A Rule-Based Approach for Minimizing Power Dissipation of Digital Circuits

Subrata Das*, Parthasarathi Dasgupta†, Petr Fiser‡, Sudip Ghosh§ and Debesh Kumar Das*
*Department of Computer Science & Engineering, Jadavpur University, Kolkata, India
Email: dsubrata.mt@gmail.com, debeshd@hotmail.com
†MIS Group, Indian Institute of management Calcutta, Kolkata, India
Email: partha@iimcal.ac.in
‡Faculty of Information Technology, Czech Technical University in Prague, Czech Republic
Email:fiserp@fit.cvut.cz
§School of VLSI Technology, IIEST, Shibpur, Howrah, India
Email:sudipghosh2005@gmail.com

***Abstract*—Minimization of power dissipation of VLSI circuits is one of the major concerns of recent digital circuit design primarily due to the ever decreasing feature sizes of circuits, higher clock frequencies and larger die sizes. The primary contributors to power dissipation in digital circuits include leakage power, short-circuit power and switching power. Of these, power dissipation due to the circuit switching activity constitutes the major component. As such, an effective mechanism to minimize the power loss in such cases often involves the minimization of the switching activity. In this paper, we propose an intelligent rule-based algorithm for reducing the switching activity of the digital circuits at logic optimization stage. The proposed algorithm is empirically tested for several standard digital circuits with Synopsys EDA tool and the results obtained are quite encouraging.**

***Index Terms*—Switching activity, low-power VLSI circuits, CMOS, power dissipation, dynamic power, logic optimization.**

I. Introduction

Traditionally, the major concerns of VLSI designers include minimization of the chip area, enhancement of performance, testability, reduction of manufacturing cost and the improvement of reliability. With increasing use of portable devices and wireless communication systems, the reduction of energy consumption and hence the reduction of power dissipation and optimization of chip temperature are current issues in recent VLSI design [1]. Power dissipated by a digital system increases the temperature of the chip and affects battery life of the digital devices [2]. Aggressive device scaling also causes excessive increase in power per unit area of the chip. As such, heat generation and its removal from a chip is a matter of serious concern [3]. In $CMOS$ circuits the three primary sources of power dissipation are [4]

1) The switching activity occurs due to logic transitions. When the nodes of a digital circuit make transition back and forth between two logic levels, parasitic capacitances are charged and discharged. Consequently current flows through the channel resistance of the transistors, and electrical energy is converted into heat [4].
2) The short-circuit current that flows from supply to ground when both the p-subnetwork and n-subnetwork of a $CMOS$ gate conduct [4].
3) The leakage current [4] caused by substrate injection at p-n junctions and sub-threshold effects determined by the fabrication technology.

The first two sources of power dissipation are known as *dynamic power dissipation* and the third one constitutes the *static power dissipation*. In the present-day technology about 80% of the total power loss occurs due to the switching activity [4]. Thus, minimization of the power dissipation of VLSI circuits necessitates minimization of the dynamic power and hence minimization of the switching activity. Minimization of switching activity can be done at the logic optimization stage. However, the focus of earlier works in logic optimization is primarily on reduction of the number of appearances of literals, minimum number of literals in a Sum of Products (SOP) or Product of Sums (POS) expression and minimum number of terms in a SOP expression [5].
In this paper, we propose an algorithm to obtain for a given input logic expression, an equivalent logic expression with minimized switching activity.

II. Literature Review

Minimization of power consumption of CMOS digital circuits is studied in the past, considering all levels of the design such as physical, circuit and logic level [6], [7]. In digital CMOS circuits the measure of power dissipation is the circuit switching activity or the average number of transitions. Minimization of the average number of transitions of CMOS digital circuits nodes is discussed in [8]. The work in [9] provides an interesting repository of recent techniques of power modeling and low-power design based on high-level synthesis. The evaluation and the reduction of the switching activity in combinational logic circuits considering both the transitions $1 \rightarrow 0$ and $0 \rightarrow 1$ at any output node is proposed in [10]. In order to satisfy the classical probabilistic approach that limits the maximum value of the switching activity to 1, the definition of the switching activity as proposed in [10]

was customized in [11]. An algorithmic approach at the gate level using Karnaugh maps for reducing the switching activity in combinational logic circuits is presented in [12]. However, the use of Karnaugh maps restricts the number of variables to around 6. Moreover, for the method proposed in [12], the switching activity can be minimized only for some specific switching functions. In [13] the authors proposed a method to estimate the switching activity using a variable delay model. The work of [14] has discussed the system level dynamic power management in chip multiprocessor (CMP) architectures. [15] proposed an algorithm to minimize logic functions with reduced area and interconnects that will improve the circuit performance. Pre-computation-based optimization for low power that computes the output logic values of the circuit in one clock cycle before they are computed, was discussed in [16].

III. Preliminaries

A definition of the switching activity based on the classical probabilistic approach is given in [11]. For a logic expression of a switching function for an output node i, let $\mid N_i \mid$ and $\mid X_i \mid$ represent the number of 1's and the number of 0's. The probabilities of occurrence of a 0 and a 1 respectively at the output node i are given by the following equations:

$$P_0 = \frac{\mid X_i \mid}{\mid N_i \mid + \mid X_i \mid} \tag{1}$$

$$P_1 = \frac{\mid N_i \mid}{\mid N_i \mid + \mid X_i \mid} \tag{2}$$

In the computation of the switching activity, it is assumed that the distribution of 0's and 1's at the primary inputs (PIs) is uniform.

Definition 1. For a given node of a circuit the probability of transition either from 0 to 1 or from 1 to 0 is known as switching activity of that node.

Thus, the switching activity of node i is given by the composite probability

$$SA = P_0 \times P_1 = \frac{\mid N_i \mid \times \mid X_i \mid}{(\mid N_i \mid + \mid X_i \mid)^2} \tag{3}$$

As already mentioned, power dissipation in digital circuits can be reduced by minimizing their total switching activity (TSA).

It can be easily shown that the switching activity is maximum when the number of 1's ($\mid N_i \mid$) and the number of 0's ($\mid X_i \mid$) in the output column of the truth table are equal and the switching activity is minimum when the difference between the number of zeroes and the number of ones in the output column in the truth table is maximum.

A. Calculation of Switching Activities of Logic Gates

To calculate the switching activity of a logic circuit, it is important to determine the switching activity for the constituent logic gates. Based on the classical probabilistic definition of the switching activity [11], we can easily calculate the switching activity of the basic gates. For an AND, OR, $NAND$ and NOR gate with n inputs, the output is 0 or 1 for exactly one input combination (input vector). Hence the value of $\mid N_i \mid$ or $\mid X_i \mid$ is 1 or $2^n - 1$. Hence $P_0 = \frac{1}{2^n}$ or $\frac{2^n-1}{2^n}$, respectively, and the corresponding $P_1 = \frac{2^n-1}{2^n}$ or $\frac{1}{2^n}$. Thus, the switching activity is given by $\frac{2^n-1}{2^{2n}}$. Hence, as the number of inputs to the above mentioned logic gates increases, the switching activity of these gates decreases.

It is clear to see that the switching activity for the NOT gate is maximum and of value $\frac{1}{4}$. The switching activity for XOR and $XNOR$ gates is independent of the number of inputs to the gate and is equal to $\frac{1}{4}$.

B. Calculation of Switching Activity for a Logic Expression

Without loss of generality, we assume the logic gates to have at most two inputs. Computation of the switching activity for a logic expression is illustrated by an example. Consider a logic expression $f(a,b,c,d) = a\bar{b}c + \overline{ab} + \overline{c+d}$. Consider Figure 1 and the following logic expressions, $f_1 = ac$, $f_2 = \bar{b}$, $f_3 = \overline{a.b}$, $f_4 = \overline{c+d}$, $f_5 = a\bar{b}c$, $f_6 = a\bar{b}c + \overline{ab} = f_3 + f_5$, $f = f_6 + \overline{c+d} = f_6 + f_4$. The switching activity for f_5 is $\frac{7}{64}$. This is due to the fact that the output is 1 only for the input vector 101. As such, the number of 1's and 0's in the output column are 1 and 7 respectively. For the implementation of $a\bar{b}c$, a NOT gate is required for $f_2 = \bar{b}$, having $SA = \frac{1}{4}$, an AND gate is required for $f_1 = ac$ with $SA = \frac{3}{16}$. The outputs of these NOT gate and AND gate are inputs to a second AND gate having $SA = \frac{7}{64}$. f_6 can be represented in sum of minterms as $\sum 0,1,2,3,4,5$. Hence the switching activity for f_6 is $\frac{12}{64}$ (since $\mid N_i \mid = 6$, $\mid X_i \mid = 2$). The total switching activity for f_1, f_2, f_5 and f_6 is thus $\frac{1}{4}+\frac{3}{16}+\frac{7}{64}+\frac{12}{64}$. Clearly, the switching activities for $f_3 = \overline{ab}$ and $f_4 = \overline{c+d}$ are $\frac{3}{16}$ each. When represented as a sum of minterms, the function is given by
$f = \sum 0,1,2,3,4,5,6,7,8,9,10,11,12$ and the switching activity is $\frac{39}{256}$. (since $\mid N_i \mid = 13$, $\mid X_i \mid = 3$). Thus, the total switching activity of the circuit is $\frac{1}{4} + \frac{3}{16} + \frac{3}{16} + \frac{3}{16} + \frac{7}{64} + \frac{12}{64} + \frac{39}{256} = \frac{323}{256}$ (Figure 1). Signal probability can also

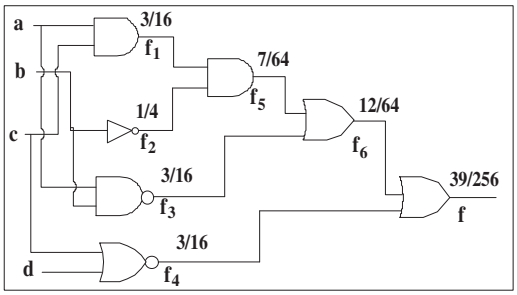

Fig. 1. Circuit schematic and switching activity for $a\bar{b}c + \overline{ab} + \overline{c+d}$

be used to estimate the switching activity. Figure 2 shows the propagation of signal probabilities through the gates. Here it is assumed that at the PIs the signal probabilities are equal. Hence in Figure 1, $P(a) = P(b) = P(c) = P(d) = \frac{1}{2}$. The Algorithm, Compute Signal Probabilities [4], [17], can

be used to estimate signal probabilities at the internal nodes of a logic circuit. From Figure 1, using this algorithm and propagation of signal probabilities through gates, we can obtain $P1 = P(a).P(c) = \frac{1}{4}$, $SA = \frac{3}{16}$. Similarly, the switching activities for nodes f_2 to f_6 and f can be computed.

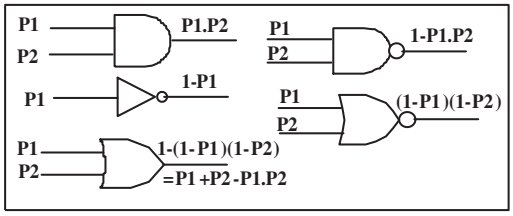

Fig. 2. Propagation of probabilities through gates

The controllability/observability concepts as described in [19] can also be used to calculate the switching activity. Unfortunately, both approaches give inaccurate results in presence of reconvergent paths. The approach presented in [20] tries to partially solve this problem.

IV. Motivation of the work

In this paper, we attempt to design a generalized method to minimize the switching activity for logic expressions. The proposed method accepts any SOP / POS expression as input and transforms it into a functionally equivalent multi-level expression with minimized switching activity. Figure 3 illustrates a motivating example. Consider the logic expression of a Full Subtractor with b_{in}, x, y as input variables and b_{out} and $diff$ as output variables. Here only the b_{out} is of interest. The logic expressions corresponding to b_{out} and $diff$ are respectively given by $(\overline{x}y + \overline{x}b_{in} + yb_{in})$ and $(x \oplus y \oplus b_{in})$. With the discussions in next few sections, we will find that the total switching activity of this logic expression will be $\frac{83}{64}$. But the logic expression for borrow can also be written as $(x(y + b_{in}) \oplus (y + b_{in})) + yb_{in}$ and the switching activity for this expression will be $\frac{70}{64}$.

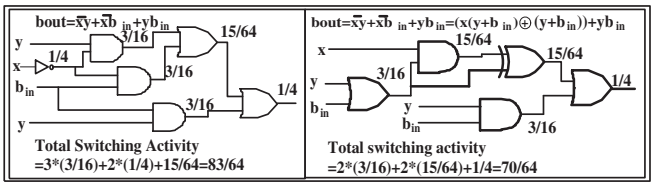

Fig. 3. Reduction of switching activity of Full Subtractor

A Rule-based systems for logic synthesis through local transformations has been proposed in [18]. In our paper different rules are used to minimize the switching activity.

V. Design for Minimal Switching Activity

Based on the discussions in Section III-A and Section III-B, it is observed that the switching activity of a circuit realization depends on its implementation, which can be described by the corresponding logic expression. Here, we consider a straightforward realization of the expression.

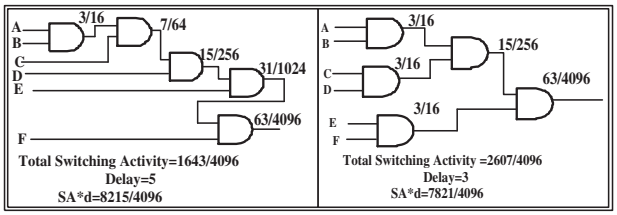

Fig. 4. Comparison of switching activity between ((((A*B)*C)*D)*E)*F and ((A*B)*(C*D))*(E*F)

TABLE I
Switching Activity (SA) and Delay (D) Product

Tree Structure				*Chain Structure*		
PI	SA	d	$SA \times d$	SA	d	$SA \times d$
2	0.1875	1	0.1875	0.1875	1	0.1875
3	0.2461	2	0.4922	0.2461	2	0.4922
4	0.4336	2	0.8672	0.3555	3	1.0664
5	0.4639	3	1.3916	0.3857	4	1.5430
6	0.6365	3	1.9094	0.4011	5	2.0056
7	0.7382	3	2.2147	0.4089	6	2.4532
8	0.8711	3	2.6132	0.4128	7	2.8894
9	0.8884	4	3.5536	0.4147	8	3.3177
10	1.0596	4	4.2382	0.4157	9	3.7412

Lemma 1. *If a product (or sum) term contains more than two literals* $(e.g.\ f = x_{n-1} * x_{n-2} * x_{n-3} * x_{n-4} * \ldots * x_1 * x_0)$*, the function can be implemented as* $((x_{n-1} * x_{n-2}) * x_{n-3}) * \ldots) * x_1 * x_0)$ *(chain representation) instead of* $((x_{n-1} * x_{n-2}) * (x_{n-3} * x_{n-4}) * \ldots * (x_1 * x_0))$ *if* n *is even or* $((x_{n-1} * x_{n-2}) * (x_{n-3} * x_{n-4}) * \ldots * (x_2 * x_1) * x_0)$ *(tree representation) if* n *is odd. The switching activity is less in the chain representation [7].*

Figure 4 shows the implementation of a 6 variable (A, B, C, D, E, F) product term. The first implementation switching activity ($\frac{1643}{4096}$) is less in comparison to the second one ($\frac{2607}{4096}$). A similar type of operation can be done for a sum term as well.

Due to delay in different gates there may be power loss due to glitches. The width of the glitch depends on the delay of logic gates and interconnections [4]. It is clear that glitches can occur in a chain structure and in the tree structure the occurrence of glitches is much less. Hence, minimization of the product of the switching activity and the delay may be a good design parameter. Table I shows the product of the switching activity and the delay for AND gate in the Tree structure and the Chain structure. Here, it is assumed that for each gate the delay is 1.

Observation 1. *For AND and OR gates, it can be shown that if the switching activity and the delay product is considered, then for* $n \leq 9$ *the tree structure is better, and if* $n > 9$ *the chain structure is better. Here* n *is the number of primary inputs.*

For XOR and XNOR gates the tree structure is better since the switching activities for these gates are independent of the number of inputs.

Lemma 2. *Let an n variables product term contains an odd number of complemented literals, say, x_0 to x_k literals are in complemented form and x_{k+1} to x_{n-1} literals are in prime form, k is even, i.e., $(\prod_{i=0}^{i=k} \overline{x_i}).(\prod_{j=k+1}^{j=n-1} x_j)$ then the following transformations will reduce the switching activity:*
$(\prod_{i=0}^{i=k} \overline{x_i}).(\prod_{j=k+1}^{j=n-1} x_j)$
$=(\prod_{i=0}^{i=k-1} \overline{x_i}).(\prod_{j=k+1}^{j=n-1} x_j).\overline{x_k}$
$=\overline{(x_0 + x_1} \ldots \overline{x_{k-2} + x_{k-1}}).(\prod_{j=k+1}^{j=n-1} x_j).x_k \oplus$
$\overline{(x_0 + x_1} \ldots \overline{x_{k-2} + x_{k-1}}).(\prod_{j=k+1}^{j=n-1} x_j)$

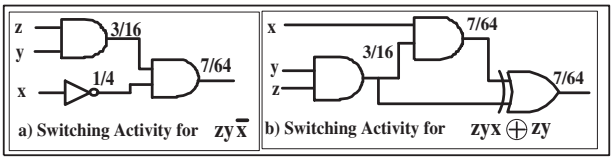

Fig. 5. Reduction of switching activity by introducing XOR gate

Figure 5 shows the reduction of the switching activity if the logic expression $zy\overline{x}$ is modified as $zyx \oplus zy$. The total switching activities for $zy\overline{x}$ and $zyx \oplus zy$ are respectively $\frac{35}{64}$ and $\frac{26}{64}$.

Lemma 3. *If a sum term in a logic expression contains an odd numbers of complemented terms, then the switching activity can be reduced by introducing $XNOR$ gate. i.e.* $\sum_{i=1}^{p} x_i + \sum_{j=k}^{2\times k} \overline{x_j} + \overline{x_l} = \sum_{i=1}^{p} x_i + (\overline{x_k x_{k+1}}) \ldots (\overline{x_{2\times k-1} x_{2\times k}}) + \overline{x_l} = E + \overline{x_l} = E x_l \odot x_l$ *where* $E = \sum_{i=1}^{p} x_i + (\overline{x_k x_{k+1}}) \ldots (\overline{x_{2\times k-1} x_{2\times k}})$.

From Lemma 2 and Lemma 3, it can be clearly seen that the introduction of XOR or $XNOR$ gate in a sum term or in a product term with an odd number of literals reduces the total switching activity and hence reduces the power dissipation of digital circuits.

VI. Algorithm For Minimization Of Total Switching Activity

The proposed algorithm for minimization of the switching activity in a circuit is shown in Figure 6. The algorithm takes the truth table of any switching function as input and gives the output as a logic expression of the switching function with minimized switching activity. The following Rules (i.e. transformations) are applied to the switching expression in order to reduce the switching activity.

Rule 1:

1) $A.B + B.C = B.(A + C)$
2) $A_1.B_1 + A_2.B_1 + A_1.B_2 + A_2.B_2 + \ldots + A_1.B_n + A_2.B_n$
   $= (A_1 + A_2).(B_1 + B_2 + \ldots B_n)$
3) $A_1.A_2 \ldots A_n.B_1.B_2 \ldots B_m + B_1.B_2 \ldots B_k.C_1.C_2 \ldots C_p$
   $= B_1.B_2 \ldots . B_k(A_1.A_2.A_3 \ldots A_n + C_1.C_2 \ldots C_p)$
4) $A_1.A_2 \ldots . A_n.B_1.B_2 \ldots . B_m$
   $+ B_1.B_2 \ldots . B_k.C_1.C_2 \ldots . C_p$
   $= B_1 \ldots . B_k.(A_1 \ldots . A_n.B_{k+1} \ldots B_m + C_1.C_2 \ldots C_p)$
   Here n, m and p are in general different and $m \geq k$.

TABLE II
Increment / Decrement of delay and area due to each rule

Rule	*Area*	*Delay*
1	*Decreased*	*Decreased*
2	*Unchanged*	*Decreased in Tree Representation* / *Increased in Chain Representation*
3	*Decreased*	*Decreased*
4	*Decreased*	*Unchanged*
5	*Increased*	*Unchanged*$(n = 2)$*or Decreased*$(n > 2)$
6	*Increased*	*Increased*
7	*Increased*	*Decreased*
8	*Increased*	*Increased*
9	*Decreased*	*Decreased*
10	*Increased*	*Increased*
11	*Increased*	*Increased*
12	*Increased*	*Increased*
13	*Increased*	*Increased*

Rule 2: Apply the transformation of Lemma 1 and Observation 1.

Rule 3: If a product (sum) term contains even number of complemented literals, then replacing each pair of complemented literals with their NOR $(NAND)$ combination by application of De Morgan's theorem will reduce the switching activity of the entire term.

Rule 4: Applying Consensus Theorem [5] i.e.
$(AB + \overline{A}C + BC) = (AB + \overline{A}C)$.

Rule 5: If $(n \geq 2)$ and n is even then apply
$A_1 + \overline{A_2}.\overline{A_3}. \ldots . \overline{A_n} = A_1 + \overline{A_1 + A_2 + \ldots A_n}$

Rule 6: If $(n \geq 3)$ then apply
$A_1 + \overline{A_2}.\overline{A_3}. \ldots . \overline{A_{n-1}}.A_n = A_1 + \overline{A_1 + A_2 + \ldots A_{n-1}}.A_n$.

Rule 7: For two or more than two variables the Boolean expression

1) $(\overline{A} + AB) = (\overline{(\overline{A + B})} + B)$.
2) $\overline{A_1} + \overline{A_2} + \ldots \overline{A_{n-1}} + A_1.A_2. \ldots . A_n$
   $= \overline{A_1.A_2. \ldots . A_{n-1} + A_n} + A_n$

Rule 8:

1) $\overline{A}B = (\overline{AB})B$.
2) $\overline{A_1}A_2A_3 \ldots A_n = \overline{A_1A_2 \ldots A_n}A_2A_3 \ldots A_n$.
3) $(\prod_{i=0}^{i=n} x_i).(\prod_{j=1}^{j=2\times p} \overline{x_j}).(\overline{x_k})$
   $= \overline{(\prod_{i=0}^{i=n} x_i).(\overline{x_1 + x_2}) \ldots (\overline{x_{2\times p-1} + x_{2\times p}})}.$
   $\overline{(\prod_{i=0}^{i=n} x_i).(\overline{x_1 + x_2}) \ldots (\overline{x_{2\times p-1} + x_{2\times p}})(x_k)}$

Rule 9:

1) $A\overline{B} + B = A + B$. This is known as absorption rule [5].
2) $\overline{A}B + \overline{B} = \overline{A.B}$.
3) $\overline{A}C + A\overline{B}C = \overline{A}C + \overline{B}C = C(\overline{AB})$.
   The last two transformation can be defined as a modified absorption rule.

Rule 10: $\overline{A}B = AB \oplus B$.

Rule 11: $A + \overline{B} = AB \odot B$.

Rule 12: Apply Lemma 2.

Rule 13: Apply lemma 3.

Algorithm Minimize Switching Activity()

Input: Truth Table
Output: Function for minimal switching activity

1. Obtain the minimal Sum-of-Product (f_{SOP}) or Product-of-Sum (f_{POS}) expression using any standard method (such as K-map, Quine- McCluskey method, Espresso)for the minimization of given switching function f. *E-XOR*($\oplus$) operations can also be considered to minimize the logic expression.
2. Apply Rule 1 and Rule 2 to reduce the switching activity.
3. If either the f_{SOP} or f_{POS} does not contain any complemented literal then
4. Calculate the total switching activity of f_{SOP} or f_{POS}.
5. Take the function with minimum switching activity.
6. Endif
7. If number of complemented literals of a product term in a f_{SOP} is even then
8. Replace a pair of complemented terms by NOR gate using De Morgan's theorem. (Rule 3).
9. Endif
10. If number of complemented literals of a sum term in a f_{POS} is even then
11. Replace a pair of complemented terms by $NAND$ gate using De Morgan's theorem. (Rule 3).
12. Endif
13. If applicable then
14. Apply Rule 4, Rule6, Rule 7, Rule 9 to reduce the switching activity.
15. Endif
16. If a f_{SOP} (f_{POS}) contains only one complemented literal $\overline{x}_i$ then
17. Apply Rule 8 or Rule 10 (in case of SOP).
18. Apply Rule 5 or Rule 11 (in case of POS).
19. Take the function with minimum switching activity.
20. Endif
21. If the product term in f_{SOP} (f_{POS}) contains odd number of complemented literals then
22. Apply rule 12 in case of SOP.
23. Or Apply Rule 13 in case of POS
24. Take the function with minimum switching activity.
25. Endif
26. end.

Fig. 6. Algorithm For Minimal Switching Activity

A. Effectiveness of the Proposed Algorithm

The total switching activity of a logic expression calculated by the proposed algorithm is less than or equal to the switching activity of the equivalent logic expression by any standard logic method.

In our proposed algorithm, first, any standard logic optimization method is used to find the minimized logic expression. Then a part of or the whole logic expression is replaced with an equivalent logic expression for which the switching activity is minimized. The transformations as given by Rule 1 to Rule 13 decrease the switching activity by i) reducing the number of NOT gates ii) increasing the number of inputs to AND or OR gates, iii) introducing XOR or XNOR gate (Lemma 2, Lemma 3) iv) a combination of these. For some specific logic expressions, it is not possible to replace the logic expression with an equivalent logic expression with less switching activity. Hence the total switching activity of a logic expression calculated by the proposed algorithm is less than or equal to the switching activity of the same logic expression calculated by any standard method. In order to reduce the switching activity, some times, the area or the delay of the circuit may increase. Table II shows the decrease or the increase of area or delay due to each transformation.

VII. Experimental Results

The implementation and power estimation of the proposed rule-based algorithm has been done by using Synopsys EDA tool -DESIGN VISION version I-2013.12-SP1, 20, 2014 under CENT OS and TSMC 120 nm library.

The experiment is done on some basic circuits and three other benchmarks circuit (alu1, cm138a, z9sym) [21]. The basic circuits and also the 2-adder and the 3-adder circuits are manually constructed. The switching activity of circuits and the associated dynamic power dissipation using conventional SOP (POS) method implemented with two-input gates with the help of chain structure and our proposed method are summarized in Table III.

We observe that the total switching activity for our proposed method never exceeds, and is less in most of the cases than the one obtained using the traditional logic optimization. To calculate the area of a circuit it is assumed that the area of inverter is 1. The area of AND, OR, $NAND$, NOR, and XOR gate are assumed to be 3, 3 2, 2 and 7 respectively. The delay of NOT, AND, OR, $NAND$, NOR, and XOR gate are assumed to be 1, 3, 3, 2, 2 and 7 respectively.

A. Comparison of Our Proposed Method with the Existing Method of [12]

In [12] the authors basically modify the Karnaugh maps to reduce the switching activity. Logic optimization using Karnaugh maps is limited to 6 variables switching functions. The reduction of the switching activity and hence the power dissipation of CMOS VLSI circuits by our proposed algorithm is not restricted to 6 variables switching functions and it is applicable for any kind of circuits. Moreover, the method of [12] is not applicable for all types of switching functions. For instance if the logic expression of a two-variable switching function is $\overline{A}B$, then the method in [12] cannot reduce the switching activity. On the other hand, our proposed method reduces the switching activity.

B. Power-Delay Tradeoff

Logic optimization using our proposed method surely minimizes the switching activity. But in order to minimize the switching activity by the proposed rules, it may also happen that a NOR gate is used instead of a single NOT gate (e.g. transformation of Rule 7, i.e., $(\overline{A}+AB) = ((\overline{A+B})+B)$ and so on). In this case, the total transistors count of the circuit increases. In our proposed method we do not consider the circuit delay. Logic optimization to minimize the switching activity and the delay as a joint objective will be the future direction of our work.

VIII. Conclusion

In this paper we propose a rule-based approach to reduce the switching activity of combinational logic circuits. This would reduce the dynamic power and hence the total power

TABLE III
COMPARISON OF SWITCHING ACTIVITY AND DYNAMIC POWER OF DIFFERENT COMBINATIONAL CIRCUITS

Circuits	TSA		% Reduction of TSA	Dynamic Power (micro watt)		Area		Delay	
	Conv	our		Conv	Our	Conv	Our	Conv	Our
3: 8 decoder	2.75	1.8125	34.09	15.7121	14.3159	39	50	7	9
4:1 MUX	1.6875	1.4375	14.81	7.3628	5.2588	35	42	13	16
Half Adder	0.4375	0.4375	0	5.1288	5.1288	10	10	7	7
Full Adder	1.5469	1.3594	12.12	17.8317	11.1890	29	26	12	12
Half Subtractor	0.6875	0.625	9.09	6.5248	5.8172	11	12	7	7
Full Subtractor	1.8125	1.4375	20.69	15.3178	14.7618	30	33	12	13
2 Bit Comparator	5.9609	4.7891	19.66	18.3713	15.4985	82	94	13	16
Priority Encoder	3.5392	2.9064	17.88	14.0391	9.0759	52	50	19	14
2-Adder	4.2218	2.9710	29.63	17.8394	14.7104	59	51	24	25
3-Adder	6.2285	3.7479	39.83	32.5028	28.608	91	89	36	39
ALU1	10.034	7.3476	26.77	21.5760	18.8346	113	96	14	13
cm18a	8.0374	5.3484	33.46	7.1527	5.97	92	77	14	12
z9sym	19.1958	15.8665	17.34	61.3653	58.35	231	210	50	57

dissipation, enabling the design of power-efficient circuits with several useful applications. Experimental results show significant reduction of the switching activity. In this paper, together with all the limitations, SOP representations are used and switching possibility of input signal are assumed to be 50% . The method can be improved by making the computation scalable and precise. Introduction of signal probability to compute the switching activity for logic circuits can be used for this computation. The method presented in this paper can be further improved by considering both delay and power as objective functions and implementation and testing of the improved algorithm with standard benchmark circuits.

ACKNOWLEDGMENT

This research has been partially supported by a grant GA16-05179S of the Czech Grant Agency, "Fault-Tolerant and Attack-Resistant Architectures Based on Programmable Devices: Research of Interplay and Common Features" (2016-2018).

REFERENCES

[1] S. Chattopadhyay and N. Choudhary, "Genetic Algorithm based Approach for Low Power Combinational Circuit Testing",Procedings of the 16th International conference on VLSI Design (VLSI'03).

[2] P.Girard, C.Landraut, S.Pravossoudovitch and D.Severac, "Reducing Power Consumption During Test Application by Test Vector Ordering", Int'l Symposium on Circuits and Systems (ISCAS 98), pp 296-299.

[3] P. Ghosal, T. Samanta, H. Rahaman and P. Dasgupta, "Thermal-aware placement of standard cells and gate arrays: Studies and observations", IEEE Computer Society Annual Symposium on VLSI, April 2008. ISVLSI'08., pp 369-374.

[4] K. Roy and S.C.Prasad, "Low Power CMOS VLSI Circuit Design", Wiley India Edition, Reprint 2011.

[5] Z. Kohavi and N. K. Jha, "Switching and Finite Automata Theory", 3^{rd} Edition, Cambridge University Press, 2010.

[6] N. Wehn and M. Munch, "Minimising power consumption in digital circuits and systems: An overview", Kleinheubacher Berichte, Band 43, pages pp.308-319, September, 1999, Kleinheubach, Germany, Invited Talk.

[7] A. P. Chandrakasan and R. W. Brodersen, "Minimizing Power Consumption in Digital CMOS Circuits", Proceedings of the IEEE, Vol 83, No. 4, April 1995.

[8] K. Roy and S. C. Prasad, "Circuit Activity Based Logic Synthesis for Low Power Reliable Operations", IEEE Transactions on VLSI, Vol-1, No. 4, December 1993, pp 503-512.

[9] S. Ahuja, A. Lakshminarayana and S. K. Shukla, "Low Power Design with High-Level Power Estimation and Power-Aware Synthesis", Springer Pub., 2011.

[10] I. Brzozowski and A. Kos, "Minimization of Power Consumption in Digital Integrated Circuits of Reduction of Switching Activity", 25^{th} Euromicro Conference, 1999, pp 376-380 Vol. 1.

[11] R. V. Menon, S. Chennupati, N. K. Samala, D. Radhakrishnan and B. Izadi, "Power Optimized Combinational Logic Design", Proceeding of the International Conference on Embeded Systems and Applications, June 2003, pp. 223-227.

[12] R. V. Menon, S. Chennupati, N. K. Samala, D. Radhakrishnan and B. Izadi, "Switching Activity Minimization in combinational Logic Design", Proceeding of the International Conference on Embeded System and Application, 2004, pp 47-53.

[13] J. Monteiro, S. Devadas, A. Ghosh, K. Keutzer, and J. White, "Estimation of average Switching Activity in Combinational Logic Circuits Using Symbolic Simulation", IEEE Transaction on Computer Aided Design of Integrated Circuits and Systems, Vol. 16, NO. 1, 1997, pp 121-127.

[14] M. Ghasemazar and M. Pedram, "Variation Aware Dynamic Power Management for Chip Multiprocessor Architectures", Design, Automation & Test in Europe Conference & Exhibition (DATE), 2011.

[15] P. Dasgupta, P. Dasgupta, D. K. Das, "A Novel Algorithm for Interconnect-aware Two level Optimization of Multi-output SOP functions", Proceeding of the 11^{th} International Workshop on Boolean Problems, Freiberg, September 2014, pp 219-226.

[16] M. Alidina, J. Monterio, S. Davadas, A. Ghosh and M. Papaeftymiou, "Precomputation-Based Sequential Logic Optiomization for Low Power", IEEE International Conference on Computer Aided design, 1994, pp 74-81.

[17] P. Parker and E. J. McCluskey, "Probabilistic treatment of General Combinatorial Networks", IEEE trans. Computers, Vol c-24, pp 668-670, 1975

[18] J. A. Darringer, W. H. Joyner, Jr. C. Leonard Berman and L. Trevillyan, "Logic Synthesis Through Local Transformations", IBM J. RES. DEVELOP. Vol. 25 NO. 4 July 1981.

[19] L. H. Goldstein, "Controllability / Observability Analysis of Digital Circuits", IEEE Transactions on Circuit and Systems, Vol. CAS-26, No. 9, 1979, pp 685-693.

[20] S. C. Seth, V. D. Agrawal, "A new model for Computation of Probabilistic testability in combinational circuits", Integration, The VLSI Journal 7(1989) pp 49-75, 1989.

[21] S. Yang, "Logic Synthesis and Optimization Benchmarks User Guide," Technical Report 1991 IWLS-UG-Saeyang, MCNC, Research Triangle Park, NC, January 1991, p. 45.