Hindawi Publishing Corporation
VLSI Design
Volume 2016, Article ID 3191286, 14 pages
http://dx.doi.org/10.1155/2016/3191286

Research Article

Shared Reed-Muller Decision Diagram Based Thermal-Aware AND-XOR Decomposition of Logic Circuits

Apangshu Das and Sambhu Nath Pradhan

Department of Electronics & Communication Engineering, National Institute of Technology Agartala, Agartala, Barjala, Jirania, Tripura 799046, India

Correspondence should be addressed to Apangshu Das; apangshuextc@gmail.com

Received 15 December 2015; Accepted 29 March 2016

Academic Editor: Jose Carlos Monteiro

The increased number of complex functional units exerts high power-density within a very-large-scale integration (VLSI) chip which results in overheating. Power-densities directly converge into temperature which reduces the yield of the circuit. An adverse effect of power-density reduction is the increase in area. So, there is a trade-off between area and power-density. In this paper, we introduce a Shared Reed-Muller Decision Diagram (SRMDD) based on fixed polarity AND-XOR decomposition to represent multioutput Boolean functions. By recursively applying transformations and reductions, we obtained a compact SRMDD. A heuristic based on Genetic Algorithm (GA) increases the sharing of product terms by judicious choice of polarity of input variables in SRMDD expansion and a suitable area and power-density trade-off has been enumerated. This is the first effort ever to incorporate the power-density as a measure of temperature estimation in AND-XOR expansion process. The results of logic synthesis are incorporated with physical design in CADENCE digital synthesis tool to obtain the floor-plan silicon area and power profile. The proposed thermal-aware synthesis has been validated by obtaining absolute temperature of the synthesized circuits using HotSpot tool. We have experimented with 29 benchmark circuits. The minimized AND-XOR circuit realization shows average savings up to 15.23% improvement in silicon area and up to 17.02% improvement in temperature over the sum-of-product (SOP) based logic minimization.

1. Introduction

With the rapid increase in the functional complexity and miniaturization of chips, power-density is becoming a critical concern in VLSI design and synthesis methodologies. Feature size scaling to meet the demand of the portability and performance issues increased the total power utilization of the chip. Consequently, the power-density becomes extensive and generates a thermal effect, which reduces the performance and efficiency of the circuit. Even the integrated circuit (IC) chip may burn out due to thermal runaway. In recent time, power-density is an important constraint for designing the VLSI circuits to reduce the thermal effect, because power-density directly converges to temperature [1]. So, optimized realization of a circuit taking power-density as a parameter in cost is very much important to limit the temperature generation. Temperature was given importance by researchers in physical design domain, but the cooling cost became high. With the high performance processors, the cooling solutions are rising at $1–3 or more per watt of power dissipation [2, 3]; this shows that cooling costs are increased exponentially with the increase of power-density. So, design-time thermal-aware techniques can be used to improve the power and thermal characteristics of integrated circuits.

Logic minimization plays an important role in combinational synthesis domain to optimize the circuit by increasing the shared logic within the functions. Once the minimized circuit is obtained, it is the switching activity and transition probabilities of the logic (dynamic power) that determine the power consumption in the circuit. Then, the power-density is obtained by taking the ratio of the power consumption and the utilized chip area. Here, in this paper, we have proposed a logic synthesizer which tries to optimize the chip area and power-density by providing trade-offs between the two and tries to reduce the thermal effect of the combinational logic circuits.

Multioutput function optimization aims at reducing the circuit area by extracting common subexpressions within the subfunctions. The most popular CAD tool packages which utilize the above logic are Espresso [4], SIS [5], and ABC [6]. Espresso targeted AND-OR based PLA structure and is more commonly known as two-level minimizer. On the other hand, SIS and ABC utilize multilevel logic circuits to increase the sharing between subfunctions. The logic implementation reported in [4, 5, 7] utilizes AND-OR realization to reduce the circuit area. However, in many real-life circuits used in the fields of coding theory, telecommunication, linear system, computer arithmetic coding circuits, error detection-correction circuits, and data encryption and decryption circuits are inherently the basic functions of mod-2 sum form. In such cases, AND-XOR minimized algorithms often produce more compact circuit than the AND-OR based realizations. AND-XOR based PLA realization offers higher testability than AND-OR based circuits. However, applications of AND-XOR based circuits have so far not become popular due to the following two obstacles:

(i) XOR gates have slow speed and require large silicon area to realize in comparison with OR gates.

(ii) The problem of optimization of AND-XOR functions is difficult although there has been a great deal of research in recent years.

With the development of new technologies and the advent of various field programmable gate array (FPGA) devices, the first obstacle has been solved. In programmable devices, the XOR gate is either easily realized in "universal modules" or directly available. For example, ATMEL FPGA series AT6000 uses two various input gates such as XORs, ANDs, and NANDs to configure logic blocks [8]. Regarding the second obstacle, more recently, there has been some success in achieving area reduction by employing optimization techniques specifically targeted towards initial AND/XOR representations in the well known Reed-Muller (RM) form.

In order to develop an AND-XOR based circuits realization, there are several types of expressions such as positive polarity Reed-Muller (PPRM), fixed polarity Reed-Muller (FPRM), pseudo Reed-Muller, generalized Reed-Muller, XOR sum of products, and Kronecker and pseudo Kronecker forms [9]. Each of these circuits has its own advantages. As far as XOR synthesis is concerned, this paper concentrates on the synthesis of FPRM circuits only.

In the above background, the problem of the current work can be addresses as follows.

A multi-input, multioutput Boolean function F and weight factors perform FPRM decomposition and share the product terms. Minimization depending on weighted sum approach for area (number of product terms) and power-density is performed. The circuit realization is carried up to physical design synthesis to obtain the actual area and temperature.

The rest of the paper is organized as follows. Section 2 illustrates the motivation and previous work on AND-XOR synthesis. Section 3 presents thermal-aware AND-XOR problem formulation and synthesis approach. Section 4 illustrates the GA formulation for thermal-aware SRMDD based AND-XOR network synthesis. Section 5 presents experimental results and finally Section 6 draws the conclusion and future works.

2. Motivation and Previous Works

The motivation of AND-XOR realization comes from Example 1.

Example 1. Consider a Boolean function F consisting of 3-input and 2-output functions consisting of the following subfunctions:

$$\begin{aligned} f_1(z, y, x) &= z'y'x + z'yx + zy'x + zyx' + zyx, \\ f_2(z, y, x) &= z'y'x + z'yx + zy'x + zyx'. \end{aligned} \tag{1}$$

By realization of function F using AND-OR network, it requires 4 product terms: xyz', $x'z$, $y'z$, and z. If we realize the same function F using AND-XOR network with all positive polarities, it will provide 3 product terms. The FPRM forms of f_1 and f_2 subfunctions are

$$\begin{aligned} f_1 &= zyx \oplus yx \oplus z, \\ f_2 &= yx \oplus z. \end{aligned} \tag{2}$$

The Reed-Muller (RM) canonical expansion of a k-variable Boolean function f can be represented by 2^k terms. The general expansion of RM is given by

$$\begin{aligned} f(x_1, x_2, \ldots, x_n) = a_0 \oplus a_1 x_1 \oplus a_2 x_2 \oplus \cdots \\ \oplus a_{2k-1} x_1 x_2, \ldots, x_k, \end{aligned} \tag{3}$$

where $a_i \in \{0, 1\}$. All x_i input variables appear in positive polarities in the expansion. Several modified versions of this basic canonical form have been studied. If the variables are allowed to take both positive and negative polarities, this is known as generalized Reed-Muller (GRM) form. Any arbitrary Boolean function $f(x_1, x_2, \ldots, x_n)$ can be expanded to represent it into AND-XOR network by deriving the Davio expansions [11]. The expansions are

$$\begin{aligned} f(x_1, x_2, \ldots, x_n) &= x_i \cdot f_{x_i} \oplus x_i' \cdot f_{x_i'} && \text{Shannon's Exp (SE)} \\ f(x_1, x_2, \ldots, x_n) &= f_{x_i'} \oplus x_i \cdot \left(f_{x_i} \oplus f_{x_i'}\right) && \text{Positive Davio (pD)} \\ f(x_1, x_2, \ldots, x_n) &= f_{x_i} \oplus x_i' \cdot \left(f_{x_i} \oplus f_{x_i'}\right) && \text{Negative Davio (nD),} \end{aligned} \tag{4}$$

where $f_{x_i} = f(x_1, x_2, \ldots, x_{i-1}, 1, x_{i+1}, x_n)$ and $f_{x_i'} = f(x_1, x_2, \ldots, x_{i-1}, 0, x_{i+1}, x_n)$ are called the cofactor of x_i.

If we decompose the function f using Shannon's Expansion, three gates are required (two ANDs and one XOR), whereas only two gates (one AND and one XOR) are required to realize the same function f using positive Davio (pD) or negative Davio (nD). In this work, we are applying the positive Davio expansion or negative Davio expansion to the given function f using either positive or negative polarity of variables but not both for each variable. Then, the Boolean function f is logically expressed as fixed polarity Reed-Muller (FPRM) expansion. For an n-variable function, there are at most 2^n different FPRMs. The minimization problem is to find one with the minimum products among 2^n possible FPRMs. To solve the above problem, we have applied a Genetic Algorithm (GA) based formulation to identify the best polarity assignment to the input variables to get the desired output.

Detailed descriptions of two-level AND-XOR network synthesis have been done in [12, 13]. Better and minimized realization can be possible using AND-XOR logic synthesis compared with that of AND-OR synthesis in terms of fewer product terms and that has been reported by Sasao and Besslich in [14] and Ye and Roy in [15]. Sasao et al. deal with the problem of minimizing the two-level AND-XOR PLAs by utilizing both positive and negative polarity of variables and proposed several heuristic methodologies in mod-2 SOPs in [14, 16]. Realization of Boolean functions in the positive polarity AND-XOR form has long been proposed as Reed-Muller expansion in [17]. The modified versions of this basic canonical form have been studied by several researchers as time passes. The representation in which a variable can have either positive or negative polarity throughout the function is known as fixed polarity Reed-Muller (FPRM) form as given by Davio and Deschamps [18]. FPRM expansion utilizes a much smaller number of product terms than the original Reed-Muller form with high testability. An FPRM based heuristic approach has been proposed by Sarabi and Perkowski to find out the best polarity assignment [19]. A GA based polarity selection of FPRM realization scheme for multioutput Boolean function to minimize the area was presented by Chattopadhyay et al. in [12]. Low-power decomposition of XOR based synthesis has been presented by Narayanan and Liu [10]. In [13], a GA based area power trade-off analysis has been reported by Pradhan and Chattopadhyay. Elaborated survey of the work done so far has been given in [20]. However, all the above works did not consider the power-densities as a coefficient of estimating temperature of AND-XOR based circuits to analyze the thermal effect. We have contributed a trade-off analysis by taking power-density along with area. In logic synthesis level absolute value of temperature is unknown, so to evaluate temperature we have to consider power-density for temperature from the following equation. Temperature is directly proportional to power-density and this can be established by [21]

$$T_{\text{chip}} = T_a + R_\theta \frac{P_{\text{total}}}{A}. \tag{5}$$

In (5), T_{chip} is the average chip temperature. T_a is the ambient temperature ($T_a = 25^\circ\text{C}$). R_θ is the equivalent thermal resistance of the substrate (Si) layer plus the package and heat sink ($\text{cm}^2 \cdot {}^\circ\text{C/W}$). P_{total} (in W) is the total power consumption. A (in cm^2) is the chip area.

Keeping ambient temperature constant in (5), it can be concluded that temperature generation depends only on power and area (since equivalent thermal resistance is constant for a particular substrate). This has led us to consider power-density as the constraint of temperature and subsequently consider power-density minimization along with area during polarity selection of FPRM based AND-XOR network synthesis of circuits.

3. Thermal-Aware AND-XOR Problem Formulation and Synthesis Approach

To represent a Boolean function efficiently into FPRM, the critical issue is to select the polarity thoughtfully for maximum sharing considering the optimization parameters. In this section, we have explained the method for assigning the input variable polarity and calculation of area and power-density depending on polarity assigned.

n-input, m-output Boolean function can be realized as an FPRM expansion by maintaining each variable with a fixed polarity, either positive or negative throughout the expansion. A variable appears either in true or in complemented form within the expansion. This can be achieved by the following steps:

(i) Boolean functions are expressed into disjoint cube representation.

(ii) Without affecting the functionality, ORs are replaced with XORs.

(iii) Each variable is assigned with consistent polarity for FPRM representation.

(iv) Decompose the literals into a consistent polarity one.

The output subfunctions are represented into AND-OR cubes into Boolean functions. The AND-OR cubes are converted to a set of disjoint cubes. Then, without affecting the functionality, the ORs can be replaced with XORs. After obtaining the set of AND-XOR cubes, the next task is to determine the polarity assignment. The polarity can be assigned as $\langle p_1, p_2, p_3, \ldots, p_n \rangle$, $p_i \in \{0, 1\}$ to the variables $\langle x_1, x_2, x_3, \ldots, x_n \rangle$ to maximize the sharing of product terms to obtain the desired output of the FPRM realization. A variable with polarity 1 occurs in true form in all product terms, whereas variable with polarity 0 occurs only in complemented form. To obtain the FPRM form, the literals having polarities different from that assigned to the corresponding variables are replaced by $(1 \oplus x_i)$ if input variable polarity p_i is 1. Otherwise, the variable is x_i'. Depending on the polarity, the Boolean function gets represented in different FPRM realization. Each realization provides a different area in the form of a number of product terms and respective power-densities and allows a trade-off between the two. Example 2, in Section 3.1, explains the area computation which is preceded by power-density estimation.

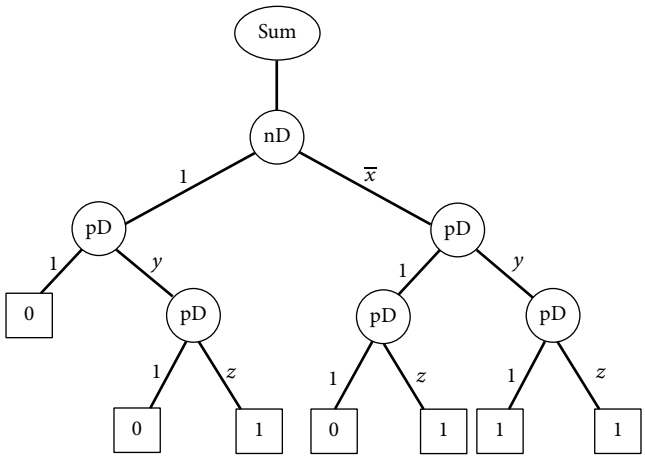

Figure 1: FPRM expansion tree for Sum = $\overline{x}yz \oplus \overline{x}y \oplus \overline{x}z \oplus yz$.

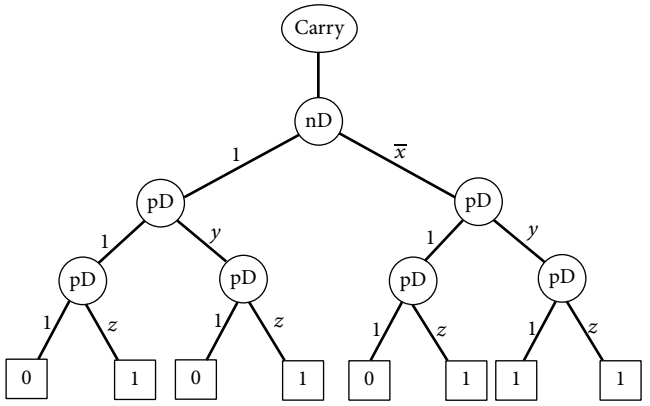

Figure 2: FPRM expansion tree for Carry = $\overline{x}yz \oplus \overline{x}y \oplus \overline{x}z \oplus yz \oplus y \oplus z$.

3.1. Shared Reed-Muller Decision Diagram (SRMDD) Decomposition Based on Fixed Polarity and Area Computation. Shared decision diagrams are used to represent multioutput Boolean functions, like

$$F = (f_0, f_1, \ldots, f_{m-1}) : B^n \longrightarrow B^m, \tag{6}$$

where $B = \{0, 1\}$ and n and m denote the number of input and number of output variables, respectively. m different logic functions are decomposed into AND-XOR based realization by maintaining a fixed polarity. The realized functions share the identical terms, which are represented by a common part of the SRMDD. In this paper, the shared FPRMs within the subfunctions are termed SRMDD. By iteratively applying FPRM decomposition and sharing the identical product terms, we obtain a compact SRMDD. Example 2 shows the formation of SRMDD of the full-adder circuit.

Subsequently, area computation has been illustrated.

Example 2. In full-adder circuit, x, y, and z are the three inputs added to produce the "Sum" and "Carry" outputs. The functions are given by

$$\begin{aligned} \text{Sum} &= \overline{x}\,\overline{y}z + \overline{x}y\overline{z} + x\overline{y}\,\overline{z} + xyz, \\ \text{Carry} &= \overline{x}yz + x\overline{y}z + xy\overline{z} + xyz. \end{aligned} \tag{7}$$

Both output functions can be realized as FPRM based AND-XOR network by applying the positive Davio expansion to y and z and negative Davio to x. y and z appear as true form and x appears as complemented form by substituting $x = (\overline{x} \oplus 1)$, $\overline{y} = (y \oplus 1)$, and $\overline{z} = (z \oplus 1)$ into the output functions Sum and Carry. After decomposition, we have

$$\begin{aligned} \text{Sum} &= \overline{x}yz \oplus \overline{x}y \oplus \overline{x}z \oplus yz, \\ \text{Carry} &= \overline{x}yz \oplus \overline{x}y \oplus \overline{x}z \oplus yz \oplus y \oplus z. \end{aligned} \tag{8}$$

The identical product terms, such as $\overline{x}yz$, $\overline{x}y$, $\overline{x}z$, and yz, are shared between the two output functions.

Figure 1 illustrates the formation of FPRM expansion tree of Sum function. The nodes with pD denote the positive Davio expansions, and the nodes with nD denote the negative Davio expansions. In each path from the root node to constant 1, the logical product of the labels in the path corresponds to a product term in an FPRM. Figure 2 shows the FPRM expansion tree for Carry function. After sharing the identical product terms, the SRMDD tree of Sum and Carry generates 6 product terms, whereas if we expand the tree separately it would require 10 product terms. The product terms are the representative area for the Boolean functions. By changing the polarity of a variable in a given function, the structure of the circuit is changed. Initially, the circuit was in the form of AND-OR circuit. After polarity assignment, the circuit is represented in the form of AND-XOR circuit. But the functionality of both structures is the same. In particular, the set of input responses of both circuits is the same. So, the truth vector formed in AND-XOR may differ from its primary initial AND-OR realization keeping the functionality unchanged. In Reed-Muller AND-XOR realization chance of sharing product terms among the subfunction increases, which results in area reduction.

Shared Reed-Muller Decision Diagram based on fixed polarity obeys the commutative law of addition and multiplication. Variable ordering does not affect the decision diagram. We kept the variable ordering fixed and changed only the polarity of the variable. In Figure 3, the bold straight lines are shared between Sum and Carry functions and dotted lines are only expanded branches for Carry.

3.2. Powers-Density Estimation. Power-density can be defined as the amount of power drawn per unit area. In CMOS logic circuits, the power utilization takes place mainly due to three components: switching (capacitive), short-circuit, and leakage power dissipation. Among these, switching power is the main contributor, and short-circuit and leakage powers become significant when technology scales down to 65 nm technology. Switching power consumption occurs due to charging and discharging of load and parasitic capacitors. It can be evaluated by

$$\begin{aligned} p_{\text{dynamic}} &\approx p_{\text{switching}} \\ &= (\text{ESA})_L \cdot C_L \cdot V_{\text{DD}}^2 \cdot f + \sum_i (\text{ESA})_i \cdot C_i \\ &\quad \cdot V_{\text{DD}} \left(V_{\text{DD}} - V_T\right). \end{aligned} \tag{9}$$

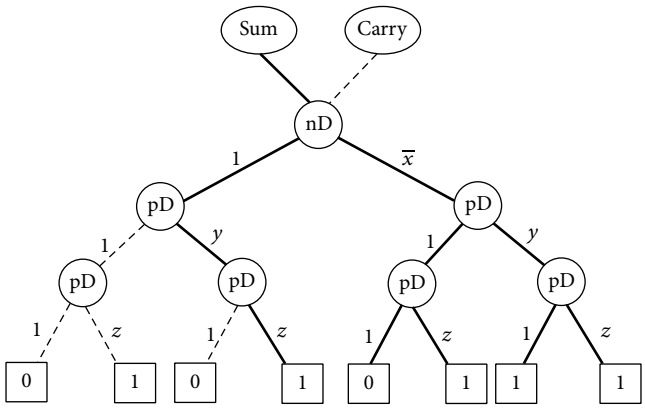

Figure 3: SRMDD expansion tree for Sum and Carry.

Here, ESA_L and ESA_i are the switching activity at the load and the internal node of the circuit, respectively, V_{DD} is the supply voltage, f is the frequency of operation, and C_L and C_i are the load and internal gate capacitances, respectively. To estimate the switching power dissipation, we need to compute the expected switching activity (ESA) of the logic gates. It is defined as the expected number of signal transitions at the outputs of the gates of a combinational logic circuit, and we have used the same technique used in [13]. We assume that primary inputs are uncorrelated and are statically independent of each other, and the primary input probability can be expressed as

$$\mathrm{Prob}\left(\mathrm{input} = 1\right) = \mathrm{Prob}\left(\mathrm{input} = 0\right) = \frac{1}{2}. \tag{10}$$

First, we need to determine the ESA of a single gate. The logic gate changes its state when the current state of the output differs from the previous one. Therefore, the probability of the output of a gate changing its state can be evaluated by

$$\begin{aligned}&\mathrm{Prob}\left(\mathrm{current\_op} = 0\right) * \mathrm{Prob}\left(\mathrm{previous\_op} = 1\right)\\ &\quad + \mathrm{Prob}\left(\mathrm{current\_op} = 1\right)\\ &\quad * \mathrm{Prob}\left(\mathrm{previous\_op} = 0\right).\end{aligned} \tag{11}$$

We assume that the probability does not change with time. So, the estimated switching activity of logic gate ($\mathrm{ESA_g}$) is given by

$$\mathrm{ESA_g} = 2 * \mathrm{Prob}\left(\mathrm{op} = 0\right) * \mathrm{Prob}\left(\mathrm{op} = 1\right). \tag{12}$$

The estimated switching activity for an "i" input AND gate with primary inputs ($\mathrm{ESA_{AND}}$) is given by

$$\mathrm{ESA_{AND}} = 2 * \left(\frac{1}{2^i}\right) * \left[1 - \left(\frac{1}{2^i}\right)\right]. \tag{13}$$

For "n" such AND gates in the first level, switching activity is given by

$$\mathrm{ESA_{AND(}}_{n)} = \sum_n 2 * \left(\frac{1}{2^i}\right) * \left[1 - \left(\frac{1}{2^i}\right)\right]. \tag{14}$$

1st	2nd	3rd	4th	5th
1	0	1	0	1

Figure 4: Structure of a chromosome.

To compute the ON-probability of second-level XOR gates, we consider the Boolean function implemented by them. If a gate with "i" inputs realizes a function with "p" ON-terms, the ON-probability is given by $p/2^i$. Thus, switching activity of the node is estimated by

$$\mathrm{ESA_{XOR}} = 2 * \left[\frac{p}{2^i}\right] * \left[1 - \frac{p}{2^i}\right]. \tag{15}$$

After estimation of switching activity that represents the switching power dissipated by the required Boolean function, it is divided by the estimated area of the logic after the realization of SRMDD expansion to obtain the power-density:

$$\mathrm{Power\text{-}density_{SRMDD}} = \frac{\mathrm{ESA_{SRMDD}}}{\mathrm{Node\_Count_{SRMDD}}}. \tag{16}$$

This parameter participates while calculating the fitness of particular expansion. The estimated switching activity of SRMDD expansion ($\mathrm{ESA_{SRMDD}}$) of Example 2 by the abovementioned procedure is 1.53. Total product terms ($\mathrm{Node\_Count_{SRMDD}}$) are 6. Therefore, the power-density ($\mathrm{Power\text{-}density_{SRMDD}}$) is (1.53/6) 0.255.

4. Genetic Algorithm Formulation for SRMDD AND-XOR Network Synthesis

Genetic Algorithm (GA) is a stochastic heuristic search method that utilizes the mechanism of natural selection. In this section, we structured the solutions of AND-XOR SRMDD of each circuit as chromosomes to reduce the area and power-density (temperature). The genetic formulation involves the careful and proficient choice of proper encoding of the input variables to form chromosome (each chromosome represents a possible solution), cost function measuring the suitability of the chromosomes in a population, elitism (direct copy to save the best chromosomes), crossover operator, mutation operator, and termination criterion.

4.1. Chromosome Encoding. n-input, m-output Boolean function is elegantly represented as chromosome into a string of bits of length n. The chromosome is a set of n variables $(i_1, i_2, i_3, \ldots, i_n)$ of positive and negative polarity. If the ith bit is "1," it represents the notion that the ith variable is implemented in positive polarity, whereas if the jth bit is "0," it represents the notion that the jth variable is in the form of negative polarity. For a five-input Boolean function, the structure of a chromosome may be described as in Figure 4. The first, third, and fifth bits are represented as "1"; that is, the first, third, and fifth input variables are represented as positive polarity, whereas the second and fourth bits are represented as "0" which means the corresponding variables are represented as the negative polarity. We considered population size of 50 to 100 depending on the number of outputs.

4.2. Fitness Function Measurement. The fitness of a chromosome is determined by the suitability of the resulting circuit. We have used a weighted linear combination method for the area (number of product terms) and estimated power-density. Fitness of a particular chromosome c can be determined by using the following formula:

$$\text{fitness}(c) = w_1 \frac{\text{area}(c)}{\text{area\_max}} + w_2 \frac{\text{power-density}(c)}{\text{power-density\_max}}. \quad (17)$$

In (17), "area_max" and "power-density_max" are the maximum area and maximum power-density of any chromosome after SRMDD realization of the circuit in the first generation. For a chromosome (c), the area and power-density are represented by "area(c)" and "power-density(c)." The weights w_1 and w_2 can be set by the designer with $w_1 + w_2 = 1$.

4.3. Elitism (Direct Copy). Elitism is a technique to prevent losing the best-found solution in a population [22]. The best 20% chromosomes of the present generation are directly copied to the next generation and these are considered as the "elite group." The best solutions are propagating to the next generation by elitism methodology. Elitism guarantees that the best solutions are not lost or inadvertently degraded by crossover or mutation.

4.4. Crossover. The crossover operator constructs new solutions by crossing over two parent chromosomes at randomly selected crossover points. In our GA formulation, the selection of parent chromosomes is not fully random; it is conditionally biased towards the better fitness chromosomes. Using two-point methodology, crossover operation generates 60% of the chromosomes and propagates them to the next generation. The selection of participating chromosomes for crossover is biased towards the "elite group" of the total population. To obtain the "elite group," the whole population is sorted depending on the fitness value and 20% of the population with better fitness value is considered.

To select a chromosome participating in crossover, first, a uniform random number between 0 and 1 is generated. If the number is greater than 0.5, a chromosome from the "elite group" is selected randomly. Otherwise, a chromosome is selected from the entire population. This biased selecting method enables generating better offspring as compared to the truly random one. After generating each pair of chromosomes, a check is made with the members of the present population and duplicate chromosomes are eliminated.

Figures 5 and 6 show the two methods of crossover operation. Two parents, Chromosome (x_1) and Chromosome (x_2), are selected from the present generation for the evolution of offspring, which will participate as chromosomes for the next generation. Two crossover points (Pt1 and Pt2) are selected randomly. These two points segment the parent chromosomes into three parts. Chromosome (x_1) is divided into x_{11}, x_{12}, and x_{13} whereas Chromosome (x_2) is divided into x_{21}, x_{22}, and x_{23} segments. In case of Method 1, it produces Chromosome (y_1) as $x_{11}(x_{22})x_{13}$. Figure 6 shows the second method of crossover offspring generation. In this case, Chromosome (y_2) is generated as $x_{21}(x_{12})x_{23}$. After redundancy check, the generated offspring contribute the population of chromosome for the next generation.

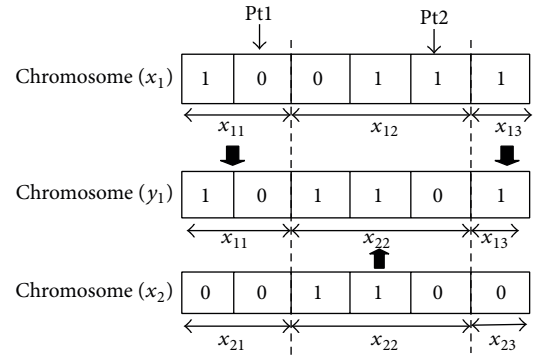

Figure 5: Crossover operation (Method 1).

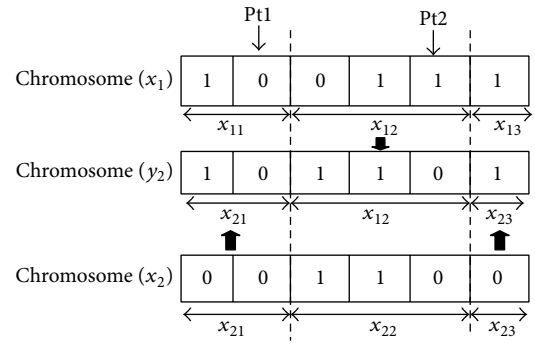

Figure 6: Crossover operation (Method 2).

4.5. Mutation. In Genetic Algorithm, the genetic diversity from one generation of population to the next is maintained by mutation operation. It is intended to prevent falling of all solutions in the population into a local optimum of the solved problem. 20% of the chromosomes of the next generation will be produced using mutation operator. To perform mutation, few randomly selected bit positions (mutation points) within the chromosome are inverted. Figure 7 illustrates the mutation operation. Chromosome (x) is chosen from the present generation for mutation operation. Randomly, two positions are selected as mutation points (Mp1 and Mp2) by the abovementioned procedure used in crossover. The selected position bit gets inverted from "0" to "1" and/or "1" to "0"; other remaining position bits are unaltered. The newly generated offspring become the chromosome of the next generation.

4.6. Termination Criterion. The termination of algorithm depends on the fitness criteria. GA terminates if there is no improvement in fitness value for 100 consecutive generations. The best chromosome at the final generation is taken as the optimum solution with respect to weighted sum of area and power-density (temperature).

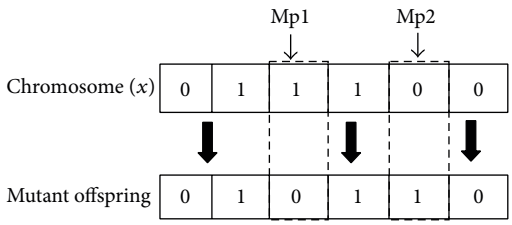

Figure 7: Mutation operation.

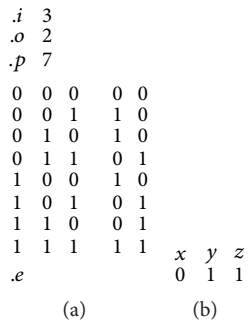

Figure 8: (a) Full-adder circuit in the form of ".pla" (full adder with 3 inputs (x, y, z) and 2 outputs (S, C)). (b) Chromosome encoding by assigning polarities to input variables. Input variable encoding (x: negative polarity; y and z: positive polarity).

5. Experimental Results

In this section, we present the proposed GA based thermal-aware SRMDD AND-XOR implementation technique. The algorithm has been implemented in C language and simulated on a Pentium IV machine with 3.4-GHz clock frequency and 3-GB RAM memory using Linux platform. For experimental validation, we applied the algorithm on a number of benchmark circuits from LGSynth93 benchmark suit. Discussion is parted into three sections. The first part concerns the simulation results based on area and power-density aware SRMDD AND-XOR circuits, which proceeded by RTL synthesis of algorithmic resultant circuits using CADENCE "RTL Compiler." The last segment contains the ASIC implementation using CADENCE "encounter."

5.1. Simulation Results Based on Area and Power-Density Aware SRMDD AND-XOR Circuits. Boolean functions formatted as ".pla" file are considered as input benchmark circuits. Figure 8 shows the format of full-adder circuit as a .pla file. Subsequently, input variable encoding and SRMDD AND-XOR circuit realization of full adder has been explained. Functions are encoded into chromosomes as explained in Section 4.1. After encoding, the next task is to implement the .pla file into SRMDD AND-XOR format. That can be achieved by the method explained in Section 3.1.

The sum function (S) can be converted into FPRM AND-XOR decomposition by replacing all the OR gates by XOR gates after obtaining disjoint cube. Then, by maintaining

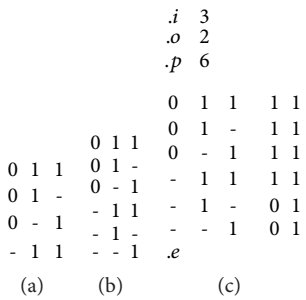

Figure 9: (a) Input bit format of FPRM based Sum. (b) Input bit format of FPRM based Carry. (c) .pla format of SRMDD based full-adder circuit.

assigned polarities to each variable, the function can be written as

$$\begin{aligned} S &= \overline{x}\,\overline{y}z + \overline{x}y\overline{z} + x\overline{y}\,\overline{z} + xyz \\ &= \overline{x}\left(y \oplus 1\right) z \oplus \overline{x}y\left(z \oplus 1\right) \oplus \left(\overline{x} \oplus 1\right)\left(y \oplus 1\right)\left(z \oplus 1\right) \\ &\quad \oplus \left(\overline{x} \oplus 1\right) yz = \overline{x}yz \oplus \overline{x}y \oplus \overline{x}z \oplus yz. \end{aligned} \tag{18}$$

By maintaining the same rule, the function for carry (C) can also be transformed into FPRM based AND-XOR decomposition. The elaboration is given below:

$$\begin{aligned} C &= \overline{x}yz + x\overline{y}z + xy\overline{z} + xyz \\ &= \overline{x}yz \oplus \left(\overline{x} \oplus 1\right)\left(y \oplus 1\right) z \oplus \left(\overline{x} \oplus 1\right) y\left(z \oplus 1\right) \\ &\quad \oplus \left(\overline{x} \oplus 1\right) yz = \overline{x}yz \oplus \overline{x}y \oplus \overline{x}z \oplus yz \oplus y \oplus z. \end{aligned} \tag{19}$$

The input format generated by Sum and Carry after FPRM transformation is given in Figures 9(a) and 9(b), respectively. Figure 9(c) shows the SRMDD encoding of both functions into single .pla file.

Maintaining the same procedure, we have applied the SRMDD to a number of benchmark circuits. Table 1 shows a comparative study of the best area results of our approach with the RM tree based and heuristic algorithm based area reported in [10]. It has been observed that our GA based SRMDD formulation is quite comparable because we obtained around 5.00% improvement with respect to heuristic based approach. Except for three circuits (misex1, rd73, and squar5), all other benchmark circuits reached the optimal value with respect to RM tree based approach. Columns 2, 3, and 4 show the input, output, and number of product terms present in the benchmark circuits. Column 5 shows our area optimal results and columns 6 and 7 show the area reported in [10].

The trade-offs have been observed by varying the weights of area and power-density as shown in (17). Table 2 shows some of the example cases for area power-density trade-offs for different weight values assigned to area (w_1) and power-density (w_2) in a range of 0 to 1. When ($w_1 = 1$, $w_2 = 0$), 100% weightage is given to area and no control is paid towards the power-density constraint. Similarly, when ($w_1 = 0$, $w_2 = 1$), the circuit shows power-density aware optimization only. Column 16 shows the max. CPU time required in all

Table 1: Area comparison of SRMDD with [10].

Benchmark circuits	I	O	P	Area (number of product terms) ($w_1 = 1, w_2 = 0$)		
				SRMDD based AND-XOR	RM tree [10]	Heuristic [10]
5xp1	7	10	75	61	61	61
alu2	10	6	251	225	—	—
apex4	9	19	438	445	445	512
clip	9	5	192	206	206	206
cm82a	5	3	23	16	—	—
cm162a	14	5	47	25	—	—
cm163a	16	5	34	18	—	—
duke2	22	29	87	255	255	267
ex5	8	63	256	113	113	171
f51m	8	8	76	56	56	73
inc	7	9	34	48	48	49
lal	26	19	102	107	—	—
misex1	8	7	32	32	20	20
pbo2	15	15	123	174	—	—
pcler8	27	17	61	40	—	—
rd53	5	3	32	20	20	20
rd73	7	3	141	64	63	63
shiftc	8	21	75	47	—	—
sqrt8	8	4	40	26	—	—
squar5	5	8	30	23	22	22
tcon	17	16	24	24	—	—
ttt2	24	21	158	107	—	—
x2	10	7	35	30	—	—
xor5	5	1	16	5	5	5
z4ml	7	4	59	33	—	—
Z5xp1	7	10	128	61	—	—
o64	130	1	65	61	—	—
apex5	117	88	1227	986	—	—
x3	135	99	739	702	—	—
Average % improvement with respect to RM tree [10]				**−3.61**		
Average % improvement with respect to heuristic [10]				**5.00**		

decompositions and it is expressed in millisecond (ms). To outline the results, we have calculated average percentage increase with respect to minimum area and minimum power-density.

Trade-off analysis between area and power-density is shown in Table 3. It is clear from Table 3 that the optimum area result is obtained at weight ($w_1 = 1, w_2 = 0$) where area is minimum of all other combinations but power-density is increased by 20.13% from its minimum value. Similarly, optimum power-density aware circuit is obtained at weight ($w_1 = 0, w_2 = 1$), where power-density is minimum of all other combinations but, on the contrary, area value is increased by 79.39% on average. It is clear from the graph in Figure 10 that the optimal result with respect to area and power-density is obtained at weight ($w_1 = 0.6, w_2 = 0.4$) where average percentage increases in area and power-density are 9.17% and 13.84% with respect to minimum area and power-density, respectively. In the next section, we are going to discuss RTL (Register-Transfer Level) synthesis of the result obtained from algorithmic level to find the absolute temperature of the optimized circuit as given in [23, 24].

5.2. Synthesis Using CADENCE RTL Compiler. The optimum results obtained from algorithmic level are translated into

Table 2: SRMDD based area power-density trade-off analysis.

Benchmark circuits	$w_1 = 0, w_2 = 1$		$w_1 = 0.2, w_2 = 0.8$		$w_1 = 0.4, w_2 = 0.6$		$w_1 = 0.5, w_2 = 0.5$		$w_1 = 0.6, w_2 = 0.4$		$w_1 = 0.8, w_2 = 0.2$		$w_1 = 1, w_2 = 0$		Max CPU time (ms)
	Area	Pow_den	Area	Pow_den	Area	Pow_den	Area	Pow_den	Area	Pow_den	Area	Pow_den	Area	Pow_den	
5xp1	91	0.191	72	0.194	61	0.201	61	0.201	61	0.201	61	0.201	61	0.201	0.157
alu2	503	0.084	288	0.087	245	0.091	245	0.091	245	0.091	233	0.095	225	0.100	3.172
apex4	445	0.116	445	0.116	445	0.116	445	0.116	445	0.116	445	0.116	445	0.116	47.95
clip	206	0.092	206	0.092	206	0.092	206	0.092	206	0.092	206	0.092	206	0.092	1.360
cm82a	16	0.230	16	0.230	16	0.230	16	0.230	16	0.230	16	0.230	16	0.230	0.019
cm162a	234	0.087	127	0.087	82	0.106	76	0.109	64	0.118	42	0.150	25	0.219	0.032
cm163a	72	0.164	44	0.168	45	0.172	44	0.168	31	0.195	23	0.230	18	0.283	0.024
duke2	420	0.029	420	0.029	396	0.031	255	0.049	255	0.049	255	0.049	255	0.049	20.292
ex5	119	0.173	119	0.173	119	0.173	113	0.188	113	0.188	113	0.188	113	0.188	49.131
f51m	101	0.194	56	0.195	56	0.195	56	0.195	56	0.195	56	0.195	56	0.195	0.201
inc	107	0.181	77	0.187	59	0.199	59	0.199	49	0.225	49	0.225	49	0.225	0.210
lal	108	0.296	107	0.289	107	0.289	107	0.289	107	0.289	107	0.289	107	0.289	1.861
misex1	60	0.254	32	0.262	32	0.262	32	0.262	32	0.262	32	0.262	32	0.262	0.049
pbo2	194	0.058	194	0.058	174	0.065	174	0.065	174	0.065	174	0.065	174	0.065	6.638
pcler8	56	0.233	56	0.233	56	0.233	42	0.255	40	0.268	40	0.268	40	0.268	0.716
rd53	27	0.256	21	0.267	21	0.267	21	0.267	20	0.280	20	0.280	20	0.280	0.005
rd73	64	0.211	64	0.211	64	0.211	64	0.211	64	0.211	64	0.211	64	0.211	0.458
shiftc	52	0.312	52	0.312	47	0.347	47	0.347	47	0.347	47	0.347	47	0.347	0.048
sqrt8	26	0.199	26	0.199	26	0.199	26	0.199	26	0.199	26	0.199	26	0.199	0.098
squar5	32	0.315	32	0.315	27	0.336	23	0.380	23	0.380	23	0.380	23	0.380	0.025
tcon	34	0.411	34	0.411	25	0.560	25	0.560	24	0.583	24	0.583	24	0.583	0.018
ttt2	135	0.168	110	0.200	110	0.200	110	0.200	110	0.200	107	0.220	107	0.220	1.658
x2	66	0.144	66	0.144	47	0.157	47	0.157	34	0.179	33	0.183	30	0.197	0.038
xor5	6	0.083	6	0.083	6	0.083	5	0.100	5	0.100	5	0.100	5	0.100	0.005
z4ml	33	0.160	33	0.160	33	0.160	33	0.160	33	0.160	33	0.160	33	0.160	0.059
Z5xp1	91	0.191	72	0.195	61	0.202	61	0.202	61	0.202	61	0.202	61	0.202	0.198
o64	76	0.235	76	0.235	68	0.254	68	0.254	68	0.254	61	0.201	61	0.201	3251.13
apex5	986	0.312	986	0.312	986	0.312	986	0.312	986	0.312	986	0.312	986	0.312	9512.51
x3	812	0.116	812	0.116	710	0.165	710	0.165	710	0.165	702	0.171	702	0.171	7259.89

Table 3: Trade-off analysis area power-density.

w_1, w_2	Average % increase with respect to min. area	Average % increase with respect to min. Pow_den
0, 1	**72.39**	**1.00**
0.2, 0.8	37.05	1.30
0.4, 0.6	21.61	7.35
0.5, 0.5	16.00	11.33
0.6, 0.4	**9.17**	**13.84**
0.8, 0.2	3.77	15.92
1, 0	**1.00**	**20.13**

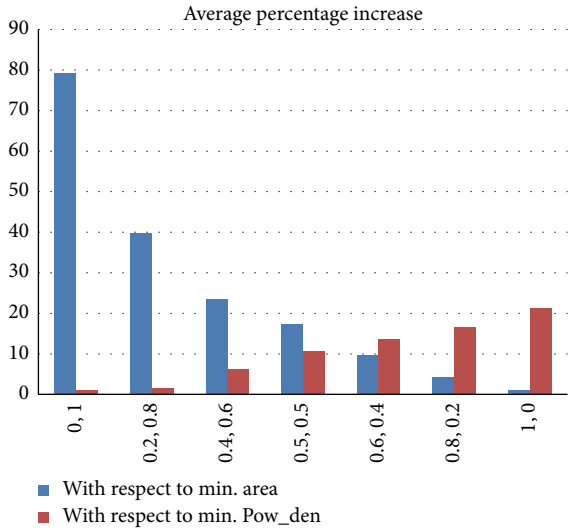

Figure 10: Average percentage improvement with respect to min. area and min. power-density.

Verilog hardware descriptive language (HDL) and they are fed as an input into CADENCE digital synthesis domain RTL Compiler (RC) for synthesis. This tool converts the RTL to standard cell based gate level netlist. The generated standard cell based gate level netlist is used in physical design level to generate the layout. The necessary inputs to perform synthesis are RTL, standard cell library, and constraints. The timing constraints information is provided using SDC file format. To perform the synthesis, we use the command "synthesis-to_mapped-effort medium" which combines the generic, mapped, and incremental synthesis and medium effort is given to synthesis process. The effort can be set to "low," "medium," or "high" depending upon the application area. By using the "report" commands, we can write out the results for area, power, and time utilization at synthesis level. After completion of synthesis, we need to write postsynthesis HDL (netlist file) and constraints generated into Verilog HDL and SDC file, respectively, for layout. Table 4 shows the postsynthesis analysis of the SRMDD AND-XOR circuit realization.

We have considered only the optimum output combinations with respect to area, power-density, and combination of

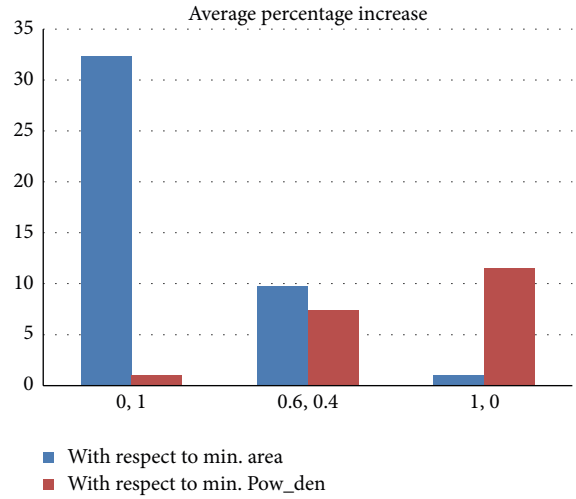

Figure 11: Average percentage improvement with respect to min. area and min. power-density (postsynthesis report).

both, that is, for the combinations ($w_1 = 1, w_2 = 0$), ($w_1 = 0, w_2 = 1$), and ($w_1 = 0.6, w_2 = 0.4$), respectively, for analyzing the postsynthesis results. The last two columns of the table report maximum CPU computation time (in seconds) and maximum delay (in nanoseconds). The CPU computation time is computed with the help of "get_attribute runtime" command and maximum delay is computed with the help of "report timing" command in RC. To sum up the result analysis, we have computed the average percentage increment with respect to optimum area and optimum power-density and tabulated the results in Table 5. The results of Table 5 are plotted in graph and shown in Figure 11. From the figure, it has been enumerated that if application is area specific, we can consider the weight ($w_1 = 1, w_2 = 0$) which is optimized area, but power-density in that case is at its maximum of 10.11% greater than its minimum value. For temperature specific application, we can go with the weight combination ($w_1 = 0, w_2 = 1$). In this case, power-density is at its minimum value but area is increased by about 30.18% compared with its minimum value. As observed from the graph, the optimum result considering area and temperature (power-density) is obtained at weight combination ($w_1 = 0.6, w_2 = 0.4$) for which area and power-density are 9.24% and 6.61% higher compared to their respective minimum value. Yet, we have not discussed the absolute temperature estimation. That has been discussed in the next section in physical synthesis level.

5.3. Physical Design Using CADENCE Encounter at 45 nm Technology. To obtain the actual silicon floor-plan area (in micrometer square) and absolute temperature (in degrees Celsius) of a logic circuit, we carry the design process to physical design domain of CADENCE encounter digital implementation (EDI) tool for physical implementation. For EDI, the input requirements are netlist, SDC (Synopsis Design Constraints) library, and physical details of the standard cell which is present in Library Exchange Format (LEF) file. Design netlist created from synthesis stage (as explained

Table 4: Postsynthesis analysis of SRMDD.

Benchmark circuits	$w_1 = 0, w_2 = 1$		$w_1 = 0.6, w_2 = 0.4$		$w_1 = 1, w_2 = 0$		Max_Delay (nS)	CPU time (s)
	Area (μm^2)	PD (nW/μm^2)	Area (μm^2)	PD (nW/μm^2)	Area (μm^2)	PD (nW/μm^2)		
5xp1	136	14.26	122	14.46	122	14.46	0.73	7
alu2	610	10.53	523	13.43	523	13.43	1.10	7
apex4	2790	13.25	2790	13.25	2790	13.25	1.99	9
clip	238	13.07	238	13.07	238	13.07	0.52	7
cm82a	12	13.76	7	14.03	7	14.03	0.28	6
cm162a	87	16.82	62	17.60	56	18.30	0.49	6
cm163a	104	12.00	45	14.47	28	15.04	0.56	7
duke2	379	10.21	255	12.86	255	12.86	1.18	7
ex5	397	11.29	397	11.29	189	14.73	0.78	7
f51m	132	15.16	131	15.36	131	15.36	0.55	7
inc	146	12.85	143	13.15	122	17.47	0.53	7
lal	125	12.39	125	12.39	125	12.39	0.71	7
misex1	88	10.88	69	12.56	57	13.75	0.68	6
pbo2	179	11.14	152	12.89	152	12.89	0.56	7
pcler8	156	9.46	142	11.54	142	11.54	0.49	7
rd53	33	14.42	26	12.89	26	12.89	0.37	6
rd73	93	19.21	93	19.21	93	19.21	0.34	6
shiftc	101	15.11	101	15.11	101	15.11	0.42	7
sqrt8	96	16.39	96	16.39	96	16.39	0.61	6
squar5	73	12.62	53	15.11	53	15.11	0.32	6
tcon	36	9.58	36	9.58	27	11.73	0.22	6
ttt2	147	14.76	137	15.46	137	15.46	0.54	7
x2	58	12.12	39	16.95	39	16.95	0.46	6
xor5	5	12.87	5	12.87	5	12.87	0.11	6
z4ml	18	12.68	18	12.68	18	12.68	0.32	6
Z5xp1	130	12.88	130	12.88	130	12.88	0.64	7
o64	98	12.46	91	11.36	86	10.07	0.73	9
apex5	561.6	8.13	561.6	8.13	561.6	8.13	1.10	15
x3	612.32	10.42	556.12	11.25	510.2	11.83	1.99	14

Table 5: Postsynthesis area power-density trade-off.

w_1, w_2	Average % increase with respect to min. area	Average % increase with respect to min. Pow_den
0, 1	30.18	1.00
0.6, 0.4	9.24	6.61
1, 0	1.00	10.11

in Section 5.2) was imported and after assigning the LEF file we configured the design analysis table. We have selected "macro" and "tech" LEF files at 45 nm technology provided by CADENCE for our analysis. In the design analysis table, we set the maximum and minimum delay library file, cap table, and technology library file at 45 nm technology. Placing of standard cell has been done in step Placement in design tool by using command "place standard cells." After setting all the criteria and placing standard cells, we have saved the floor-plan information, which will act as an input to HotSpot tool to calculate temperature profile. After designing, there are several verification processes under physical verification steps to rectify any errors generated while designing. The report generated after the process completion includes all the information about standard cell area (in micrometers), power dissipation by standard cell (in nanowatts), computational time required, and so forth for designing a particular circuit. The power information acts as another input file for HotSpot tool. The HotSpot tool considered the floor-plan and power profile information and provides the temperature profile in degrees celsius.

We reported the area in micrometer square (μm^2) and temperature in degrees celsius (°C) in Table 6. We have considered only the optimum output combinations, that is, (0, 1), (0.6, 0.4), and (1, 0), as explained in Section 5.2. Column with "std. cell area" gives the total silicon floor-plan area in micrometer square. The column with "Max_Temp" reported the maximum temperature generated by a logic

Table 6: Postlayout analysis of area (in μm^2) and temperature (in $^\circ$C) trade-off analysis.

Benchmark circuits	$w_1 = 0, w_2 = 1$		$w_1 = 0.6, w_2 = 0.4$		$w_1 = 1, w_2 = 0$		Max_Delay (nS)	Max CPU time (s)	Memory usage (MBytes)	Espresso (AND-OR)				
	Std. cell area (μm^2)	Max_Temp ($^\circ$C)	Std. cell area (μm^2)	Max_Temp ($^\circ$C)	Std. cell area (μm^2)	Max_Temp ($^\circ$C)				Std. cell area (μm^2)	Max_Temp ($^\circ$C)	Delay (nS)	CPU time (s)	Memory usage (MBytes)
5xp1	596.47	76.70	559.88	82.12	559.88	82.12	0.36	0.92	601.39	678.59	96.57	0.32	0.84	607.29
alu2	2646.63	64.05	2583.16	72.57	2583.16	73.98	0.54	1.74	602.25	2611.22	136.31	0.63	1.34	622.11
apex4	3997.00	96.12	3997.00	96.12	3997.00	96.12	0.57	1.97	622.46	13967.55	104.57	1.08	2.04	627.81
clip	229.73	77.86	229.73	77.86	229.73	77.86	0.57	0.87	617.29	507.90	102.74	0.58	0.92	624.77
cm82a	26.61	65.29	26.61	65.29	46.57	70.95	0.54	0.65	618.71	139.71	91.94	0.56	0.68	615.63
cm162a	495.63	74.63	143.04	82.65	119.75	87.97	0.56	1.03	613.19	272.77	96.57	0.55	1.00	615.64
cm163a	342.62	73.98	179.83	78.91	169.65	84.00	0.55	0.96	615.67	239.50	85.64	0.55	1.11	615.09
duke2	1350.52	84.51	868.19	96.57	868.19	96.57	0.58	1.18	623.47	2298.54	99.91	0.65	1.36	619.46
ex5	1470.27	98.50	705.20	109.36	705.20	109.36	0.59	1.63	621.88	2558.00	135.41	0.66	1.60	619.62
f51m	283.16	77.91	219.75	84.34	219.75	84.34	0.54	0.96	615.36	675.26	103.02	0.57	1.03	623.45
inc	683.16	67.93	559.81	80.44	543.04	82.30	0.38	0.84	601.83	728.48	112.25	0.39	0.90	610.10
lal	758.42	97.53	758.42	97.53	758.42	97.53	0.60	1.26	624.14	532.20	95.38	0.58	1.15	625.26
misex1	329.94	73.02	276.51	77.66	239.71	78.91	0.54	0.96	615.50	385.86	103.70	0.57	1.00	621.40
pbo2	1252.81	80.32	1169.65	85.14	1169.65	85.14	0.56	1.16	614.02	1270.69	119.33	0.60	1.15	622.60
pcler8	656.34	67.93	606.24	70.10	606.24	70.10	0.55	1.35	615.625	695.22	78.91	0.57	1.29	624.73
rd53	199.98	84.90	149.87	89.32	149.87	89.32	0.55	0.75	619.30	212.89	91.80	0.54	0.87	615.80
rd73	446.57	78.41	446.57	78.41	446.57	78.41	0.54	0.85	620.18	508.94	108.97	0.57	1.24	624.10
shiftc	372.56	101.84	372.56	101.84	372.56	101.84	0.41	1.15	610.12	372.56	85.64	0.38	1.15	608.89
sqrt8	375.88	91.82	375.88	91.82	375.88	91.82	0.54	0.85	620.42	352.60	91.82	0.57	0.88	623.80
squar5	249.89	73.98	243.24	75.94	243.24	75.94	0.53	0.94	621.22	352.60	95.66	0.56	0.90	625.80
tcon	33.22	67.93	33.22	67.93	23.14	92.32	0.55	0.65	613.60	6.56	106.53	0.54	0.62	621.40
ttt2	975.47	97.08	898.96	98.04	898.96	98.04	0.43	1.34	611.04	1024.50	95.66	0.40	1.58	605.02
x2	299.38	96.57	166.53	73.42	166.53	73.42	0.51	1.07	619.32	299.38	96.57	0.55	1.11	622.21
xor5	116.63	76.64	116.63	76.64	116.63	76.64	0.54	0.86	620.22	106.45	91.32	0.55	0.98	622.29
z4ml	226.61	93.67	226.61	93.67	226.61	93.67	0.54	0.79	620.11	209.56	97.50	0.55	0.84	617.63
Z5xp1	646.57	87.93	539.92	94.45	539.92	94.45	0.54	1.34	618.58	661.95	100.87	0.57	1.07	626.16
o64	96.90	81.70	90.32	82.45	85.84	84.17	0.78	3.56	823.14	88.59	92.38	0.72	0.84	851.96
apex5	559.86	88.56	559.86	88.56	559.86	88.56	2.12	2.83	808.09	658.72	103.12	1.63	1.34	836.29
x3	610.98	85.32	551.14	88.73	510.32	91.42	1.82	3.12	829.06	581.91	94.17	2.68	2.04	859.19
Average % savings with respect to Espresso	*−5.17*	*17.02*	*9.12*	*14.86*	*15.23*	*12.93*	*5.46*		*0.86*					

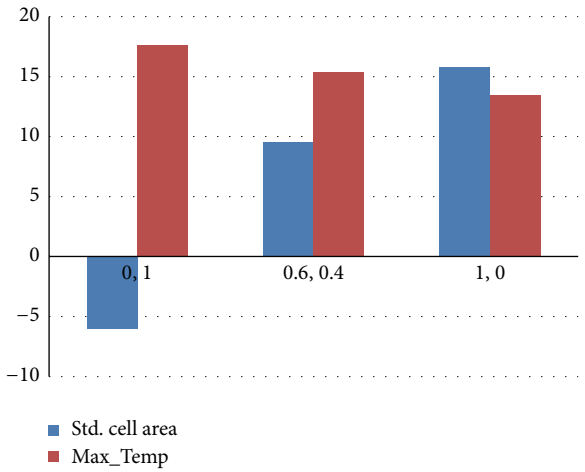

Figure 12: Average percentage savings with respect to area and temperature of AND-OR decomposed circuits (postlayout report).

design in degrees celsius. "Max_Delay" column shows the maximum time delay required by circuit in nanosecond among all combinations. The memory usage for implementation of the circuit is also reported in terms of megabytes in columns 10 and 15 for our approach and Espresso driven circuits. Columns 9 and 14 report the maximum CPU computation time (in seconds) required for implementing the circuits using "encounter." We involved three commands, "set initial_time", "set stop_time," and "set total_execution_time," to evaluate the CPU computation time. We have considered 29 benchmark circuits and compared the results with Espresso driven results which decompose the circuits into AND-OR based logic.

As observed from Table 6 and the graph shown in Figure 12, the best result at combination (0.6, 0.4) shows average percentage savings of 9.12% and 14.86% for standard cell area and maximum temperature, respectively. For the best area decomposition at combination (1, 0) shows 15.23% average savings for area and 12.93% average savings for maximum temperature. However, the best temperature combination (0, 1) shows 17.02% average savings for maximum temperature but 5.17% increase of standard cell area.

Here, a trade-off between area and temperature is shown. As we try to improve the area constraint, the temperature degrades and vice versa. We have also observed 5.46% average savings for maximum delay and 0.86% savings for memory usage in megabytes with respect to AND-OR dominated circuits.

6. Conclusion

We have presented a thermal-aware GA based heuristic approach for input polarity selection of variables in Shared Reed-Muller Decision Diagram based fixed polarity AND-XOR circuit decomposition. Efficient selection of input variables can reduce the total silicon area and temperature that has been enumerated in this paper. The paper also shows a trade-off between area utilized and temperature of the circuit. It also collaborates the logic synthesis level and physical design together. A range of solutions are achieved by varying weights of area and temperature values at logic synthesis level. The best results of logic synthesis level were brought into the physical design level and obtained a considerable improvement over AND-OR based circuits. The floor-plan and power profile information of each circuit is fed into the temperature estimation tool HotSpot for temperature estimation. In this work, we have reported 29 benchmark circuits. Within the 29 benchmark circuits, "x3.pla" has the largest input with 135 variables and 99 output functions. We have designed the algorithm dynamically. After reading the benchmark input circuit, the program assigns the memory location for bit manipulation required in different stages of optimization technique. To verify the above statement, we have run the algorithm with the benchmarks "o64.pla," "apex5.pla," and "x3.pla" which may be considered as large circuits in terms of input variables and output functions. Our algorithm successfully runs and gives optimized Shared Reed-Muller Decision Diagram based AND-XOR outputs. One important point to note is that CPU runtime depends on the frequency of the processor and memory available for usage. The same design can have different runtime in a system if parallel work is going on in the system. According to our survey, thermal-aware consideration in fixed polarity selection of AND-XOR circuit synthesis process has been done for the first time in this work. So, future work involves other XOR based circuit realizations for their thermal-aware realization like generalized Reed-Muller (GRM), mixed polarity Reed-Muller (MPRM), and pseudo Reed-Muller techniques and we are currently working on them.

Competing Interests

The authors declare that there are no competing interests regarding the publication of this paper.

Acknowledgments

This work was supported by the Fast Track Project (Ref. no. SB/FTP/ETA-341/2012) sponsored by the Science and Engineering Research Board (SERB), Department of Science and Technology (DST), Government of India, New Delhi 110016.

References

[1] L. Shang and R. P. Dick, "Thermal crisis: challenges and potential solutions," *IEEE Potentials*, vol. 25, no. 5, pp. 31–35, 2006.

[2] S. Gunther, F. Binns, D. M. Carmean, and J. C. Hall, "Managing the impact of increasing microprocessor power consumption," *Intel Technology Journal*, vol. 5, no. 1, pp. 1–9, 2001.

[3] K. Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaran-Arayanan, and D. Tarjan, "Temperature aware microarchitecture: extended discussion and results," Tech. Rep. CS-2003-08, Department of Computer Science, University of Virginia, 2003.

[4] R. K. Brayton, G. D. Hachtel, C. T. McMullen, and A. L. Sangiovanni-Vincentelli, *Logic Minimization Algorithms for VLSI Synthesis*, Kluwer Academic, Berlin, Germany, 1985.

[5] E. M. Sentovich, K. J. Singh, C. Moon, H. Savoj, R. K. Brayton, and A. Sangiovanni-Vincentelli, "Sequential circuit design using synthesis and optimization," in *Proceedings of the International Conference on Computer Design: VLSI in Computers and Processors (ICCD '92)*, pp. 328–333, IEEE, Cambridge, Mass, USA, October 1992.

[6] "ABC: A System for Sequential Synthesis and Verification," http://www.eecs.berkeley.edu/~alanmi/abc/.

[7] R. K. Brayton, G. D. Hachtel, and A. L. Sangiovanni-Vincentelli, "Multilevel logic synthesis," *Proceedings of the IEEE*, vol. 78, no. 2, pp. 264–300, 1990.

[8] L. Wang, *Automated synthesis and optimization of multilevel logic circuits [Ph.D. thesis]*, Napier University, Edinburgh, UK, 2000.

[9] T. Sasao, "AND-EXOR expressions and their optimization," in *Logic Synthesis and Optimization*, pp. 287–312, Springer US, 1993.

[10] U. Narayanan and C. L. Liu, "Low power logic synthesis for XOR based circuits," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (CAD '97)*, Digest of Technical Papers, pp. 570–574, San Jose, Calif, USA, November 1997.

[11] T. Sasao, *Switching Theory for Logic Synthesis*, Kluwer Academic, 1999.

[12] S. Chattopadhyay, S. Roy, and P. P. Chaudhuri, "Synthesis of highly testable fixed-polarity AND-XOR canonical networks—a genetic algorithm-based approach," *IEEE Transactions on Computers*, vol. 45, no. 4, pp. 487–490, 1996.

[13] S. N. Pradhan and S. Chattopadhyay, "Two-level AND-XOR networks synthesis with area-power trade-off," *International Journal of Computer Science and Network Security*, vol. 8, no. 9, pp. 365–375, 2008.

[14] T. Sasao and P. Besslich, "On the complexity of mod-2 sum PLA's," *IEEE Transactions on Computers*, vol. 39, no. 2, pp. 262–266, 1990.

[15] Y. Ye and K. Roy, "A graph-based synthesis algorithm for AND/XOR networks," in *Proceedings of the 34th Design Automation Conference*, pp. 107–112, ACM, June 1997.

[16] T. Sasao, "EXMIN2: a simplification algorithm for exclusive-OR-sum-of-products expressions for multiple valued input two-valued output functions," *IEEE Transaction on CAD*, vol. 12, no. 5, pp. 621–632, 1993.

[17] I. Reed, "A class of multiple-error-correcting codes and the decoding scheme," *Transactions of the IRE Professional Group on Information Theory*, vol. 4, no. 4, pp. 38–49, 1954.

[18] M. Davio, Y. Deschamps, and A. Thayse, *Discrete and Switching Functions*, George and McGraw-Hill, New York, NY, USA, 1978.

[19] A. Sarabi and M. Perkowski, "Fast exact and quasi-minimal minimization of highly testable fixed-polarity AND/XOR canonical networks," in *Proceedings of the 29th ACM/IEEE Design Automation Conference*, pp. 30–35, Anaheim, Calif, USA, June 1992.

[20] S. Aborhey, "Reed-Muller tree-based minimisation of fixed polarity Reed-Muller expansions," *IEE Proceedings: Computers and Digital Techniques*, vol. 148, no. 2, pp. 63–70, 2001.

[21] M. Pedram and S. Nazarian, "Thermal modeling, analysis, and management in VLSI circuits: principles and methods," *Proceedings of the IEEE*, vol. 94, no. 8, pp. 1487–1501, 2006.

[22] A. Das, S. R. Choudhury, B. K. Kumar, and S. N. Pradhan, "An elitist area-power density trade-off in VLSI floorplan using genetic algorithm," in *Proceedings of the IEEE 7th International Conference on Electrical & Computer Engineering (ICECE '12)*, Dhaka, Bangladesh, December 2012.

[23] A. Das and S. N. Pradhan, "Thermal aware output polarity selection of programmable logic arrays," in *Proceedings of the International Conference on Electronic Design, Computer Networks & Automated Verification (EDCAV '15)*, pp. 68–71, IEEE, Shillong, India, January 2015.

[24] A. Das and S. N. Pradhan, "Thermal aware FPRM based AND-XOR network synthesis of logic circuits," in *Proceedings of the IEEE 2nd International Conference on Recent Trends in Information Systems (ReTIS '15)*, pp. 497–502, Kolkata, India, July 2015.