in example 3 when using (a) integer weights [6] (b) rational 1-weights with value w/j, for when $1 \leq j \leq l$ and $l = 4$.

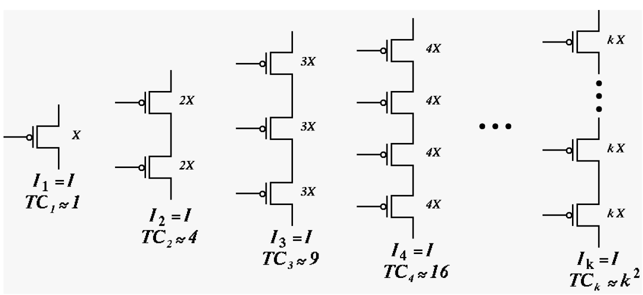


Fig. 3. k-weight components for $1 \leq k \leq 4$.

shows that the transistor count can be further reduced using rational weights.

A k-weight is implemented with k NMOS (or PMOS) transistors of the same size (according to the respective weight) which are connected in series, and the transistor gates are connected to k CTG inputs. The size of each transistor in a component that implements a k-weight with value w is set to $k \cdot w \cdot X$ to keep the active current of a unit k-weight equal to the active current of a unit 1-weight. Thus, the total transistor count of a k-weight component is k^2 times more than the transistor count of a 1-weight that implements the same weight value. This transistor count ratio is called the penalty factor and is taken into consideration to force the ILP-solver to find the minimum possible transistor count. Such gates are called k-CTGs. Fig. 3 shows the k-weight components, their transistor count, and active current through them for $1 \leq k \leq 4$.

The approach is presented in two steps. First, we consider UFs for which the ILP does not use many variables. Then, we present an ILP for BFs. The latter requires more variables and is less scalable.

An ILP formulation is presented to implement a UF as a k-CTG, as in (2) with minimum transistor count. The proposed ILP is an improvement of [13]. The ILP contains $2^n + n' + 1$ constraints with $n' + 1$ variables, where $n' = \sum_{m=1}^{k} \binom{n}{m}$ is the total number of m-weights, $1 \leq m \leq k$. There is a constraint per input pattern to satisfy the functionality, and $n' + 1$ constraints that bind the range of threshold and each input weight. Once a UF f is given, the Modified Chow's parameters [11] for all inputs and groups of inputs determine the negative weights. To form an efficient ILP, every product of m inputs $x_{i_1} \cdot x_{i_2} \cdot \ldots \cdot x_{i_m}$, $1 \leq m \leq k$, that activates an m-weight with a negative Modified Chow's parameter must be complemented. A 1-weight (m-weight when $m = 1$) with negative Modified Chow's parameter is activated as in 1-TF. A m-weight, $2 \leq m \leq k$, with negative Modified Chow's parameter will be activated when at least one of its input is set to 0. The ILP with the appropriate activation signals determines whether function f is a k-TF. The penalty factors appear as the coefficient of weights in the objective function of the ILP. The following objective function minimizes the k-CTG transistor count:

$$w_T + 1 \cdot \sum_{i_1=1}^{n} w_{i_1} + 4 \cdot \sum_{i_1=1}^{n-1} \sum_{i_2=2}^{n} w_{i_1,i_2} + \ldots + k^2 \cdot \sum_{i_1=1}^{n-k+1} \sum_{i_2=i_1+1}^{n-k} \ldots \sum_{i_k=i_{k-1}+1}^{n} w_{i_1,i_2,\ldots,i_k} \quad (5)$$

The weight configuration will then be assigned using the ILP solution and the negation property. The following examples illustrate the concept of k-TF and the ILP-based method to identify a k-TF.

Example 4: Consider a 4-input UF $F_4 = x_1' + x_2' + x_4'$ with a set of all unknown weights $w = [w_1, w_2, w_3, w_4, w_{1,2}, w_{1,3}, w_{1,4}, w_{2,3}, w_{2,4}, w_{3,4}; w_T]$. The set of Modified Chow's parameters of either an input or a pair of inputs that activates a weight is $\vec{m}_{F_1} = (-5, -1, +3, -1, -9, -5, -9, -5, -7, -5)$. To form an efficient ILP, first every product term (activation signal) with negative m_i must be complemented. In this example, all inputs and pairs of inputs must be complemented except x_3.

Table III lists the inequalities of F_4 based on the 2-TF definition and by considering positive weights. For an input

TABLE III

THE TRUTH TABLE AND ILP CONSTRAINTS FOR F_4 CONSIDERING ALL INPUTS AND PAIRS OF INPUTS ARE COMPLEMENTED EXCEPT x_3

Truth Table			Inequalities
Input Pattern ($x_1x_2x_3x_4$)		F_4	
P_0	0000	1	$w_1+w_2+w_4+w_{1,2}+w_{1,3}+w_{1,4}+w_{2,3}+w_{2,4}+w_{3,4} \geq w_T$
P_1	0001	1	$w_1+w_2+w_{1,2}+w_{1,3}+w_{1,4}+w_{2,3}+w_{2,4}+w_{3,4} \geq w_T$
P_2	0010	1	$w_1+w_2+w_3+w_4+w_{1,2}+w_{1,3}+w_{1,4}+w_{2,3}+w_{2,4}+w_{3,4} \geq w_T$
P_3	0011	1	$w_1+w_2+w_3+w_{1,2}+w_{1,3}+w_{1,4}+w_{2,3}+w_{2,4} \geq w_T$
P_4	0100	1	$w_1+w_4+w_{1,2}+w_{1,3}+w_{1,4}+w_{2,3}+w_{2,4}+w_{3,4} \geq w_T$
P_5	0101	1	$w_1+w_{1,2}+w_{1,3}+w_{1,4}+w_{2,3}+w_{3,4} \geq w_T$
P_6	0110	1	$w_1+w_3+w_4+w_{1,2}+w_{1,3}+w_{1,4}+w_{2,4}+w_{3,4} \geq w_T$
P_7	0111	1	$w_1+w_3+w_{1,2}+w_{1,3}+w_{1,4} \geq w_T$
P_8	1000	0	$w_2+w_4+w_{1,2}+w_{1,3}+w_{1,4}+w_{2,3}+w_{2,4}+w_{3,4} < w_T$
P_9	1001	0	$w_2+w_{1,2}+w_{1,3}+w_{2,3}+w_{2,4}+w_{3,4} < w_T$
P_{10}	1010	1	$w_2+w_3+w_4+w_{1,2}+w_{1,4}+w_{2,3}+w_{2,4}+w_{3,4} \geq w_T$
P_{11}	1011	1	$w_2+w_3+w_{1,2}+w_{2,3}+w_{2,4} \geq w_T$
P_{12}	1100	0	$w_4+w_{1,3}+w_{1,4}+w_{2,3}+w_{2,4}+w_{3,4} < w_T$
P_{13}	1101	0	$w_{1,3}+w_{2,3}+w_{3,4} < w_T$
P_{14}	1110	1	$w_3+w_4+w_{1,4}+w_{2,4}+w_{3,4} \geq w_T$
P_{15}	1111	0	$w_3 < w_T$
minimize:			$w_T+1\cdot\sum_{i=1}^{4} w_i+4\cdot\sum_{i=1}^{3}\sum_{j=i+1}^{4} w_{i,j}$

pattern, weight w (either 1-weight or 2-weight) appears in the inequality when its activation signal evaluates to 1. The objective function for a 2-TF is to minimize quantity $w_T + 1 \cdot \sum_{i=1}^{4} w_i + 4 \cdot \sum_{i=1}^{3}\sum_{j=2}^{4} w_{i,j}$. For the set of constraints listed in Table III, an optimum solution is $w = [3, 0, 2, 0, 0, 0, 0, 1, 0, 1; 2]$. Using the negation property, $F_4 = [-3, 0, 2, 0, 0, 0, 0, -1, 0, -1; -2]$. When considering $C = 5\%$ weight variation, the weight configuration becomes $w = [-6, 0, 4, 0, 0, 0, 0, -2, 0, -2; -5]$. □

Example 5: Function $F_5 = x_1x_2' + x_1'x_2 + x_1x_2x_3'$ is neither an 1-TF nor 2-TF. It is a 3-TF and the weight configuration to implement as a 3-CTG is $w = [w_1, w_2, w_{1,2,3}; w_T] = [2, 2, -4; 1]$ considering 5% weight variation ($C = 5\%$). □

The solutions for functions F_4 and F_5 in Examples 4 and 5 illustrate that many 1-weight, 2-weights, and 3-weights are assigned to zero. This means that a k-TF does not necessarily need all k-weight components when implementing as a k-CTG. Moreover, the transistor count, and hence, the hardware requirement of many existing threshold functions (1-TFs) is reduced using higher order components. The following show that a 1-TF can be implemented as proposed 2-CTG with lower cost than the respective 1-CTG.

Example 6: Consider the 4-variable function $F_6 = x_4x_3 + x_3x_2 + x_3x_1 + x_2x_1$. It is a 1-TF with optimum weight configuration $w = [w_1, w_2, w_3, w_4; w_T] = [4, 4, 6, 2; 7]$ using the ILP in [11]. The transistor count or the total sum of threshold and input weights of 1-CTG is 23. Let X denote the area of a unit 1-weight transistor. The total area of the input networks is $23 \cdot X$.

However, this function can be implemented as a 2-CTG with weight configuration $w = [w_1, w_2, w_3, w_4, w_{1,2}, w_{1,3}, w_{1,4}, w_{2,3}, w_{2,4}, w_{3,4}; w_T] = [2, 2, 2, 0, 0, 0, 0, 0, 0, 2; 3]$. Each 2-weight requires 4 more times the transistor count of a 1-weight that implements the same weight. The transistor count reduces to $(2 + 2 + 2) + 4 \cdot (2) + (3) = 17$, and thus, the total area of the input networks of the 2-CTG reduces to $17 \cdot X$.

Fig. 4 shows the 1-CTG [6] and proposed 2-CTG implementations of F_6 and the size of transistors that implements 1-weights and 2-weights considering X as the size of a minimum width transistor to implement a unit 1-weight. The length of all the PMOS and NMOS transistors is the same and determined by the used technology. □

The remaining of the section considers BFs. The correlation between the sign of the Modified Chow's parameters and the sign of weights only holds for UFs. The ILP for a BF works as in [13] and contains $2^n + 3(n + n' + 1)$ constraints with $3(n + n' + 1)$ variables, where $n' = \binom{n}{2}$ is the total number of 2-weights. Each weight can be either positive or negative. The objective function is to minimize quantity

$$|w_T| + 1 \cdot \sum_{i_1=1}^{n} |w_{i_1}| + 4 \cdot \sum_{i_1=1}^{n-1}\sum_{i_2=2}^{n} |w_{i_1,i_2}| + \ldots + k^2 \cdot \sum_{i_1=1}^{n-k+1}\sum_{i_2=i_1+1}^{n-k} \cdots \sum_{i_k=i_{k-1}+1}^{n} |w_{i_1,i_2,\ldots,i_k}| \quad (6)$$

where $|w_x|$, denotes the absolute value for each weight w_x, and $w_x \in \{w_T, w_i, w_{i,j}\}$. Let y_x be a binary variable, and U denote a predetermined upper bound of $|w_x|$ for each weight w_x. For each w_x, two variables w_x^+ and w_x^- are used, and the bound on the absolute value w_x is enforced using the following constraints:

$$\begin{cases} 0 \leq w_x^+ \leq U \cdot y_x \\ 0 \leq w_x^- \leq U \cdot (1 - y_x) \\ y_x \in \{0, 1\} \end{cases} \quad (7)$$

Then $w_x = w_x^+ - w_x^-$. In addition, for CTG the ILP should minimize quantity

$$w_T^+ + w_T^- + \sum_{i_1=1}^{n} ({w_{i_1}}^+ + w_{i_1}^-) + 4 \cdot \sum_{i_1=1}^{n-1}\sum_{i_2=2}^{n} ({w_{i_1,i_2}}^+ + w_{i_1,i_2}^-) + \ldots + k^2 \cdot \sum_{i_1=1}^{n-k+1}\sum_{i_2=i_1+1}^{n-k} \cdots \sum_{i_k=i_{k-1}+1}^{n} ({w_{i_1,i_2,\ldots,i_k}}^+ + w_{i_1,i_2,\ldots,i_k}^-) \quad (8)$$

The example below illustrates the ILP-based approach to identify a BF as a 2-TF and assign optimum weights to implement 2-CTG with minimum possible transistor count.

Example 7: Consider BF $F_7 = x_1x_3' + x_2x_3$. Table IV lists the ILP inequalities of F_7 based on the k-TF formulation in (2) when $k = 2$. The last two rows show the constraints and the objective function of ILP-solver introduced in (7) and (8) to assign negative weights and minimize the sum of weights. For the set of constraints listed in Table IV, $w = [w_1, w_2, w_3, w_{1,2}, w_{1,3}, w_{2,3}; w_T] = [2, 0, 0, 0, -2, 2; 1]$ is an optimum solution for F_7 when considering 5% weight variation. □

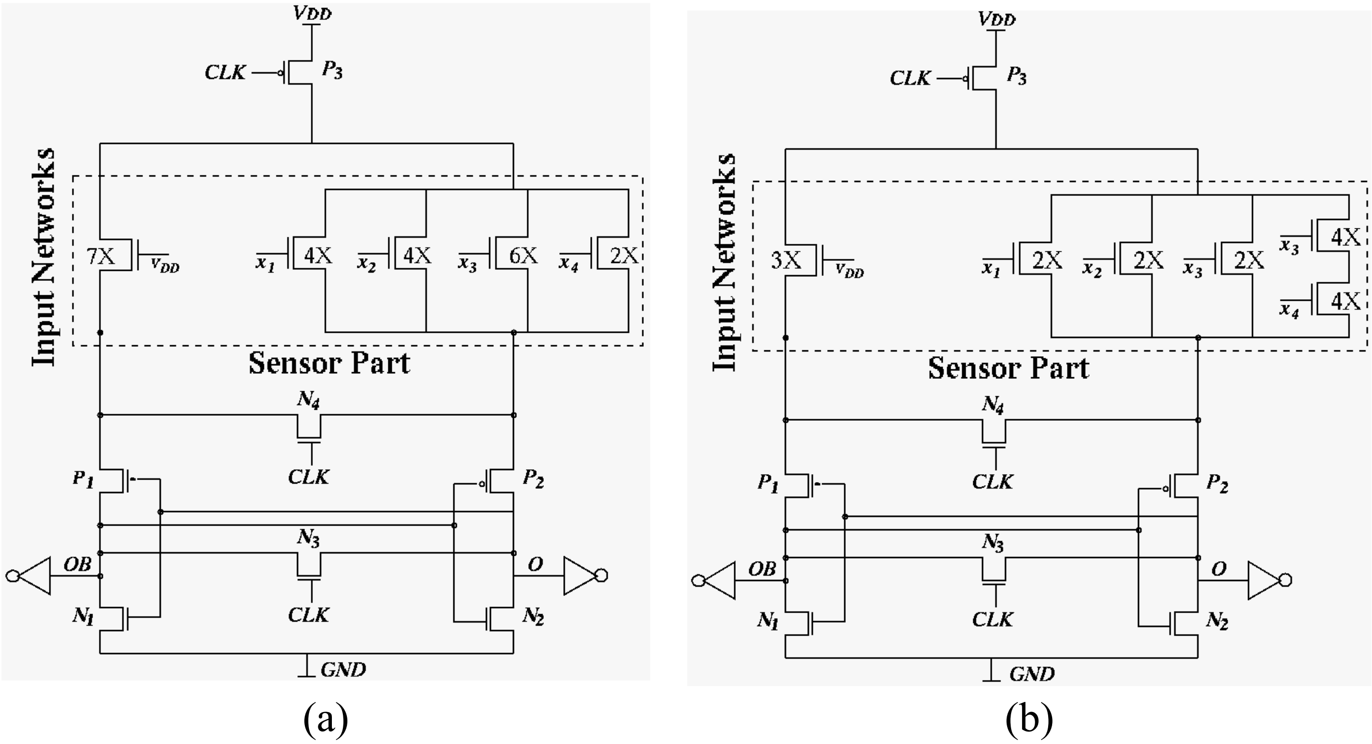


Fig. 4. The CTG implementation for function $F_6 = x_4x_3 + x_3x_2 + x_3x_1 + x_2x_1$ with (a) 1-CTG as 1-TF [6] (b) proposed 2-CTG as 2-TF [34].

TABLE IV

THE TRUTH TABLE AND THE ILP CONSTRAINTS FOR BF F_7

Truth Table			Inequalities
Input Pattern ($x_1x_2x_3$)		F_7	
P_0	000	0	$0 < {w_T}^+ - {w_T}^-$
P_1	001	0	${w_3}^+ - {w_3}^- < {w_T}^+ - {w_T}^-$
P_2	010	0	${w_2}^+ - {w_2}^- < {w_T}^+ - {w_T}^-$
P_3	011	1	${w_3}^+ - {w_3}^- + {w_2}^+ - {w_2}^- + {w_{2,3}}^+ - {w_{2,3}}^- \geq {w_T}^+ - {w_T}^-$
P_4	100	1	${w_1}^+ - {w_1}^- \geq {w_T}^+ - {w_T}^-$
P_5	101	0	${w_3}^+ - {w_3}^- + {w_1}^+ - {w_1}^- + {w_{1,3}}^+ - {w_{1,3}}^- < {w_T}^+ - {w_T}^-$
P_6	110	1	${w_2}^+ - {w_2}^- + {w_1}^+ - {w_1}^- + {w_{1,2}}^+ - {w_{1,2}}^- \geq {w_T}^+ - {w_T}^-$
P_7	111	1	${w_3}^+ - {w_3}^- + {w_2}^+ - {w_2}^- + {w_1}^+ - {w_1}^- + {w_{2,3}}^+ - {w_{2,3}}^- + {w_{1,3}}^+ - {w_{1,3}}^- + {w_{1,2}}^+ - {w_{1,2}}^- \geq {w_T}^+ - {w_T}^-$
$\forall\, {w_x}^+, {w_x}^- \in \{{w_T}^+, {w_T}^-, {w_1}^+, {w_1}^-, {w_2}^+, {w_2}^-, {w_3}^+, {w_3}^-, {w_{1,2}}^+, {w_{1,2}}^-, {w_{1,3}}^+, {w_{1,3}}^-, {w_{2,3}}^+, {w_{2,3}}^-\}$:			$\begin{cases} 0 \leq {w_x}^+ \leq U \cdot y_x \\ 0 \leq {w_x}^- \leq U \cdot (1 - y_x) \\ y_x \in \{0,1\} \end{cases}$
minimize: $\quad {w_T}^+ + {w_T}^- + \sum_{i=1}^{3}({w_i}^+ + {w_i}^-) + 4 \times \sum_{i=1}^{2}\sum_{j=i+1}^{3}({w_{i,j}}^+ + {w_{i,j}}^-)$			

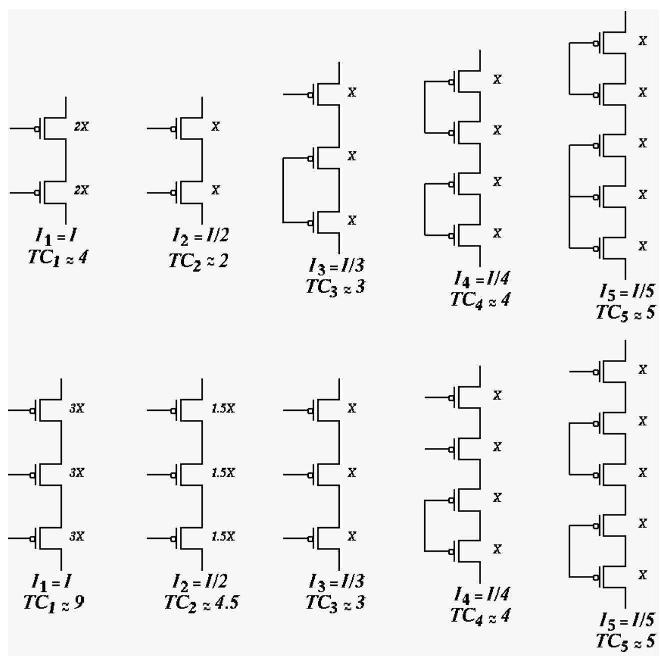


Fig. 5. Rational k-weights components with values $1/j$ for $2 \leq k \leq 3$ and $1 \leq j \leq 5$.

V. EFFICIENT DESIGN OF HIGHER-ORDER THRESHOLD FUNCTIONS USING RATIONAL WEIGHTS

This section applies the method of Section III on high order TFs so that their implementation has less transistor count because of rational weight assignment.

In k-TFs, each weight component may be controlled by more than one input. When $l \leq k$, a k-weight with value w/l is implemented with k transistors connected in series each with size $\frac{w \cdot k}{l} \cdot X$. In this case, the transistor count is $\frac{w}{l} \cdot k^2$ times the transistor count of a unit integer 1-weight component.

For the case when $l > k$, a k-weight can be implemented with l transistors connected in series each with size $w \cdot X$. In this case, the transistor count of this component is $l \cdot w$ times the transistor count of a unit integer 1-weight component.

The ILP will select the implementation with the smallest penalty factor in order to find the minimum possible transistor count. Fig. 5 considers $l = 5$, and shows the k-weights for $k = 2$ and $k = 3$. In particular, it shows all minimum penalty factor components for $k = 2$ and $1 < j \leq 5$ as well as for $k = 3$ and $1 < j \leq 5$.

Let I denote the active current through a minimum size transistor that implements a unit integer 1-weight. The active current through a rational k-weight component with value w/l is $\frac{w}{l} \cdot I$.

Let $w^1_{i_1,i_2,\ldots,i_m}$ be an integer value for a weight component that is activated by m inputs $i_1, i_2, \ldots,$ and i_m. Let also $w^j_{i_1,i_2,\ldots,i_m} \in \{0, 1, \ldots, j-1\}$, $2 < j \le l$, denote a rational m-weight with value $w^j_{i_1,i_2,\ldots,i_m}/j$ corresponding to the group of m inputs $i_1, i_2, \ldots,$ and i_m, and $1 \le m \le k$. The m-weight $w_{i_1,i_2,\ldots,i_m}$ is represented by $\sum_{j=1}^{l} \frac{1}{j} w^j_{i_1,i_2,\ldots,i_m}$ in the definition of the TF. Based on the above,

$$
f(x_1, x_2, \cdots, x_n) = \begin{cases} 1 \quad if \ \sum_{i_1=1}^{n}\left(\left(\sum_{j=1}^{l}\frac{1}{j}w^j_{i_1}\right)\cdot x_{i_1}\right) \\ \qquad + \ldots + \sum_{i_1=1}^{n-k+1}\sum_{i_2=i_1+1}^{n-k}\cdots\sum_{i_k=i_{k-1}+1}^{n}((\sum_{j=1}^{l}\frac{1}{j}w^j_{i_1,i_2,\ldots,i_k}) \\ \qquad \cdot x_{i_1}x_{i_2}\ldots x_{i_k}) \ge \sum_{j=1}^{l}\frac{1}{j}w^j_T \\ 0 \quad otherwise, \end{cases} \tag{9}
$$

where x_{i_1}, $i_1 = 1, \ldots, n$, are binary input variables, j, $w^j_{i_1,i_2,\ldots,i_m}$, and w^j_T are integer values, and w^j_T is the threshold weight with value w^j_T/j.

Before we elaborate on the details in forming the ILP of a function, we show that we have identified k-TFs whose transistor count is reduced when considering rational weights due to the flexibility in selecting the appropriate weight values.

Example 8: Consider again function F_5 in Example 5. The weight configuration changes to $w = [w^1_1, w^2_1, w^1_2, w^2_2, w^1_{3,4,5}; w^1_T] = [1, 1, 1, 1, 1; 2]$ with transistor count 17 and $w = [w^1_1, w^1_2, w^3_{3,4,5}; w^1_T, w^3_T] = [1, 1, 2; 1, 1]$ with transistor count 12 when l is 2 and 3, respectively. The transistor count may decrease when l increases. □

When comparing to a minimum size transistor as a unit integer 1-weight, the component that implements any rational k-weight with value w/l, for $k > 1$ or $l > 1$, imposes more transistor count. Therefore, an effective ILP-based framework is needed to identify a TF and assign appropriate weights using minimum possible number of non-integer higher order weights.

The ILP formulations to identify and implement either a UF or a BF as a k-CTG with minimum transistor count is an extension of the formulations presented in Section IV. In particular, we start with UF, and then we present the ILP for BF.

The ILP formulation to identify and implement a UF as a k-CTG using non-integer weights has $2^n + l(n'+1)$ constraints with $l(n'+1)$ variables, where $n' = \sum_{i=1}^{k}\binom{n}{i}$ is the total number of m-weights, $1 \le m \le k$. The Modified Chow's parameters of all groups of m inputs determine the negative weights (including all m-weights). To form an efficient ILP, every product of m inputs $(x_{i_1} \cdot x_{i_2} \cdots x_{i_m})$, that activates a m-weight, with a negative Modified Chow's parameter must be complemented. An m-weight with negative Modified Chow's parameter will be activated when at least one of x_{i_1}, x_{i_2}, and x_{i_m} is set to 0. The ILP with the appropriate activation signals determines whether function f is a k-TF.

TABLE V

THE TRUTH TABLE AND THE ILP CONSTRAINTS FOR UF F_4. ($0 < w^1_T + 0.5 \cdot w^2_T \iff 0 < 2 \cdot w^1_T + w^2_T$)

Truth Table			Inequalities
Input Pattern ($x_1x_2x_3$)		F_4	
P_0	000	0	$0 < 2w^1_T + w^2_T$
P_1	001	0	$2w^1_3 + w^2_3 < 2w^1_T + w^2_T$
P_2	010	0	$2w^1_2 + w^2_2 < 2w^1_T + w^2_T$
P_3	011	1	$2w^1_2 + w^2_2 + 2w^1_3 + w^2_3 + 2w^1_{2,3} + w^2_{2,3} > 2w^1_T + w^2_T$
P_4	100	1	$2w^1_1 + w^2_1 > 2w^1_T + w^2_T$
P_5	101	1	$2w^1_1 + w^2_1 + 2w^1_3 + w^2_3 + 2w^1_{1,3} + w^2_{1,3} > 2w^1_T + w^2_T$
P_6	110	1	$2w^1_1 + w^2_1 + 2w^1_2 + w^2_2 + 2w^1_{1,2} + w^2_{1,2} > 2w^1_T + w^2_T$
P_7	111	1	$2w^1_1 + w^2_1 + 2w^1_2 + w^2_2 + 2w^1_3 + w^2_3 + 2w^1_{1,2} + w^2_{1,2}$ $+2w^1_{1,3} + w^2_{1,3} + 2w^1_{2,3} + w^2_{2,3} > 2w^1_T + w^2_T$
minimize: $w^1_T + 2 \cdot w^2_T + \sum_{i=1}^{n}(w^1_i + 2 \cdot w^2_i) + \sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(4 \cdot w^1_{i,j} + 2 \cdot w^2_{i,j})$			

For each input pattern, any m-weight $w_{i_1,i_2,\ldots,i_m}$, $1 \le m \le k$, in Section IV is substituted with l unknown variables so that $w_{i_1,i_2,\ldots,i_m} = \sum_{j=1}^{l}\frac{1}{j}w^j_{i_1,i_2,\ldots,i_m}$. Likewise, the threshold weight is replaced by (5). In addition, the ILP must minimize the transistor count considering that any m-weight with value w/j requires $j \cdot f(j-k) + \frac{m^2}{j} \cdot f(k-j)$ multiplied by the minimum size transistor that implements a unit integer 1-weight, where $f(t)$ is the unit step function that evaluates to 1 when $t \ge 0$, and evaluates to 0 when $t < 0$. Therefore, the ILP must minimize quantity

$$
\sum_{j=1}^{l} j \cdot w^j_T + \sum_{i_1=1}^{n}\left(\sum_{j=1}^{l} j \cdot w^j_{i_1}\right) + \ldots + \sum_{i_1=1}^{n-k+1}\sum_{i_2=i_1+1}^{n-k}\cdots\sum_{i_k=i_{k-1}+1}^{n} \times(\sum_{j=1}^{l}(j \cdot f(j-k) + \frac{k^2}{j} \cdot f(k-j)) \cdot w^j_{i_1,i_2,\ldots,i_k}) \tag{10}
$$

The weight configuration will then be assigned using the solution from ILP and the negation property. The example below illustrates the ILP-based approach to identify a 2-TF and assign optimum half integer weights ($l = 2$).

Example 9: Consider again the UF F_4 in Example 4 with a set of non-zero integer weights $w = [w_1, w_3, w_{2,3}, w_{3,4}; w_T] = [-6, 4, -2, -2; -5]$. Let each input weight variable in weight set w is replaced by $\sum_{j=1}^{2}\frac{1}{j}w^j_{i_1,i_2,\ldots,i_m}$. Likewise, the threshold variable w_T is replaced by $\sum_{j=1}^{2}\frac{1}{j}w^j_T$. Table V lists the inequalities of F_4. The last row shows the objective function of ILP-solver introduced in Equation 10. For the set of constraints listed in Table V, $w = [w^1_1, w^1_2, w^1_3, w^1_4; w^1_T, w^2_T] = [2, 1, 1; 1, 1]$ is an optimum solution for F_4. □

Many of the constraints in the ILP are redundant. We use the simplification method which is the extension of the one in [22] to eliminate redundant constraints which makes the ILP formulation smaller and possibly faster to solve. As an example consider the UF in Example 9. If $2w^1_1 + w^2_1 > 2w^1_T + w^2_T$, any constraint containing

TABLE VI
THE TRUTH TABLE AND THE ILP CONSTRAINTS FOR BF F_7

Truth Table			Inequalities
Input Pattern ($x_1x_2x_3$)		F_7	
P_0	000	0	$0 < 2w_T^{1+} - 2w_T^{1-} + w_T^{2+} - w_T^{2-}$
P_1	001	0	$2w_3^{1+} - 2w_3^{1-} + w_3^{2+} - w_3^{2-} < 2w_T^{1+} - 2w_T^{1-} + w_T^{2+} - w_T^{2-}$
P_2	010	0	$2w_2^{1+} - 2w_2^{1-} + w_2^{2+} - w_2^{2-} < 2w_T^{1+} - 2w_T^{1-} + w_T^{2+} - w_T^{2-}$
P_3	011	1	$+2w_3^{1+} - 2w_3^{1-} + w_3^{2+} - w_3^{2-} + 2w_2^{1+} - 2w_2^{1-} + w_2^{2+} - w_2^{2-}$ $+2w_{2,3}^{1}{}^{+} - 2w_{2,3}^{1}{}^{-} + w_{2,3}^{2}{}^{+} - w_{2,3}^{2}{}^{-} \geq 2w_T^{1+} - 2w_T^{1-} + w_T^{2+} - w_T^{2-}$
P_4	100	1	$2w_1^{1+} - 2w_1^{1-} + w_1^{2+} - w_1^{2-} \geq 2w_T^{1+} - 2w_T^{1-} + w_T^{2+} - w_T^{2-}$
P_5	101	0	$2w_1^{1+} - 2w_1^{1-} + w_1^{2+} - w_1^{2-} + 2w_3^{1+} - 2w_3^{1-} + w_3^{2+} - w_3^{2-}$ $+2w_{1,3}^{1}{}^{+} - 2w_{1,3}^{1}{}^{-} + w_{1,3}^{2}{}^{+} - w_{1,3}^{2}{}^{-} < 2w_T^{1+} - 2w_T^{1-} + w_T^{2+} - w_T^{2-}$
P_6	110	1	$2w_1^{1+} - 2w_1^{1-} + w_1^{2+} - w_1^{2-} + 2w_2^{1+} - 2w_2^{1-} + w_2^{2+} - w_2^{2-}$ $+2w_{1,2}^{1}{}^{+} - 2w_{1,2}^{1}{}^{-} + w_{1,2}^{2}{}^{+} - w_{1,2}^{2}{}^{-} \geq 2w_T^{1+} - 2w_T^{1-} + w_T^{2+} - w_T^{2-}$
P_7	111	1	$2w_1^{1+} - 2w_1^{1-} + w_1^{2+} - w_1^{2-} + 2w_2^{1+} - 2w_2^{1-} + w_2^{2+} - w_2^{2-}$ $+2w_3^{1+} - 2w_3^{1-} + w_3^{2+} - w_3^{2-} + 2w_{1,2}^{1}{}^{+} - 2w_{1,2}^{1}{}^{-} + w_{1,2}^{2}{}^{+} - w_{1,2}^{2}{}^{-}$ $+2w_{1,3}^{1}{}^{+} - 2w_{1,3}^{1}{}^{-} + w_{1,3}^{2}{}^{+} - w_{1,3}^{2}{}^{-} + 2w_{2,3}^{1}{}^{+} - 2w_{2,3}^{1}{}^{-} + w_{2,3}^{2}{}^{+}$ $- w_{2,3}^{2}{}^{-} \geq 2w_T^{1+} - 2w_T^{1-} + w_T^{2+} - w_T^{2-}$
$\forall\, w_x{}^{\mp} \in \{w_T^{1\mp}, w_T^{2\mp}, w_1^{1\mp}, w_1^{2\mp}, w_2^{1\mp}, w_2^{2\mp}, w_3^{1\mp}, w_3^{2\mp} w_{1,2}^{1}{}^{\mp}, w_{1,2}^{2}{}^{\mp}, w_{1,3}^{1}{}^{\mp}, w_{1,3}^{2}{}^{\mp}, w_{2,3}^{1}{}^{\mp}, w_{2,3}^{2}{}^{\mp}\}$:			$\begin{cases} 0 \leq w_x{}^{+} \leq U \cdot y_x \\ 0 \leq w_x{}^{-} \leq U \cdot (1 - y_x) \\ y_x \in \{0,1\} \end{cases}$
minimize:			$w_T^{1+} + w_T^{1-} + 2w_T^{2+} + 2w_T^{2-} + \sum_{i=1}^{3}\sum_{j=1}^{2} j \cdot (w_i^{j+} + w_i^{j-}) +$ $\sum_{a=1}^{2}\sum_{b=a+1}^{3}\sum_{j=1}^{2}(j \cdot f(j-2) + \frac{4}{j} \cdot f(2-j)) \cdot (w_{a,b}^{j}{}^{+} + w_{a,b}^{j}{}^{-})$

$2w_1^1 + w_1^2$ must be greater than $2w_T^1 + w_T^2$. Therefore, the last 3 constraints are redundant and can be removed from ILP.

The correlation between the sign of the Modified Chow's parameters and the sign of weights only holds for UFs. The ILP formulation to identify and implement a BF as a k-CTG using non-integer weights contains 3 times more unknown variables that the one for a UF. Each weight can be either positive or negative. The objective function is to minimize the sum of absolute value of variables that are implemented using Equation 7. In addition, for k-CTG the ILP should minimize quantity

$$\sum_{j=1}^{l} j\,(w_T^{j+} + w_T^{j-}) + \sum_{i_1=1}^{n}\left(\sum_{j=1}^{l} j\left(w_{i_1}^{j+} + w_{i_1}^{j-}\right)\right) + \ldots + \sum_{i_1=1}^{n-k+1}\sum_{i_2=i_1+1}^{n-k}\cdots\sum_{i_k=i_{k-1}+1}^{n} \times (\sum_{j=1}^{l}(j \cdot f\,(j-k) + \frac{k^2}{j} \cdot f\,(k-j)) \times (w_{i_1,i_2,\ldots,i_k}^{j+} + w_{i_1,i_2,\ldots,i_k}^{j-})) \tag{11}$$

The simplification method in [22] is not extendable to BFs. Therefore, the ILP for a BF slower than the one for a UF. The example below illustrates the ILP-based approach to identify a BF as a 2-TF and assign optimum integer and non-integer weights to implement efficient 2-CTG.

Example 10: Consider again the BF F_7 in Example 7 with set of integer weight values $w = [w_1, w_{1,3}, w_{2,3}; w_T] = [2, -2, 2; 1]$ and transistor count 19. Table VI lists the ILP inequalities of F_7 based on the k-TF formulation described in Equation 9. For simplicity, we consider $k = 2$ and $l = 2$. The last two rows show the ILP constraints and the ILP objective function of introduced in Equations 7 and 11 to assign negative weights and minimize the sum of weights. For the set of constraints listed in Table VI, $w = [w_1^1, w_1^2, w_{1,3}^1, w_{1,3}^2, w_{2,3}^1, w_{2,3}^2; w_T^1, w_T^2] = [1, 0, -1, 0, 1, 0; 0, 1]$ is an optimum solution for F_7. The transistor count reduces to 11. □

VI. EXPERIMENTAL RESULTS

The proposed ILP-based approach has been implemented in the C++ language on an Intel Xenon 2.4GHz with 8GB memory. To evaluate its impact, we examined non-scalable functions with up to fifteen inputs. An n-input non-scalable function is a function that requires non-empty levels of variables in the Binary Decision Diagram (BDD) representation for some ordering of the variables [35]. In another word, in a non-scalable function, no input variable is don't care, and, therefore, all variables (and/or their complements) appear in the minimum sum-of-product expression of the function.

Table VII presents non-scalable n-input k-TFs using rational k-weights with value w/l for different n, k, and l that were derived using the ILP of Section V. For each value of n we found the 1-TFs with maximum transistor count. This was set as a bound to the objective function for any ILP formulation for $l \geq 4$ and $k \geq 4$. The first column in Table VII shows the number of inputs (value of n). The goal in this paper is to provide an indication of the percentage of all n-input functions that benefit from the proposed method. When n is large it is impossible to examine all functions and, therefore, functions are selected randomly. For functions with $n \geq 4$, the entries in Table VII were obtained by sampling randomly 100 thousand functions. For $n \geq 4$, the 2^n bit output vector of an n-input function was filled with either 0 or 1 at randomly selected positions (determined by randomly selecting an integer mod 2^n), and so that the number of ones in the function obeyed the distribution of functions based on this property. (For example, the number of 5-input functions with 16 ones in the output bit vector is approximately 10 times more than the number of 5-input functions with 10 ones.) In order for the experiment to have more statistical significance, we only considered non-scalable functions, and when a function is generated we applied the procedure described earlier in this section to determine that it is non-scalable. (It is asserted that the distribution of non-scalable n-input functions based on the number of ones in their output bit vector is the same as the one described earlier for n-input functions.)

The second column in Table VII lists the number of 1-TFs identified by using the existing ILP-based method in [11] considering integer weights. These functions are implementable with existing CTGs. The third column shows the examined values of l, $l \in \{1, 2, 3, 4\}$. The fourth column shows the number of 2-TFs that do not have transistor count higher than any of the 1-TF in column two. The fifth column shows the percentage increase over the number of 1-TFs in column two. Columns six to nine show similar results for $k \in \{3, 4\}$. For all examined functions, the value of C was set to 11% to take into consideration that weights may vary primarily due

TABLE VII

THE NUMBER OF k-CTGs WITH RATIONAL k-WEIGHTS OF VALUE w/l WHOSE TRANSISTOR COUNT IS NO MORE THAN 1-CTGS WITH INTEGER WEIGHTS

n	1-TF	l	$k=2$	INCREASE RATIO	$k=3$	INCREASE RATIO	$k=4$	INCREASE RATIO
1	2	1	2	1	2	1	2	1
		2	2	1	2	1	2	1
		3	2	1	2	1	2	1
		4	2	1	2	1	2	1
2	8	1	8	1	10	1.25	10	1.25
		2	10	1.25	10	1.25	10	1.25
		3	10	1.25	10	1.25	10	1.25
		4	10	1.25	10	1.25	10	1.25
3	72	1	72	1	72	1	72	1
		2	188	2.61	192	2.66	192	2.66
		3	214	2.97	214	2.97	214	2.97
		4	216	3	218	3.02	218	3.02
4	1536	1	2566	1.7	4454	2.9	4761	3.1
		2	9012	5.87`	11059	7.2	11366	7.4
		3	9872	6.43	11063	7.2	12748	8.3
		4	11464	7.46	21043	13.7	21196	13.8
5*	367	1	4514	12.3	8110	22.1	9395	25.6
		2	7743	21.1	10202	27.8	10716	29.2
		3	7964	21.7	10312	28.1	11046	30.1
		4	8514	23.2	11450	31.2	12698	34.6
6*	105	1	1176	11.2	1827	17.4	1848	17.6
		2	1659	15.8	2467	23.5	2740	26.1
		3	1690	16.1	2478	23.6	2772	26.4
		4	1827	17.4	2887	27.5	3318	31.6
7*	89	1	1593	17.9	1877	21.1	1993	22.4
		2	2607	29.3	2919	32.8	3262	36.7
		3	2776	31.2	3017	33.9	3271	36.7
		4	2860	36.2	3506	39.4	3871	43.5
8*	66	1	1498	22.7	3438	52.1	3517	53.3
		2	3095	46.9	4468	67.7	4659	70.6
		3	3590	54.4	4481	67.9	4699	71.2
		4	3973	60.2	4765	72.2	4870	73.8
9*	306	1	2234	7.3	3610	11.8	3855	12.6
		2	5385	17.6	5905	19.3	6671	21.8
		3	5393	17.6	6089	19.9	6762	22.1
		4	7160	23.4	9057	29.6	9394	30.7
10*	119	1	1499	12.6	1880	15.8	1892	15.9
		2	2034	17.1	2983	25.1	3058	25.7
		3	2094	17.6	2991	25.1	3082	25.9
		4	2094	17.6	3177	26.7	3344	28.1
11*	65	1	886	14.3	1241	19.1	1521	23.4
		2	1670	25.7	2054	31.6	2132	32.8
		3	1813	27.9	2073	31.9	2190	33.7
		4	2067	31.8	2216	34.1	2314	35.6
12*	45	1	513	11.4	1053	23.4	1206	26.8
		2	801	17.8	1624	36.1	1773	39.4
		3	869	19.3	1643	36.5	1778	39.5
		4	886	19.7	1818	40.4	1939	43.1
13*	92	1	1343	14.6	2658	28.9	-	-
		2	2907	31.6	5042	54.8	-	-
		3	2925	31.8	5048	54.8	-	-
		4	2930	31.8	5981	65.0	-	-
14*	45	1	810	18.0	1404	31.2	-	-
		2	1219	27.1	2448	54.4	-	-
		3	1242	27.6	2583	57.4	-	-
		4	1480	32.9	3289	73.1	-	-
15*	97	1	1571	16.2	-	-	-	-
		2	2473	25.5	-	-	-	-
		3	2475	25.5	-	-	-	-
		4	2774	28.6	-	-	-	-
TOTAL	**3014**		48257	16.01	72193	23.7	75218	**24.9**

* Out of 100 thousand randomly selected non-scalable functions.

to aging. This value for C was obtained by performing SPICE simulations on a single transistor that implements a unit integer 1-weight in corner cases using 45nm technology [28] while the transistor was continuously under stress (worst-case aging scenario). We used the static aging model in [36] and we found that the transistor threshold voltage shifted by 50 mV under continuous stress. This increase in threshold voltage amounted to 11% decrease in the current.

TABLE VIII

AVERAGE EXECUTION TIME (ms) PER FUNCTION FOR n-INPUT k-TFs (UF AND BF), $6 \le n \le 15$, $1 \le k \le 4$, $C = 11\%$ CONSIDERING RATIONAL k-WEIGHTS WITH VALUE w/l, $l \in \{1, 4\}$

TF Type \ n	6	7	8	9	10	11	12	13	14	15
1-TF, l=1	50	54	72	115	250	490	1108	2e3	3e3	7e3
1-TF, l=4	56	60	78	122	258	505	1617	22e2	36e2	8e3
UF 2-TF, l =1	54	60	80	125	275	508	1200	23e2	4e3	8e3
UF 2-TF, l =4	60	67	113	178	320	670	1420	28e2	5e3	1e4
BF 2-TF, l=1	63	96	150	290	540	1e3	2e3	5e3	9e3	2e4
BF 2-TF, l=4	88	103	189	502	596	18e2	39e2	1e4	2e4	-
UF 3-TF, l=1	225	270	324	561	1e3	22e2	5e3	85e2	2e4	-
UF 3-TF, l=4	289	450	511	887	15e2	35e2	67e2	13e3	35e3	-
BF 3-TF, l=1	460	605	617	11e2	21e2	5e3	8e3	-	-	-
BF 3-TF, l=4	481	619	705	16e2	29e2	9e3	13e3	-	-	-
UF 4-TF, l =1	13e2	14e2	2e3	3e3	65e2	14e3	3e4	-	-	-
UF 4-TF, l =4	17e2	17e2	25e3	7e3	91e2	17e3	4e4	-	-	-
BF 4-TF, l =1	2e3	2e3	4e3	8e3	13e3	-	-	-	-	-
BF 4-TF, l =4	23e3	3e3	51e2	15e3	-	-	-	-	-	-

TABLE IX

THE NUMBER OF 1-TFs WITH LOWER TRANSISTOR COUNT WHEN CONSIDERING RATIONAL k-WEIGHTS WITH VALUE w/l, $k \le 4$, $l \le 4$, $C = 11\%$

n	1-TF	Δ (percentage reduction in CTG transistor count)			
		Δ<40%	40%≤Δ<50%	50%≤Δ<60%	Δ≥60%
1	2	2	0	0	0
2	8	8	0	0	0
3	72	72	0	0	0
4	1536	112	1242	176	6
5*	367	3	263	101	0
6*	105	0	67	31	7
7*	89	0	50	37	2
8*	66	0	57	9	0
9*	306	0	233	73	0
10*	119	0	102	14	3
11*	65	0	53	10	2
12*	45	0	41	4	0
TOTAL	2780	197	2108	455	20

* Out of 100 thousand randomly selected non-scalable functions.

The results in Table VII show that for higher value of k and l, the ILP has more flexibility to assign weights so that the total transistor count decreases. Therefore, when k and l increase more functions can be implemented as current mode gates using a transistor count similar to that for the significantly less 1-TFs that were implemented as 1-CTG. In particular, when $k = 4$ and $l = 4$, about 24.9 times more functions can be implemented as CTGs with similar or less transistor count.

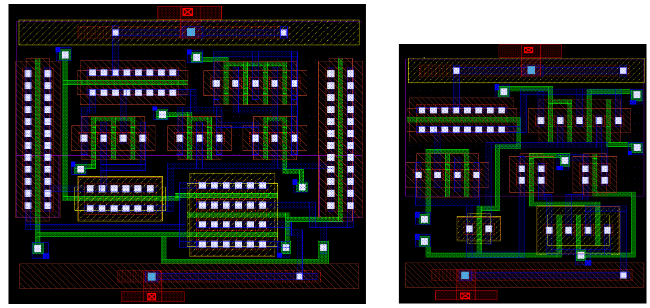
Table VIII lists the average execution time by the proposed ILP method to determine whether an examined n-input function (BF and UF) was implementable as proposed k-TF described in Equation 9 for different values of k and l, and C set to 11%. In our experimental evaluation, we set up an execution time upper bound of 60 second per TF, at which point the function was aborted. Character "-" indicates that no results were obtained due to violation of the execution time upper bound. Observe, however, that the approach can handle all the UFs with at most 12 inputs. They can be implemented to up to the 4^{th} order when considering rational weight with value up to 4. These results show that the average execution time increases as the values of n, k and l increases. This is due to the increase in the number of unknown variables in the ILP. For higher input functions (functions with more fan-ins), heuristic approaches as in [14]–[16] can be used to implement UFs. However, they will not ensure that all TFs can be identified, and the weight configuration of the identified TFs is not necessarily optimum. Furthermore, they do not apply to BFs.

Table IX lists the number of 1-TFs identified by using the existing ILP-based method in [11] considering integer weights. Let Δ denote the percentage reduction in CTG transistor count. Columns three to eight show the number of TFs listed in second column that were implemented with less transistor count when considering the proposed k-CTG described in Equation 9 for $k, l \leq 4$ non-integer weights. These columns were generated based on different ranges of Δ, and C set to 11%.

The results in Table IX show that by increasing either k or l, many 1-TFs were implemented as CTG with lower transistor count, and hence with lower area and power dissipation. In particular, in particular, 93% of selected 1-TFs were implemented with approximately 45% lower transistor count.

The following compares the transistor count of input networks, power dissipation, and delay of the CTG implementation of randomly selected 1-TFs and 2-TFs described by the weight configuration set. All functions were implemented using the CTG in [6] and [35] considering only integer weights and using the proposed k-CTG described in Equation 9 considering $k \leq 4$ and $l \leq 4$, and C set to 11%. SPICE simulation took place for each function using the Berkeley Predictive Technology Models (PTM) for 45nm CMOS transistors [37]. The V_{DD} was set to 1.1V. The applied voltages for clk were 1.1V and 0V for high voltage and low voltage, respectively. The applied load was a minimum size CMOS inverter which had a PMOS transistor with width 240nm, and a NMOS transistor with width 120nm. The length of all the PMOS and NMOS transistors were fixed to 45nm. The optimum sensor size was obtained using the approach in [5].

The first Column in Table X lists some randomly selected functions. They are denoted by the integer weight assignment that reflects minimum transistor count. Columns two to four list the transistor count of the input networks using the traditional approach, the power dissipation, and the delay of the CTG implementation for each function, respectively, using the approaches in [6] and [35]. These values were obtained with SPICE simulations while considering corner cases by simultaneously varying the width and length of all transistors in input networks as well as the sensor part. Let d_V denote the voltage difference between two output nodes. Due to the clock enable in CTGs the delay is calculated as the time difference between the time that clock is at 50% of its final value and the time that d_V is at 50% of its final value [5]. The power value reported in column three is an average value that includes leakage and dynamic power dissipation [38], [39].



(a) (b)

Fig. 6. (a) The layout for 1-CTG implementation of function $[w_1, w_2, w_3; w_T] = [4, 2, 2; 3]$ as in [6] (b) the layout of the same function implemented using rational weights $\left[w_1^1, w_2^1, w_3^1; w_T^1, w_T^2\right] = [2, 1, 1; 1, 1]$.

Columns five to eight show similar results when each function is implemented by the proposed k-CTG of Section IV based on the formulation described in Equation 9. In particular, column five lists the obtained weights, column six lists the transistor count, column seven the power, and column eight the delay. The values in columns seven and eight were obtained by SPICE simulations.

The last three columns in Table X list the percentage reduction in transistor count, power dissipation, and delay, respectively, when compared to the traditional 1-TF current-mode implementation as in [6] or the 2-TF current-mode implementation as in [35]. The results show a significant decrease in power dissipation as well as a significant decrease in the delay due to the rational higher-order weights.

After the functions in Table X were synthesized by both the proposed method and the traditional in [6] and [35], we proceeded to obtaining their layouts in 45nm using the Berkeley PTM model, and we derived the silicon area. Furthermore, we conducted post-layout simulation to determine the power and delay (leakage and dynamic). This experiment was conducted in order to confirm that optimizing the transistor count at the input networks (as obtained by proposed ILP method) results into area reduction when compared to the traditional CTG methods in [6] and [35], and that delay and power are also reduced proportion to the saving shown by simulation at the synthesis level.note that post-layout simulation taken to consideration the circuit parasitics which were extracted from the layout of the CTG.

As an example, Fig. 6 (a) shows the layout of the first function listed in Table X with integer weights $[w_1, w_2, w_3; w_T] =$ [4, 2, 2; 3] as in [6], and Fig. 6 (b) shows the layout using rational weights $\left[w_1^1, w_2^1, w_3^1; w_T^1, w_T^2\right] =$ [2, 1, 1; 1, 1] by the proposed method. In this case, the reduction in the area of the layout is 40%. For the remaining functions in Table X we observed that the reduction in area is even more significant.

Table XI provides with details on the layout area savings for all functions in Table X. Please see the listed results in columns two, six, and nine. It is important to observe that the reduction

TABLE X

SIMULATION RESULTS: TRANSISTOR COUNT, POWER DISSIPATION, AND DELAY OF RANDOMLY SELECTED TFs IN 45 nm TECHNOLOGY USING THE CTG IN [6] AND [35] AND THE PROPOSED k CTG BASED ON (9) USING k-WEIGHTS WITH VALUE w/l, $k \leq 4$, $l \leq 4$, $C = 11\%$

CTG Implementation [6] and [36]				Proposed k-CTG with rational weights				% Reduction		
Function with Integer Weights	Transistor count	Power (μW)	Delay (ps)	Optimized Function for $k \leq 4$ and $l \leq 4$	Transistor count	Power (μW)	Delay (ps)	Transistor Count	Power	Delay
$[w_1, w_2, w_3; w_T] =$ $[4,2,2; 3]$	11	1.98	98	$[w_1^1, w_2^1, w_3^1; w_T^1, w_T^2] =$ $[2,1,1; 1,1]$	7	1.40	66	36%	30%	32%
$[w_1, w_3, w_{2,3}, w_{3,4}; w_T] =$ $[-6,4,2,-2; -3]$	29	3.73	113	$[w_1^1, w_3^1, w_{2,3}^1, w_{3,4}^2; w_T^1, \mathrm{w}_\mathrm{T}^2] =$ $[-3,2,1,-1; -1,-1]$	14	1.98	87	51%	47%	23%
$[w_1, w_2, w_3, w_4; w_T] =$ $[-4,4,-2,2; 3]$	15	2.80	142	$[w_1^1, w_2^1, w_3^1, w_4^1; w_T^1, w_T^2] =$ $[-2,2,-1,1; 1,1]$	10	1.74	105	33%	38%	26%
$[w_1, w_2, w_3, w_4, w_5; w_T] =$ $[-7,9,2,-5,-12; -4]$	39	4.25	158	$[w_1^1, w_1^4, w_2^1, w_2^4, w_3^2, w_4^1, w_4^4, w_5^1; w_T^1] =$ $[-2,1,2,1,1,-1,-1,-3; -1]$	22	2.12	122	43%	50%	23%
$[w_1, w_2, w_3, w_4, w_5, w_6; w_T] =$ $[4,4,7,-4,-11,-11; 5]$	43	4.08	129	$[w_1^1, w_2^1, w_3^1, w_4^1, w_5^1, w_6^1; w_T^1, w_T^4] =$ $[1,1,2,-1,-3,-3; 1,1]$	16	1.92	103	62%	53%	20%
$[w_1, w_2, w_3, w_{1,4}; w_T] =$ $[2,2,4,2; 3]$	19	3.21	118	$[w_1^1, w_2^1, w_3^1, w_{1,4}^1; w_T^1, \mathrm{w}_\mathrm{T}^2] =$ $[1,1,2,1; 1,1]$	11	1.93	87	42%	40%	26%
$[w_1, w_2, w_3, w_4, w_5, w_6, w_7, w_8, w_9; w_T] =$ $[-5,-5,-5,-5,-5,-2,2,3,3; 4]$	39	3.30	183	$[w_1^1, w_2^1, w_3^1, w_4^1, w_5^1, w_6^1, w_6^3, w_7^1, w_7^3, w_8^1, w_9^1; w_T^1, w_T^3]$ $[-2,-2,-2,-2,-2,-1,1,1,-1,1,1; 1,1]$	24	2.15	143	38%	35%	22%
$[w_1, w_2, w_3, w_4, w_5; w_T] =$ $[5,5,3,3,-3; 4]$	23	3.76	114	$[w_1^1, w_2^1, w_3^1, w_4^1, w_5^1; w_T^1, w_T^3] =$ $[2,2,1,1,-1; 1,1]$	11	1.88	80	52%	50%	30%
$[w_1, w_2, w_3, w_4, w_5; w_T] =$ $[-2,-2,4,6,6; 1]$	21	3.67	173	$[w_1^1, w_2^1, w_3^1, w_4^1, w_5^1; w_T^2] =$ $[-1,-1,2,3,3; 1]$	12	1.83	130	42%	50%	25%
$[w_1, w_{1,3}, w_{2,3}; w_T] =$ $[2,-2,2; 1]$	19	3.21	151	$[w_1^1, w_{1,3}^1, w_{2,3}^1; w_T^2] =$ $[1,-1,1; 1]$	11	1.80	96	42%	44%	36%
$[w_1, w_2, w_3, w_4, w_5, w_6, w_7, w_8, w_9, w_{10}; w_T] =$ $[2,2,2,2,2,2,2,2,2,2; 1]$	21	4.37	163	$[w_1^1, w_2^1, w_3^1, w_4^1, w_5^1, w_6^1, w_7^1, w_8^1, w_9^1, w_{10}^1; w_T^2] =$ $[1,1,1,1,1,1,1,1,1,1; 1]$	12	1.53	112	43%	65%	31%

TABLE XI

POST-LAYOUT RESULTS: CHIP AREA, POWER DISSIPATION, AND DELAY OF RANDOMLY SELECTED TFs, IN 45nm TECHNOLOGY USING THE CTG IN [6] AND [35] AND THE PROPOSED k-CTG BASED ON (9) USING k-WEIGHTS WITH VALUE w/l, $k \leq 4$, $l \leq 4$, $C = 11\%$

CTG Implementation [6] and [36]				Proposed k-CTG with rational weights				% Reduction		
Function with Integer Weights	Area (μm²)	Power (μW)	Delay (ps)	Optimized Function for $k \leq 4$ and $l \leq 4$	Area (μm²)	Power (μW)	Delay (ps)	Area	Power	Delay
$[w_1, w_2, w_3; w_T] =$ $[4,2,2; 3]$	10.50	4.40	167	$[w_1^1, w_2^1, w_3^1; w_T^1, w_T^2] =$ $[2,1,1; 1,1]$	6.25	1.98	129	40%	55%	23%
$[w_1, w_3, w_{2,3}, w_{3,4}; w_T] =$ $[-6,4,2,-2; -3]$	16.80	5.90	196	$[w_1^1, w_3^1, w_{2,3}^1, w_{3,4}^2; w_T^1, \mathrm{w}_\mathrm{T}^2] =$ $[-3,2,1,-1; -1,-1]$	8.25	1.94	160	51%	67%	18%
$[w_1, w_2, w_3, w_4; w_T] =$ $[-4,4,-2,2; 3]$	13.10	9.32	169	$[w_1^1, w_2^1, w_3^1, w_4^1; w_T^1, w_T^2] =$ $[-2,2,-1,1; 1,1]$	6.80	3.63	133	48%	61%	21%
$[w_1, w_2, w_3, w_4, w_5; w_T] =$ $[-7,9,2,-5,-12; -4]$	20.00	8.44	326	$[w_1^1, w_1^4, w_2^1, w_2^4, w_3^2, w_4^1, w_4^4, w_5^1; w_T^1] =$ $[-2,1,2,1,1,-1,-1,-3; -1]$	11.40	2.95	270	43%	65%	17%
$[w_1, w_2, w_3, w_4, w_5, w_6; w_T] =$ $[4,4,7,-4,-11,-11; 5]$	23.80	11.81	470	$[w_1^1, w_2^1, w_3^1, w_4^1, w_5^1, w_6^1; w_T^1, w_T^4] =$ $[1,1,2,-1,-3,-3; 1,1]$	9.10	5.07	399	62%	57%	15%
$[w_1, w_2, w_3, w_{1,4}; w_T] =$ $[2,2,4,2; 3]$	13.00	9.21	283	$[w_1^1, w_2^1, w_3^1, w_{3,4}^1; w_T^1, \mathrm{w}_\mathrm{T}^2] =$ $[1,1,2,1; 1,1]$	6.90	4.32	223	47%	53%	21%
$[w_1, w_2, w_3, w_4, w_5, w_6, w_7, w_8, w_9; w_T] =$ $[-5,-5,-5,-5,-5,-2,2,3,3; 4]$	18.40	9.16	351	$w_1^1, w_2^1, w_3^1, w_4^1, w_5^1, w_6^1, w_6^3, w_7^1, w_7^3, w_8^1; w_T^1, w_T^3] =$ $[-2,-2,-2,-2,-2,-1,1,1,-1,1,1; 1,1]$	11.40	4.67	284	38%	49%	19%
$[w_1, w_2, w_3, w_4, w_5; w_T] =$ $[5,5,3,3,-3; 4]$	13.80	8.25	490	$[w_1^1, w_2^1, w_3^1, w_4^1, w_5^1; w_T^1, w_T^3] =$ $[2,2,1,1,-1; 1,1]$	6.90	3.05	357	50%	63%	27%
$[w_1, w_2, w_3, w_4, w_5; w_T] =$ $[-2,-2,4,6,6; 1]$	12.90	9.79	275	$[w_1^1, w_2^1, w_3^1, w_4^1, w_5^1; w_T^2] =$ $[-1,-1,2,3,3; 1]$	7.10	3.91	223	45%	60%	19%
$[w_1, w_{1,3}, w_{2,3}; w_T] =$ $[2,-2,2; 1]$	12.50	8.80	415	$[w_1^1, w_{1,3}^1, w_{2,3}^1; w_T^2] =$ $[1,-1,1; 1]$	6.90	4.05	298	45%	54%	28%
$w_1, w_2, w_3, w_4, w_5, w_6, w_7, w_8, w_9, w_{10}; w_T] =$ $[2,2,2,2,2,2,2,2,2,2; 1]$	12.90	8.45	317	$[w_1^1, w_2^1, w_3^1, w_4^1, w_5^1, w_6^1, w_7^1, w_8^1, w_9^1, w_{10}^1; w_T^2] =$ $[1,1,1,1,1,1,1,1,1,1; 1]$	7.10	2.45	253	45%	71%	20%

in layout area is similar to the transistor count reduction in the input networks of those functions, as obtained by the proposed ILP-based synthesis method. These results show the impact of using rational weights in synthesis.

Table XI also lists detailed results on power dissipation and delay obtained from post-layout simulations using the traditional approaches in [6] and [35] and the proposed method. For power-related results please see columns three, seven, and ten. For delay-related results please see columns four, eight, and eleven. Again, observe that the reduction in power and delay by the proposed method, reflect the savings that were shown earlier in Table X. In fact, the post-layout simulation showed that the saving in power is even higher than what was shown at the synthesis level.

The results in Tables X and XI clearly demonstrate the significance of using rational weights. Furthermore, the value of the C that was set to 11% accommodates circuit parasitics due to interconnections, and all functions operate correctly.

VII. Conclusion

It has been demonstrated that the presented approach can implement many more functions as current mode threshold logic gates with similar or less transistor count when compared to existing method. Also a significant percentage of existing threshold functions can be implemented as current mode threshold gates with approximately 60% less power dissipation, and 20% less delay when considering higher order non-integer weights in the presence of aging and circuit parasitics.

In future work, heuristic approaches will be investigated to implement higher input k-TFs with rational weights. In addition, we will investigate the impact of emerging technology on resistive devices such as memristors and spin torque transfer devices. Synthesis of complex circuit specifications will also be built upon existing 1-TF based synthesis methods.

References

[1] C. B. Dara, T. Haniotakis, and S. Tragoudas, "Low power and high speed current-mode memristor-based TLGs," in *Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Nanotechnol. Syst. (DFT)*, Oct. 2013, pp. 89–94.

[2] M. J. Avedillo, J. M. Quintana, A. Rueda, and E. Jiménez, "Low-power CMOS threshold-logic gate," *Electron. Lett.*, vol. 31, no. 25, pp. 2157–2159, 1995.

[3] J. Yang, N. Kulkarni, S. Yu, and S. Vrudhula, "Integration of threshold logic gates with RRAM devices for energy efficient and robust operation," in *Proc. IEEE/ACM Int. Symp. Nanosc. Archit. (NANOARCH)*, Jul. 2014, pp. 39–44.

[4] N. Kulkarni, J. Yang, J.-S. Seo, and S. Vrudhula, "Reducing power, leakage, and area of standard-cell ASICs using threshold logic flip-flops," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 24, no. 9, pp. 2873–2886, Sep. 2016.

[5] C. B. Dara, T. Haniotakis, and S. Tragoudas, "Delay analysis for current mode threshold logic gate designs," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 3, pp. 1063–1071, Mar. 2017.

[6] S. Bobba and I. N. Hajj, "Current-mode threshold logic gates," in *Proc. Int. Conf. Comput. Design*, Sep. 2000, pp. 235–240.

[7] T. Gowda, S. Vrudhula, N. Kulkarni, and K. Berezowski, "Identification of threshold functions and synthesis of threshold networks," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 30, no. 5, pp. 665–677, May 2011.

[8] V. Beiu, J. M. Quintana, M. J. Avedilo, and R. Andonie, "Differential implementations of threshold logic gates," in *Proc. Int. Symp. Signals, Circuits Syst. (SCS)*, vol. 2. 2003, pp. 489–492.

[9] M. Nikodem, M. A. Bawiec, and J. Biernat, "Synthesis of generalised threshold gates and multi threshold threshold gates," *INTL J. Electron. Telecommun.*, vol. 58, no. 1, pp. 49–54, 2012.

[10] R. O. Winder, "Threshold logic," Ph.D. dissertation, Dept. Math., Princeton Univ., Princeton, NJ, USA, 1962.

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

[12] M. Dertouzos, *Threshold Logic: A Synthesis Approach*. Cambridge, MA, USA: MIT Press, 1965.

[13] C. Wang and A. C. Williams, "The threshold order of a Boolean function," *Discrete Appl. Math.*, vol. 31, no. 1, pp. 51–69, 1991.

[14] A. K. Palaniswamy and S. Tragoudas, "An efficient heuristic to identify threshold logic functions," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 8, no. 3, pp. 19:1–19:17, 2012.

[15] A. Neutzling, M. G. A. Martins, R. P. Ribas, and A. I. Reis, "Synthesis of threshold logic gates to nanoelectronics," in *Proc. 26th Symp. Integr. Circuits Syst. Design (SBCCI)*, 2013, pp. 1–6.

[16] A. Neutzling, J. M. Matos, A. I. Reis, R. P. Ribas, and A. Mishchenko, "Threshold logic synthesis based on cut pruning," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des. (ICCAD)*, Nov. 2015, pp. 494–499.

[17] A. K. Palaniswamy and S. Tragoudas, "Improved threshold logic synthesis using implicant-implicit algorithms," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 10, no. 3, 2014, Art. no. 21.

[18] T. Gowda and S. Vrudhula, "Decomposition based approach for synthesis of multi-level threshold logic circuits," in *Proc. Asia South Pacific Design Autom. Conf.*, Mar. 2008, pp. 125–130.

[19] A. Neutzling, M. G. A. Martins, R. P. Ribas, and A. I. Reis, "A constructive approach for threshold logic circuit synthesis," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Jun. 2014, pp. 385–388.

[20] R. Zhang, P. Gupta, L. Zhong, and N. K. Jha, "Threshold network synthesis and optimization and its application to nanotechnologies," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 24, no. 1, pp. 107–118, Jan. 2005.

[21] 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.

[22] R. Zhang, P. Gupta, L. Zhong, and N. K. Jha, "Synthesis and optimization of threshold logic networks with application to nanotechnologies," in *Proc. Conf. Design, Autom. Test Eur. (DATE)*, Feb. 2004, pp. 904–909.

[23] A. Mishchenko, "Enumeration of irredundant circuit structures," in *Proc. Int. Workshop Logic Synth.*, 2014, pp. 1–7.

[24] J. Keane, X. Wang, D. Persaud, and C. H. Kim, "An all-in-one silicon odometer for separately monitoring HCI, BTI, and TDDB," *IEEE J. Solid-State Circuits*, vol. 45, no. 4, pp. 817–829, Apr. 2010.

[25] B. Eghbalkhah, M. Kamal, A. Afzali-Kusha, M. B. Ghaznavi-Ghoushchi, and M. Pedram, "CSAM: A clock skew-aware aging mitigation technique," *Microelectron. Rel.*, vol. 55, no. 1, pp. 282–290, 2015.

[26] S. Mahapatra, A. E. Islam, S. Deora, V. D. Maheta, K. Joshi, and M. A. Alam, "Characterization and modeling of NBTI stress, recovery, material dependence and AC degradation using R-D framework," in *Proc. IEEE Int. Symp. Phys. Failure Anal. Integr. Circuits (IPFA)*, Jul. 2011, pp. 1–7.

[27] L. Gao, F. Alibart, and D. B. Strukov, "Programmable CMOS/memristor threshold logic," *IEEE Trans. Nanotechnol.*, vol. 12, no. 2, pp. 115–119, Mar. 2013.

[28] J. Rajendran, H. Manem, R. Karri, and G. S. Rose, "An energy-efficient memristive threshold logic circuit," *IEEE Trans. Comput.*, vol. 61, no. 4, pp. 474–487, Apr. 2012.

[29] A. K. Maan, D. A. Jayadevi, and A. P. James, "A survey of memristive threshold logic circuits," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 8, pp. 1734–1746, Aug. 2017.

[30] J. M. Quintana, M. J. Avedillo, J. Nunez, and H. P. Roldan, "Operation limits for RTD-based MOBILE circuits," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 56, no. 2, pp. 350–363, Feb. 2009.

[31] S. Vrudhula, N. Kulkami, and J. Yang, "Design of threshold logic gates using emerging devices," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2015, pp. 373–376.

[32] N. Kimizuka, T. Yamamoto, T. Mogami, K. Yamaguchi, K. Imai, and T. Horiuchi, "The impact of bias temperature instability for direct-tunneling ultra-thin gate oxide on MOSFET scaling," in *IEEE Symp. VLSI Technol. Dig. Tech. Papers*, Jun. 1999, pp. 73–74.

[33] S. Khan, S. Hamdioui, H. Kukner, P. Raghavan, and F. Catthoor, "BTI impact on logical gates in nano-scale CMOS technology," in *Proc. IEEE Int. Symp. Des. Diagnostics Electron. Circuits Syst. (DDECS)*, Apr. 2012, pp. 348–353.

[34] S. N. Mozaffari, S. Tragoudas, and T. Haniotakis, "Reducing power, area, and delay of threshold logic gates considering non-integer weights," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2017, pp. 1–4.

[35] S. N. Mozaffari, S. Tragoudas, and T. Haniotakis, "A new method to identify threshold logic functions," in *Proc. Des., Autom. Test Eur. Conf. Exhib. (DATE)*, 2017, pp. 934–937.

[36] W. Wang, S. Yang, S. Bhardwaj, S. Vrudhula, F. Liu, and Y. Cao, "The impact of NBTI effect on combinational circuit: Modeling, simulation, and analysis," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 18, no. 2, pp. 173–183, Feb. 2010.

[37] (2013). *Berkeley Predictive Technology Models (PTM)*. [Online]. Available: http://ptm.asu.edu/modelcard/LP/45nm_LP.pm

[38] S. Gupta, J. Dworak, D. Engels, and A. Crouch, "Mitigating simple power analysis attacks on LSIB key logic," in *Proc. IEEE North Atlantic Test Workshop (NATW)*, May 2017, pp. 1–6.

[39] S. N. Mozaffari and A. Afzali-Kusha, "Statistical model for subthreshold current considering process variations," in *Proc. 2nd Asia Symp. Quality Electron. Des. (ASQED)*, 2010, pp. 356–360.

Seyed Nima Mozaffari received the M.S. degree from the School of Electrical and Computer Engineering, University of Tehran, in 2010. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, Southern Illinois University at Carbondale, Carbondale. He is a member of the Electronic Design and Test Automation Laboratory, where he is involved in research for design and test of digital circuits and systems.

His research interests include very large-scale integration design and test automation, design for testability, artificial intelligence, neural network implementations, emerging technologies, nonvolatile memories, and threshold logic gates. He is a reviewer in the IEEE journals and conferences.

Spyros Tragoudas (SM'87) received the B.S. degree in computer engineering from the University of Patras in 1986 and the M.S. and Ph.D. degrees in computer science from The University of Texas, Dallas, in 1988 and 1991, respectively. He has held faculty member appointments at the Electrical and Computer Engineering Department, The University of Arizona, and the Computer Science Department, SIUC. He is currently the Chairman and a Professor with the Electrical and Computer Engineering Department, Southern Illinois University at Carbondale (SIUC), and the Director of the National Science Foundation (NSF) Industry University Cooperative Research Center on Embedded Systems with the SIUC site.

He has been supported from industry and federal agencies, including the NSF and the U.S. Navy. His current research interests include VLSI design and test automation and embedded systems. He has authored over 80 journal papers and over 160 articles in peer-reviewed conference proceedings in these areas. He was a recipient of the ICCD'94, the ICCD'97, and the ISQED'01 Outstanding Paper Awards for research in VLSI testing. He has served on the Editorial Board of several journals, including the IEEE TRANSACTIONS ON COMPUTERS, the *VLSI Design Journal*, and the *Journal of Electrical and Computer Engineering*. He was the Program Chair of DFTS'09 and the General Chair of DFTS'10. He is a program committee member for many conferences.

Themistoklis Haniotakis (M'87) received the B.S. degree in physics and the Ph.D. degree in informatics from the University of Athens, Greece. His Ph.D. thesis was on the area of self-checking circuits. He has held a faculty member appointment at the University of Patras, Greece. He is currently a Faculty Member with the Electrical and Computer Engineering Department, Southern Illinois University at Carbondale.

He has 20 journal and over 50 Conference publications. His interests include VLSI design, fault-tolerant computing, VLSI testing and design for testability, and RF IC design and test. He received the Best Paper Award ISQED from the IEEE. He has been a reviewer in the IEEE journals and conferences and a member of various program committees.