A new method to identify threshold logic functions

Seyed Nima Mozaffari, Spyros Tragoudas, Themistoklis Haniotakis
Department of Electrical and Computer Engineering
Southern Illinois University
Carbondale, USA
{nima.mozaffari, spyros, haniotak}@siu.edu

***Abstract*—An Integer Linear Programming based method to identify current mode threshold logic functions is presented. The approach minimizes the transistor count and benefits from a generalized definition of threshold logic functions. Process variations 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 existing threshold functions is reduced.**

***Keywords*—threshold logic gate; synthesis; threshold function; current mode design; weight assignment;**

1. Introduction

Threshold Logic Gate is a promising candidate for the future digital circuits. Using TGs, circuits may be implemented with fewer number of logic gates, and hence with 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). A Boolean Threshold logic Function (TF) requires a weight, one per input variable. An n-input TF is realized using a sum of active weights where each active weight corresponds to a different input variable. For each input pattern, the function evaluates to one only when the sum of the active weights is greater than (or equal) to a weight value called the threshold weight [6]. In this paper, such a function is called a 1^{st}-order TF and will be denoted as a 1-TF.

An n-input 1-TF $f(x_1, x_2, \dots, x_n)$ is formally defined as [6]:

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

where x_i, $i = 1, \dots, n$, are binary input variables, w_i is the weight corresponding to the i^{th} input, and w_T is the threshold weight (threshold value). In (1) the term $w_i \cdot x_i$ indicates that the weight w_i is active when $x_i = 1$. The 1-TF $f(x_1, x_2, \dots, x_n)$ is also denoted by the set of threshold and input weights $[w_1, w_2, \dots, w_n; w_T]$. For implementation purposes, the weights are assumed to be integers. A small fraction of binary functions are 1-TFs [6] and can be implemented as a single TG.

In a 2-TF, the weights are more than n and for each input pattern a weight can be activated either by a single input (as in 1-TF) and it is called a 1-weight, or by a pair of inputs and it is called a 2-weight. Likewise, k-TFs are defined when $k > 2$. However, the paper focuses on 2-TF due to space limitations. The 2-TF formulation for an n-input function is [7]

$$f(x_1, x_2, \cdots, x_n) = \begin{cases} 1 & if \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} w_{i,j} x_i x_j \geq w_T \\ 0 & otherwise \end{cases}, \quad (2)$$

where x_i, $i = 1, \dots, n$, are binary input variables, w_i is the 1-weight corresponding to the i^{th} input, $w_{i,j}$ is the 2-weight corresponding to the pair of i^{th} and j^{th} inputs, and w_T is the threshold weight (threshold value). In (2) the term $w_i \cdot x_i$ indicates that the weight w_i is active when $x_i = 1$ and the term $w_{i,j} \cdot x_i \cdot x_j$ indicates that the weight $w_{i,j}$ is active when $x_i = 1$ and $x_j = 1$. A 2-TF is also denoted by the set of weights $[w_1, w_2, \dots, w_n, w_{1,2}, \dots, w_{i,j}, \dots, w_{(n-1),n}; w_T]$.

Several interesting circuit concepts have been proposed recently in [1, 4-5, 8-13], among others, for implementing TGs and synthesizing circuits using TGs. This paper focuses on the Current-mode TGs (CTGs) which is a popular PMOS- and NMOS-based implementation. (See [1, 9], among others.) however, the method is applicable to all existing implementations.

An automation framework is introduced to identify efficiently 2-TFs that can be implemented as current mode logic gates. A new method is proposed to implement CTG with minimum hardware overhead (transistor count) considering the definition in (2). The proposed identification method is an extension of the Integer Linear Programming (ILP) formulation in [6]. The objective function of the ILP is modified in order to control the number of 2-weights and minimize the area of the CTG. It will be shown that many more functions can be implemented as CTG using similar hardware overhead to that of traditional CTGs. Also, many 1-TFs are implemented as 2-TF with reduced hardware overhead. Process variations are also taken into consideration.

This paper is organized as follows. Section 2 shows how to implement current mode logic 2-TF with minimum hardware overhead in the presence of process variations. An ILP-based framework is also proposed to identify 2-TFs efficiently. Section 3 provides experimental results. Section 4 concludes the paper.

2. The Proposed Method

The CTG implementation in [9] implements a 1-TF with NMOS (or PMOS) transistors connected in parallel. Let X denote the width of a minimum size transistor which implements the unit 1-weight. Each transistor implements a 1-weight w and its width is $w \cdot X$, and the gate of the transistor is connected to an input. The gate of the NMOS transistor for the threshold is connected to the power supply (it is active for all input patterns). The length

978-3-9815370-8-6/17/$31.00 ©2017 IEEE

of all transistors is the same and is determined by the used technology. We call such gates as 1-CTG.

A 2-weight is implemented with 2 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 CTG inputs. The size of each transistor in a sub-circuit that implements a 2-weight with weight value w is set to $2 \cdot w \cdot X$ to keep the active current of a unit 2-weight equal to the active current of a unit 1-weight. Thus, the total area of a 2-weight sub-circuit is 4 times more than the area of a 1-weight that implements the same weight. This area ratio is called the penalty factor and is taken into consideration to force the ILP-solver to find the minimum possible transistor count which is the total number of unit size transistors that implements weights. (In CTG, the area of a transistor with twice the unit width is practically the same as two minimum size transistor.) Such gates are called 2-CTGs.

An ILP formulation is presented to implement a Unate Function (UF) as a 2-CTG, as in (2) with minimum transistor count. A function is a UF if it is either positive or negative in every variable [6]. A function is positive (negative) in variable x_i if the variable $\bar{x}_i$ (x_i) does not appear in the expression of the function. The proposed ILP is an improvement of [7].

The ILP contains $2^n + n$ constraints with $n + n' + 1$ variables, where $n' = \binom{n}{2}$ is the total number of 2-weights. there is a constraint per input pattern to satisfy the functionality, and n constraints that bind the range of each weight. Once a UF f is given, the Modified Chow's parameters [6] of all inputs and pairs of inputs determine the negative weights (including 1-weights and 2-weights). To form an efficient ILP, every input x_i (or product of two inputs $x_i \cdot x_j$), that activates a 1-weight (or 2-weight) with a negative Modified Chow's parameter must be complemented. A 1-weight with negative Modified Chow's parameter is activated as in 1-TF. A 2-weight with negative Modified Chow's parameter will be activated when at least one of x_i and x_j is set to 0. The ILP with the appropriate activation signals determines whether function f is a 2-TF.

Furthermore, the penalty factors appear as the coefficient of weights in the objective function of the ILP. The following objective function minimizes the CTG transistor count:

$$w_T + 1 \cdot \sum_{i=1}^{n} w_i + 4 \cdot \sum_{i=1}^{n-1} \sum_{j=2}^{n} w_{i,j} \qquad (3)$$

The weight configuration will then be assigned using the following property which is called the negation property: The negation of variable x_i, $(x_i \rightarrow \bar{x}_i)$, changes the weight configuration to $[w_1, w_2, \dots, -w_i, \dots, w_n; w_T - w_i]$. The following illustrates the ILP-based method to identify a 2-TF.

Example 1: Consider a 4-input UF $F_1 = x_1' + x_3x_2' + x_3x_4'$ 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 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 1. THE TRUTH TABLE AND ILP CONSTRAINTS FOR F_1 CONSIDERING ALL INPUTS AND PAIRS OF INPUTS ARE COMPLEMENTED EXCEPT x_3.

Truth Table			Inequalities
Input Pattern $(x_1x_2x_3x_4)$		F_1	
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:			$w_T + 1 \cdot \sum_{i=1}^{4} w_i + 4 \cdot \sum_{i=1}^{3} \sum_{j=2}^{4} w_{i,j}$

Table 1 lists the inequalities of F_1 based on the 2-TF definition and by considering positive weights. For an input 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 1, an optimum solution is $w = [3,0,2,0,0,0,0,1,0,1; 2]$. Using the negation property, $F_1 = [-3,0,2,0,0,0,0, -1,0, -1; -2]$. □

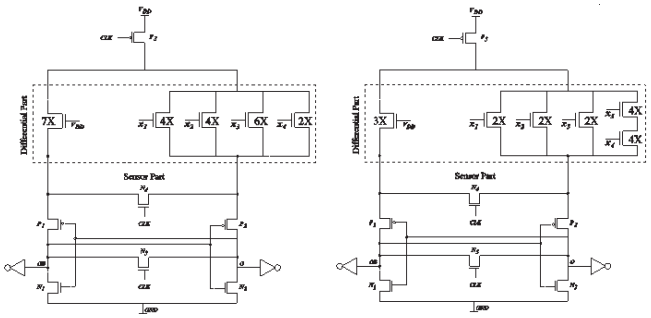
The following show that some 1-TFs can be implemented as proposed 2-CTG with lower cost transistor than the respective 1-CTG.

Example 2: Consider the 4-variable function $F_2 = x_4x_3 + x_3x_2 + x_3x_1 + x_2x_1$. 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 [6]. 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 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. Thus, the total area of the 2-CTG reduces to $X \cdot (2 + 2 + 2) + X \cdot 4 \cdot (2) + X \cdot (3) = 17 \cdot X$.

Fig. 1 shows the 1-CTG [9] and proposed 2-CTG implementations of F_2 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 correlation between the sign of the Modified Chow's parameters and the sign of weights only holds for UFs. Non-unate functions are called Binate Functions (BFs). The ILP for a BF works as in [7] and contains $2^n + 3n$ constraints with $3(n + n' + 1)$ variables, where $n' = \binom{n}{2}$ is the total number of 2-



(a) (b)

Fig. 1. The CTG implementation for function $F_2 = x_4x_3 + x_3x_2 + x_3x_1 + x_2x_1$ with (a) 1-CTG as 1-TF [9] (b) proposed 2-CTG as 2-TF.

weights. Each weight can be either positive or negative. The objective function is to minimize quantity $|w_T| + 1 \cdot \sum_{i=1}^{n}|w_i| + 4 \cdot \sum_{i=1}^{n-1}\sum_{j=2}^{n}|w_{i,j}|$, 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:

$$\begin{cases} 0 \le {w_x}^+ \le U \cdot y_x \\ 0 \le {w_x}^- \le U \cdot (1 - y_x) \\ y_x \in \{0,1\} \end{cases} \quad (4)$$

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

$$ {w_T}^+ + {w_T}^- + \sum_{i=1}^{n}({w_i}^+ + {w_i}^-) + 4 \cdot \sum_{i=1}^{n-1}\sum_{j=2}^{n}({w_{i,j}}^+ + {w_{i,j}}^-) \quad (5)$$

In a TG implementation, any variation in the assigned weights due to process variations may impact its functionality. Let constant value C denote the maximum weight deviation which can be obtained with SPICE simulations given the technology. The pattern dependent inequalities of the ILP are rewritten as $\sum_i (w_i - C \cdot w_i) \cdot x_i \ge w_T + C \cdot w_T$ 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 changes do not reduce the number of 2-TFs that can be implemented as TGs but may change the total sum of weights. For example, the weight configuration in Example 1 considering $C = 5\%$ becomes $w = [-6,0,4,0,0,0,0,-2,0,-2;-5]$.

3. Experimental Results

The proposed ILP-based approaches have been developed in the C++ language on an Intel Xenon 2.4GHz with 8GB memory. To evaluate the contribution, we examined up to eighteen-input non-scalable functions. (An n-input non-scalable function requires no less than n non-empty levels of variables in the BDD representation for some ordering of the variables [6].) The weight values were integers in the range [-25, 25].

Table 2 reveals the number of TFs that are implemented as 2-CTGs but have no more transistor count, the number of unit size transistors that implement weights, than the maximum transistor count of any examined non-scalable n-input 1-TF implemented as 1-CTG. Results for up to eighteen inputs were derived using the robust ILP-based approach considering weight variations. Penalty factors 1 and 4 are also applied to the objective function of ILP for 1-weights and 2-weights, respectively. Column one of Table 2 lists the number of inputs. the second column lists the number of 1-TFs that can be implemented as a single 1-CTG. For functions with greater than four inputs, the entries in Table 2 are obtained by sampling randomly 20,000 functions. The 2^n bits output vector of an n-input function is filled with one bit values at positions determined by randomly selecting an integer mod 2^n, and the number of ones in the function must obey 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 consider non-scalable functions, and when a function is generated we apply 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 third column shows the number of 2-TFs that do not have 2-CTG transistor count (area) higher than any of 1-TFs in column two, and the fourth column their increase on the number of TFs which were determined by the number of 2-TFs over the number of 1-TFs. Columns four to seven show similar results when considering 6% and 12% weight variations. These values obtained by SPICE simulations in corner cases using 45nm technology [14] while considering 5% and 10% variations in transistor width and length [15]. These results show that a significant percentage of functions can be implemented in current mode gates using transistor count similar to that for the very few 1-TF functions that implemented as 1-CTG, and in the presence of process variations.

Table 3 shows that many 1-TF functions can be identified as 2-TF with less area in the CTG implementation. The first row lists the values of n, the number of input variables. The second row lists the number of non-scalable 1-TF that were identified for each value of n. The third row shows the number of 1-TF which were also identified as 2-TF with less area in the CTG

Table 2. The Number of 2-CTG without Hardware Overhead.

n	# of 1-CTG	# of 2-CTG (0% variations)	Increase ratio	# of 2-CTG (6% variations)	Increase ratio	# of 2-CTG (12% variations)	Increase ratio
1	2	2	1	2	1	2	1
2	8	8	1	8	1	8	1
3	72	72	1	72	1	72	1
4	1536	2742	1.8	2678	1.75	2566	1.68
5	99	1346	13.6	1323	13.3	1217	12.3
6	120	1859	15.5	1676	13.9	1339	11.2
7	84	1804	21.5	1721	20.4	1505	17.9
8	76	2216	29.1	2103	27.6	1732	22.7
9	163	1462	8.9	1385	8.4	1204	7.3
10	137	2047	14.9	2023	14.7	1733	12.6
11	85	1887	22.2	1655	19.4	1221	14.3
12	85	1086	12.7	1046	12.2	980	11.4
13	97	1815	18.7	1422	14.6	1406	14.4
14	64	1332	20.8	1179	18.4	1096	19.1
15	83	1492	17.9	1370	16.4	1124	13.5
16	72	1123	15.6	1011	14.0	797	11.0
17	63	1247	19.8	1104	17.5	826	13.1
18	60	1098	18.3	931	15.5	603	10.0

TABLE 3. THE NUMBER OF 1-TFS WITH REDUCED AREA AS 2-TF.

TF Type	Number of Inputs															
	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
# 1-TF	72	1536	99	120	84	76	163	137	85	85	97	64	83	72	63	60
# 2-TF	0	444	65	98	75	68	142	108	70	76	73	57	48	64	52	47
% optimized	0 %	29 %	65 %	82 %	90 %	90 %	87 %	79 %	82 %	79 %	76 %	90 %	58 %	89 %	83 %	78 %

implementation, and the fourth row lists the percentage of optimized 2-TFs over the total number of 1-TFs. It is very important to observe that the percentage of these functions increases as n increases. The majority of 1-TF can be implemented with less area using 2-CTG implementation method for 2-TF for $n \geq 4$.

Table 4 reports the average execution time that was required to determine whether an examined non-scalable function was a 1-TF, a UF 2-TF and a BF 2-TF. These results show the average execution time of the ILP-based approaches when n, the number of input variables, is no more than eighteen.

The following compares the delay and power dissipation of the 1-CTG implementation of randomly selected four-input 1-TFs to the proposed 2-CTG implementation of randomly selected four-input 2-TFs. We considered 1-TFs and 2-TFs whose CTG area (transistor count) were similar. More specifically, we considered three different transistor count groups, and from each group we experimented with ten 1-TFs and ten 2-TFs. Average results are reported in Table 5.

All functions were implemented using [9]. SPICE simulation took place for each function using the Berkeley Predictive Technology Models (PTM) for 45nm CMOS transistors [14]. 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 [3].

Table 5 shows the average delay and power of the 1-TFs and 2-TFs implemented as 1-CTGs and 2-CTGs, respectively. The delay of each gate is the maximum delay among all applied input combinations. Simulation results showed a very small increase (less than 15%) in the delay for 2-CTGs due to the sub-circuits that implements 2-weights which increase the capacitance of the interior nodes. Also, the average power of the 2-CTGs is almost the same as for the 1-CTGs.

TABLE 4. AVERAGE EXECUTION TIME (ms) PER FUNCTION FOR 1-TF, UF 2-TF, AND BF 2-TF FOR DIFFERENT VALUES OF n.

TF Type	Number of Inputs												
	6	7	8	9	10	11	12	13	14	15	16	17	18
1-TF	50	54	72	115	250	490	1108	2e3	35e2	7e3	13e3	3e4	9e4
UF 2-TF	54	60	80	125	275	508	1200	23e2	4e3	8e3	15e3	4e4	11e4
BF 2-TF	63	96	150	290	540	1e3	2e3	5e3	9e3	15e3	-	-	-

TABLE 5. AVERAGE DELAY (ns) AND AVERAGE POWER (μW) FOR 1-CTGS AND 2-CTGS WITH SIMILAR TRANSISTOR COUNT CONSIDERING THREE DIFFERENT GROUPS OF FOUR-INPUT TFS.

CTG Type	Transistor Count					
	17		21		25	
	delay	power	delay	power	delay	power
1-CTG	135	1.60	153	1.76	175	1.98
2-CTG	140	1.50	170	1.80	198	2.06

4. CONCLUSIONS

It has been demonstrated that the presented approach can implement many more functions as current mode gates with less area, similar power dissipation, and slightly increased delay when compared to existing method. In future works, heuristic approaches can be used to implement 2-TFs with higher inputs by extending a work.

ACKNOWLEDGMENT

This research has been supported in part by grants NSF IIP 1432026, and NSF IIP 1361847 from the NSF I/UCRC for Embedded Systems at SIUC. 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.

REFERENCES

[1] C. B. Dara, T. Haniotakis and S. Tragoudas, "Delay Analysis for Current Mode Threshold Logic Gate Designs," in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* , vol. PP, no. 99, pp.1-9, 2016.

[2] C. B Dara, T. Haniotakis and S. Tragoudas, "Low power and high speed current-mode memristor-based TLGs," *IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, pp.89-94, 2013.

[3] M. J. Avedillo, J. M. Quintana, A. Rueda and E. Jimenez, "Low-power CMOS threshold-logic gate," *in Electronics Letters* , vol. 31, no. 25, pp. 2157-2159, 1995.

[4] J. Yang, N. Kulkarni, S. Yu and S. Vrudhula, "Integration of threshold logic gates with RRAM devices for energy efficient and robust operation," in *IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH)*, pp.39-44, 2014.

[5] N. Kulkarni, J. Yang, J. S. Seo and S. Vrudhula, "Reducing Power, Leakage, and Area of Standard-Cell ASICs Using Threshold Logic Flip-Flops," in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. PP, no.99, pp.1-14, 2016.

[6] S. Muroga, "Threshold Logic and its Applications," *John Wiley, New York*, 1971.

[7] C. Wang and A.C. Williams, "The threshold order of a Boolean function," in *Discrete Applied Mathematics*, vol. 31, no. 1, pp. 51-69, 1991.

[8] V. Beiu, J. M. Quintana, M. J. Avedilo and R. Andonie, "Differential implementations of threshold logic gates," in *proceedings of the International Symposium on Signals, Circuits and Systems 2 (SCS)*, vol.2, pp.489-492, 2003.

[9] S. Bobba and I. N. Hajj, "Current-mode threshold logic gates," in *Proceedings of the International Conference on Computer Design*, pp.235-240, 2000.

[10] J. M. Quintana, M. J. Avedillo, J. Nunez and H. P. Roldan, "Operation Limits for RTD-Based MOBILE Circuits," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol.56, no.2, pp.350-363, 2009.

[11] S. Vrudhula, N. Kulkami and J. Yang, "Design of threshold logic gates using emerging devices," *IEEE International Symposium on Circuits and Systems (ISCAS)*, pp.373-376, 2015.

[12] L. Gao, F. Alibart and D. B. Strukov, "Programmable CMOS/ Memristor Threshold Logic," *IEEE Transactions on Nanotechnology*, pp.115-119, 2013.

[13] J. Rajendran, H. Manem, R. Karri and G. S. Rose, "An Energy-Efficient Memristive Threshold Logic Circuit," in *IEEE Transactions on Computers*, vol. 61, no. 4, pp. 474-487, 2012.

[14] Berkeley Predictive Technology Models (PTM), http://ptm.asu.edu/modelcard/LP/45nm_LP.pm, 2013.

[15] M. Jafari, M. Imani and M. Fathipour, "Analysis of Power Gating in Different Hierarchical Levels of 2MB Caches, Considering Variation," in *International Journal of Electronic, Taylor and Francis*, vol. 105, no. 2, 2014.