Low Power Design of Digital Combinatorial Circuits with Complementary CMOS Logic

Padmanabhan Balasubramanian, C. Hari Narayanan, and Karthik Anantha

***Abstract*—**The increasing demand for low-power design can be addressed at different design levels, such as software, architectural, algorithmic, circuit, and process technology level [3]. This paper presents an effective approach to reduce power consumption of any arbitrary combinational logic circuit by applying transformations at the logic level, whilst preserving the desired functionality. We have considered implementation with static CMOS logic style, due to its robustness against device and process variations. Simulation results obtained using a 0.35 micron TSMC CMOS technology process report mean savings in power (without input correlation) by 25.46% over the best of existing methods; while considering spatio-temporal correlation in inputs, average power savings of 13.22% was obtained. The proposed method also enabled overall improvement in worst case delay parameter by 26.52%, over the best of other methods.

***Keywords*—**Boolean function, combinational logic, synthesis, low power design, standard cell based design, delay optimization.

I. Introduction

In the past, the major concerns of the VLSI circuit designers were area, speed and cost. In recent years, this has changed dramatically and power dissipation is being given increased weightage in comparison to area and speed design metrics. Power wall is a clear and present roadblock in the semiconductor industry [1]. The proliferation of portable and hand-held electronics combined with increasing packaging costs is forcing circuit designers to adopt low power design methodologies. Low power design of application specific integrated circuits (ASIC) result in increased battery life and improved reliability. Indeed, the Semiconductor Industry Association technology roadmap [2] has identified low power design techniques as a critical technological need. Hence it becomes imperative for circuit designers to acknowledge the importance of limiting power consumption and improving energy efficiency at all levels of the design hierarchy, starting from the lower levels of abstraction, when the opportunity to save power is significant.

Logic synthesis is a process by which an abstract form of desired circuit behaviour is turned into a design implementation in terms of logic gates. Algorithmic logic synthesis, in turn, is comprised of two stages – the technology independent phase where logic minimization is performed on the Boolean equations with no regard to physical properties and the dependent phase where mapping to a physical cell library is performed.

Circuit minimization problems have been extensively studied since the 1950s as important practical problems in logic synthesis. The number of papers on exact minimization algorithms and heuristics for sorting out essential prime implicants (implicates) are too great to survey here and so the reader is directed to [4] and [5] for an overview. Comparatively, little research has focused on the quality of implementation, resulting from the reduced solutions obtained. This paper addresses this issue after the final implementation stage. However, two-level logic circuits are of little importance in a VLSI design environment; most designs use multiple levels of logic. As a matter of fact, almost any circuit specification in register transfer level format or behavioral description is a multilevel representation. Typical practical implementations of a logic function utilize a multilevel network of logic elements, as multilevel synthesis usually produces the best cost effective realization of logic functions [14]. Next, this network is optimized using several technology-independent techniques before technology-dependent optimizations are performed. The typical cost function during the former stage is total literal count of the factored representation of the logic function; while at the latter phase, the simple cost estimate is replaced by more concrete, implementation-driven estimates during and after technology mapping, such as power, speed and area. In this context, the problem addressed in this paper is to realize a given combinational circuit using minimum number of basic library cells, with low power consumption, whilst satisfying the desired logic functionality, apart from maximizing throughput.

In this work, we consider power dissipation in non-regenerative logic designed using the standard-cell approach. One popular method employed to reduce design turn-around time is the use of standard cells. This design style accounts for 20%-50% of the total chip area of a modern processor, and a proportional amount of the power consumption. In a standard-cell based design, a logic circuit is constructed from a standard, pre-designed library of logic primitives. We would like to minimize the power consumption of a circuit realization, by judicious selection of the logic gates, made

Padmanabhan Balasubramanian is with the School of Computer Science, The University of Manchester, Lancashire, MAN M13 9PL UK (phone: +44-161-275 6294; e-mail: spbalan04@gmail.com, padmanab@cs.man.ac.uk).

C. Hari Narayanan is with the School of Electrical Sciences, Vellore Institute of Technology (University and IET, UK Accredited), Vellore, Vellore – 632 014 India (e-mail: harinara_21@yahoo.com).

Karthik Anantha is with the School of Electrical Sciences, Vellore Institute of Technology (University and IET, UK Accredited), Vellore, Vellore – 632 014 India (e-mail: karthik.anantha@gmail.com)

possible by a good synthesis scheme. To rationally validate the efficiency of our proposed approach in comparison with other methods, we consider identical gates of similar power, delay and area characteristics. This is because standard cell libraries typically contain multiple versions of the same logic function with differing strengths and physical parameters.

Depending on the application, the kind of circuit to be implemented, and the design technique used, different performance aspects become important, disallowing the formulation of universal rules for optimal logic styles. In this paper, these investigations are extended to a wider set of logic gates with respect to the latter, and with that, to arbitrary combinational circuits. The power dissipation characteristics of various existing synthesis methods for enabling low power design at the logic level are compared qualitatively and quantitatively by actual logic gate implementations and practical simulations under identical operating conditions. The examples and experimental results obtained succinctly demonstrate the effectiveness of our proposition.

The remaining part of this paper is organized as follows. Section 2 highlights the reasons underpinning the choice of the logic style considered and its typical advantages. Section 3 deals with the issue of power dissipation in CMOS circuits. Section 4 surveys the current literature pertaining to reduction of switching activity and their inherent limitations. Section 5 elucidates our proposal by means of specific examples drawn from previous literature and illustrates how the proposed method enables savings in actual power consumption. Section 6 presents the results obtained for some case studies. In section 7, we discuss the results and finally conclude.

II. Complementary CMOS logic style

Complementary CMOS logic or Static CMOS logic style consisting of complementary nMOS pull-down and pMOS pull-up networks to drive '0' and '1' outputs are used for the vast majority of logic gates in digital integrated circuits. They have good design margins, fast, low power, insensitive to device variations, easy to design, widely supported by commercial CAD tools, and readily available in standard cell libraries. When noise does not exceed the margins, the gate eventually will settle to the correct logic level. Indeed many ASIC methodologies allow only complementary CMOS circuits. Even custom designs use static CMOS for 95% of the logic [12]. They also enable low leakage designs owing to their inherent flexibility to accommodate leakage control transistors at the junction between the pull-up and pull-down network nodes [16].

Other advantages of static CMOS logic style are its robustness [12] against voltage scaling and transistor sizing and thus ensuring reliable operation at low voltages and arbitrary transistor sizes. Input signals are connected to transistor gates only, which facilitates the usage and characterization of logic cells. The layout of CMOS gates is straightforward and efficient due to the complementary transistor pairs. Given the correct inputs, it will eventually produce the correct output so long as there were no errors in logic design or manufacturing. Static CMOS logic also has the advantage that there is no precharge/predischarge operation and charge sharing does not exist. Other circuit families tend to become prone to numerous pathologies, including charge sharing, leakage, threshold drops and ratioing constraints. Basically, CMOS fulfils all the requirements regarding the ease-of-use of logic gates.

III. Power dissipation in CMOS circuits

Average power dissipation (P_{avg}) in CMOS digital circuits can be expressed as the sum of three main components [3], which are summarized in the following equation, as

$$P_{avg} = P_{short\text{-}circuit} + P_{leakage} + P_{dynamic} \quad (1)$$

$P_{short\text{-}circuit}$ is the power from stacked P and N devices in a CMOS logic gate that are in the ON state simultaneously. This happens briefly during switching. This type of power dissipation can be controlled by minimizing the transition times on nets. It usually accounts for 15%-20% of the overall power dissipation.

$P_{leakage}$ is the power dissipation due to spurious currents in the non-conducting state of the transistor. This component becomes a larger problem as device geometries shrink and transistor threshold voltages (V_t) drop. Leakage current depends upon the supply, V_{dd} (or how close it is with respect to V_t), V_t itself, transistor aspect ratio (W/L) and temperature. As the supply voltage scales down with technology, this increases exponentially and is construed to dominate the total power dissipation in ultra deep submicron technologies. Increasing die area also increases the leakage power adversely, as this increases the number of transistors.

$P_{dynamic}$ is the dynamic power dissipation, also called the switching power. This is the dominant source of power consumption in CMOS system-on-chip (SoC), accounting for roughly 75% of the total. It is generally represented by the following approximation,

$$P_{dynamic} = \alpha \,.\, C_L \,.\, V_{dd}^2 \,.\, f_{clk} \quad (2)$$

where 'α' is the switching activity factor (also called transition probability) and it tends to increase as the need for bandwidth increases, 'C_L' is the overall capacitance to be charged and discharged in a reference clock cycle. Technology scaling has resulted in smaller transistors and hence smaller transistor capacitances, but interconnect capacitance has not scaled much with process and has become the dominant component of capacitance. 'V_{dd}' is the supply voltage. Though voltage scaling has the biggest impact on power dissipation (nearly quadratic savings in power), this generally comes at an expense of an increase in delay. 'f_{clk}' is the switching frequency of a global clock for a globally synchronous design, local clock for a locally synchronous design or the input arrival rate in case of a pure static system.

IV. Switching Activity Reduction

Significant work has been carried out in developing efficient techniques to estimate the switching activity of a CMOS circuit earlier. These techniques can be divided into two classes: statistical techniques (also called dynamic techniques) [6] and probabilistic (or static) techniques [7].

Statistical techniques simulate the circuit repeatedly until the power values converge to an average power, based on statistical measures. Probabilistic techniques propagate input statistics through the circuit to obtain the switching probability for each gate in the circuit. Though both the above techniques exist, the static techniques enable a quick approximate estimate of the power consumption of a digital integrated circuit at the logic level, without the need for extensive simulation. This paper focuses on addressing both these aspects of power estimation in non-regenerative logic circuits, with representative samples.

In [8] Brzozowski et al. propose a method to minimize the total switching activity at all the gate output nodes. The method first relies upon the conventional Map method to minimize the given function into two-level logic form. The reduced sum-of-products (SOP) and product-of-sums (POS) expression for the given function is then realized using two-level logic. Since complementary input inverters are not assumed to present, as is the case with cell-based IC designs, it would essentially be a three-level logic circuit. The switching activity is then calculated at all the gate output nodes by multiplying the cardinality of the ON-set and OFF-set elements corresponding to the truth table for each gate and subsequently doubling it by 2. Based upon which realization has a lower switching activity, the reduced Boolean equation governing that particular expression is then subjected to transformations (entirely based on De-Morgan's laws of Boolean algebra) to obtain a final expression, which guarantees that all the inverters associated with the primary input variables of the function are eliminated. The final resulting circuit would be realized in terms of multilevel logic using NAND, NOR and inverter logic gates. In [9], they vary the input probabilities to arrive at an estimation of switching activity. The bottom-line in [8] and [9] is to lower the switching activity by eliminating input inverters and thereby lower the power dissipation of the circuit designed.

However, this method has some disadvantages. Firstly, switching activity is computed as an integer measure. This computation would become expensive in terms of computational time for designs with several logic gates and the problem would be complicated when the number of levels of logic also increases, beyond two or three-levels. For uniform input distribution, the computation of switching activity in terms of an integer measure for the initial three-levels might be less expensive, but for non-uniform input distribution, it would be cumbersome and become difficult. In [9], the authors have taken into account non-uniform input distribution to measure the activity rate. Also no explicit reference is provided with respect to the fan-in of the gates that should be considered for final realization. This may undermine the delay of the logic implementation even though the number of levels is a minimum. Also no direct relationship can be ascertained between this integer measure and the final power dissipation.

Finally, interpretation of activity rate as an integer measure does not seem to take into account, a reference clock frame. Furthermore, this logic does not seem to hold good for all circuits even after transformations have been applied, as is evident from the following example. For e.g. let the ON-set of a 4-variable Boolean function, F, be given by {0,1,2,3,4,5,8,9,11,12,13,15}. Direct and conventional three-level implementation of the reduced SOP and POS forms would have an integer activity measure of 544 and 592 respectively. From this, we understand according to [8] and [9], that SOP form is promising. After applying Boolean transformations, we find that the SOP and POS forms have an integer measure of 496 and 400 respectively. This goes to prove that estimation of activity rate as an integer measure does not seem to be a viable solution as it contradicts the original assumption. However, though the transformations mentioned in [8] and [9] are helpful in reducing the actual power dissipation in certain cases vis-à-vis trade-off with performance, physical power estimate is not mentioned.

In [10] and [11], Menon et al. propose a solution by addressing this issue using a probability based estimate for the switching activity of a gate output node, based on the probabilities associated with the primary inputs of the circuit. This method seems to be reasonable as it could even take into account the correlation between the input signals to estimate the activity rate and this seems to be a rational approach. This is because the switching activity would vary depending on the input probability distribution and the activity rates at the gate output nodes are dictated by the input signal probabilities. This sort of probabilistic measure would automatically hold good even in the presence of a global clock.

To avoid the input inverters that could be necessary for a certain logic design, they rely on a different, non-orthodox grouping order within the K-Map constructed for a logic function, such that the number of groups would be identically similar to that of the original solution and input inverters also get eliminated. This unnecessarily expands the global search space and may make the problem NP-hard. Also in [11] they propose an increase in the number of inputs towards realizing this objective. This does not seem to in line with the technology considerations and fan-in limitations of modern VLSI design. Further, this approach would not be able to utilize the best two-level minimized expressions, resulting from standard minimizers and hence finding a solution for problems of higher-magnitude and with multiple outputs may complicate the whole process of finding a power optimized solution and in many cases, impossible altogether. To elaborate on this, we provide an example in the next section.

This methodology would tend to adversely affect the delay of the final circuit due to accommodating more inputs and it may lead to serious charge sharing and charge redistribution problems in dynamic logic implementation, which may finally result in erroneous output states [3].

V. Proposed Heuristic for Low Power

We propose a method to reduce the power consumption of the logic circuit realization. Apart from placing too much emphasis on static estimates (especially with respect to reducing activity rate), we rely on the physical level power estimate of the final circuit implementation by subjecting it to simulation with distinct input patterns (with and without spatio-temporal correlation), at the transistor level. Through this, we understand that the final word should be on the basis of a dynamic estimation technique, although static estimation serves only as an aid and does not portray the actual figure. Our proposition also reduces the total gate switching activity of the circuit, which helps in reducing the average power dissipation of the circuit. However, we observe that at the circuit level, a strong relationship between switching activity and total power dissipation does not seem to hold good for all cases. We have found that on an average, the proposed technique ensures savings in total gate switching activity, but the percentage savings gained with respect to this parameter does not tend to correlate with the actual savings in power at the circuit level. This may be due to an increase in any of the other power components, besides switching power.

A. Algorithm

- Given a logic function, Z, use a standard two-level logic minimizer, such as ESPRESSO [13] to obtain an initial reduced form in two-level logic
- Obtain the reduced form corresponding to the POS form as well
- If there is no sharing of variables in the constituent terms of a minimized POS expression, transform it using De-Morgan's law, such that the sum of complemented variables is given as complement of the product of those variables (This would hold good for any variable order)
- If there is no sharing of variables in the constituent terms of a minimized SOP expression, do not transform it using De-Morgan's law into a complement of the product of those variables in normal form (This will ensure that the final realized circuit would inherently contain more NAND cells instead of NOR cells, which is required of a high-speed design. Besides it enables a power optimized solution even though inverters tend to get associated with some primary input variables)
- If there is a sharing of variables in the reduced SOP expression of a function, then perform algebraic factorization. This will reduce the literal count. Then apply De-Morgan's transformations appropriately
- This procedure would hold good for minimized POS expressions as well, so that a solution which would contain less inverters could be obtained
- Additionally for the POS form, distributive axiom [(a+b) (a+c) = (a+bc)] could be applied to reduce the literal count. For the SOP form, the distributive axiom [(ab + ac) = a(b+c)] is also a factoring operation
- Obtain the literal count of the minimized and transformed SOP and POS expressions. If the latter has a lesser literal count than the former, then convert the originally minimized two level POS expression into its logically equivalent SOP form by straightforward conversion. This constitutes an *output phase optimization (OPO) step* for our method and is basically a result of the complementary function approach. Then subject it to the same sequence of operations as was done for the original two-level SOP form
- Realize the final expressions corresponding to both the minimized and transformed SOP and POS form/ output phase optimized SOP form, in line with the directed acyclic graph specification (DAG) of a logic function, such that the nodes of the DAG would represent logical operators with only two inputs. Sharing of operators between two or more terms could be considered (Earlier literature [8] [9] [10] and [11] did not correspond to a binary DAG specification)
- The number of DAG nodes could be traded-off for fan-out. In order to support a node with a fan-out of more than one, buffer(s) may be inserted at such fan-out's, as deemed appropriate (This will ensure signal integrity for multilevel designs)
- Convert the DAG level representation in terms of standard gate level primitives viz. NAND, NOR and NOT library cells with minimum fan-in.
- Obtain the literal count, node count and cell count of each of the structures and make a decision regarding the final structure based on these
- In case one of the metrics is the same, a decision would be made on the basis of other two. In case of a tie in any means, proceed with dynamic estimation of power using input patterns (For most of the cases, we observed that the circuit corresponding to one with less literal count, node count and cell count consumed the least power)

From the above algorithm, we can infer that two solutions are possible at the technology-independent optimization stage. However, we recommend that the choice of a final circuit should be driven by realistic parameter estimation (based on simulation) rather than static estimation. This is because, for a majority of the cases, our algorithm has yielded the best results even after the technology-mapping phase; while in a few cases, it was contrary. The resulting circuits are subjected to a representative input sequence with and without correlation to determine physical power and worst case delay.

We consider some of the problems mentioned in previous literature [8], [9], [10] and [11] and compare with our method.

Let us define TGSA as the summation of the switching activity at all the gate output nodes, calculated based on the input signal probabilities. Unless otherwise specified, this computation is based on a uniform input distribution i.e., for the case without correlation. DC stands for device count or the number of transistors required for implementation with static CMOS logic style. TNI refers to the total number of interconnects between the primary inputs and primary output(s). Average power dissipation is represented by P_{avg} and it is obtained for two cases, an input sequence without correlation between inputs (without C) and the other with input correlation (with C). The worst case delay (Delay) is mentioned in ns. A supply of 3.3V and an input frequency of 100 MHz are used to perform all the simulations with a 0.35 micron technology node (TSMC CMOS process).

1) Case 1: F = ∑m (1,5,7,8,9,10,15) [8] [9]

The solution corresponding to Brzozowski et al. in multilevel form corresponding to a binary DAG specification would appear at the gate level as shown in Fig. 1. This would be the same for the DAG specification of the solution, obtained by Menon et al. [11]. The gate level designs obtained as per the proposed method corresponding to the real and output phase optimized ones are depicted by Figures 2 and 3 respectively.

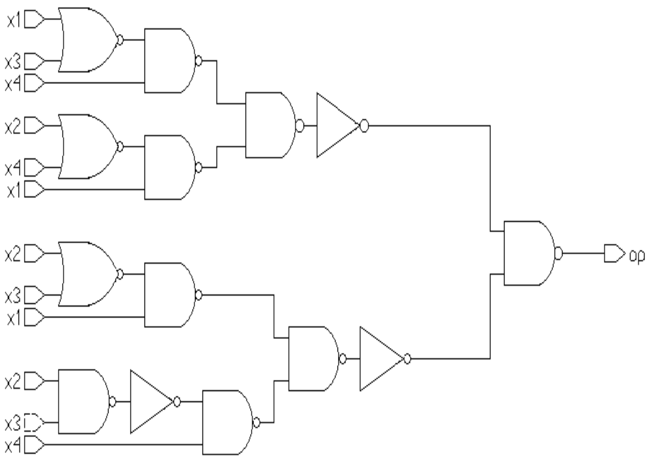

Fig. 1 DAG aware circuit for Fig. 6 of [8]

Henceforth, we will explicitly use the DAG specification, as the starting point for construction of any combinational logic circuit.

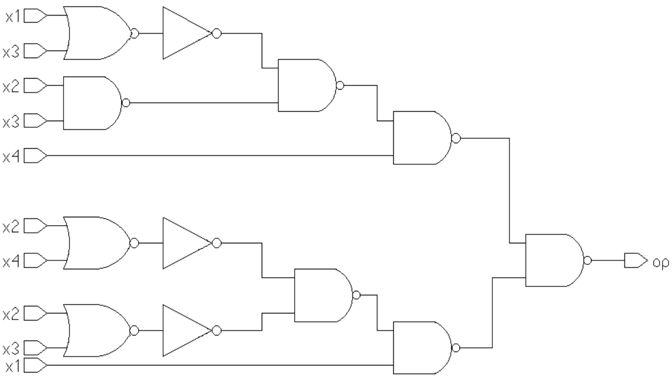

Fig. 2 Circuit corresponding to real function phase

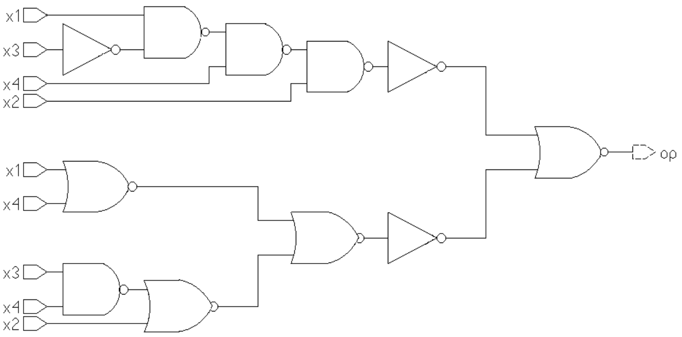

Fig. 3 Circuit with output phase optimization

The simulation results corresponding to Fig. 6 of [8] and Figures 2 and 3 are given in the following table.

TABLE I
COMPARISON OF DESIGN METRICS FOR VARIOUS METHODS

Realization	TGSA	DC	TNI	P_{avg} Without C (nW)	P_{avg} With C (nW)	Delay (ns)
Fig. 6 [8]	1.246	38	19	3.01	2.47	0.32
Brzozowski et al.	2.339	50	25	3.72	2.48	0.35
Menon et al.	2.339	50	25	3.72	2.48	0.35
Proposed (SOP)	2.393	42	21	3.08	2.37	0.29
Proposed (OPO)	2.283	38	19	2.34	2.58	0.35

From the above table, it becomes clear that TGSA also does not correlate with power consumption quite well (see the values of Fig. 6 [8] and proposed (OPO). This contradiction can also be observed in the following examples.

Henceforth, we would only consider the best solution derived based on the proposed method, for inclusion in the comparison table.

2) Case 2: F = ∏M (3,6,7,8) [8]

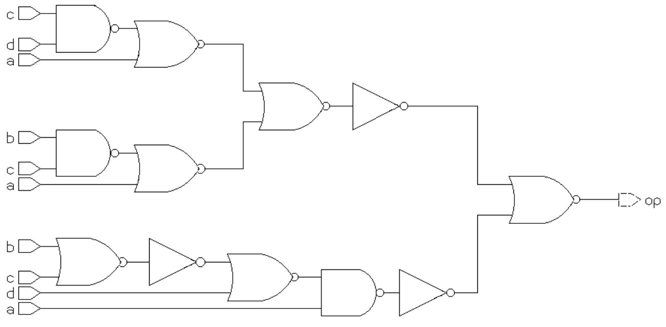

Fig. 4 DAG aware circuit for Fig. 8 [8] – Brzozowski et al.

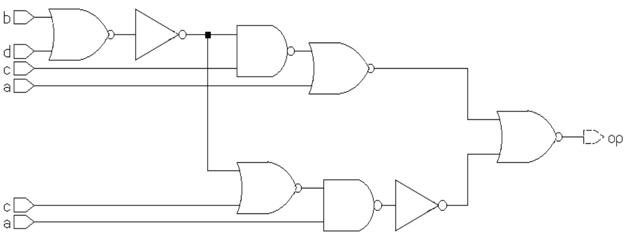

Fig. 5 Circuit synthesized based on proposed heuristic

TABLE II
COMPARISON OF DESIGN METRICS FOR VARIOUS METHODS

Realization	TGSA	DC	TNI	P_{avg} Without C (nW)	P_{avg} With C (nW)	Delay (ns)
Fig. 8 [8]	0.809	34	17	4.49	2.74	0.32
Brzozowski et al.	1.808	42	21	3.08	2.36	0.42
Menon et al.	←	does	not	converge	fully	→
Proposed (POS)	1.176	28	14	2.33	1.39	0.32

In this example, Menon et al. algorithm [11] would not be able to provide a feasible solution fully void of primary input inverters, without increasing the number of groups. Hence we have mentioned that the algorithm has not converged.

3) *Case 3: F = ∑m (1,3,4,5,9,11,13,15) [10]*

TABLE III
COMPARISON OF DESIGN METRICS FOR VARIOUS METHODS

Realization	TGSA	DC	TNI	P_{avg} Without C (nW)	P_{avg} With C (nW)	Delay (ns)
Fig. 2 (b) [10]	1.031	26	13	2.0118	1.95	0.488
Brzozowski et al.	←	does	not	converge	fully	→
Menon et al.	1.482	30	15	2.337	1.93	0.314
Proposed (SOP)	1.094	22	11	1.902	1.62	0.215

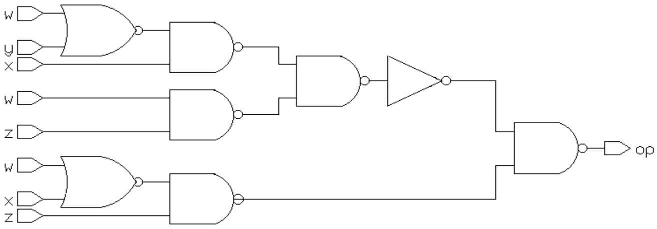

Fig. 6 DAG aware circuit for Fig. 2(b) of [10] – Menon et al.

In this case, Brzozowski et al. algorithm [8] [9] does not eliminate all the primary input inverters. Hence the algorithm has not fully converged to obtain a feasible solution.

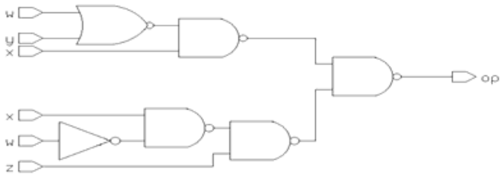

Fig. 7 Proposed circuit realization

4) *Case 4: F = ∑m (0,1,4,5,6,7,9,13,15) [11]*

TABLE IV
COMPARISON OF DESIGN METRICS FOR VARIOUS METHODS

Realization	TGSA	DC	TNI	P_{avg} Without C (nW)	P_{avg} With C (nW)	Delay (ns)
Fig. 3 (b) [11]	1.215	40	16	2.98	4.17	0.344
Fig. 4 (b) [11]	1.113	44	18	3.19	4.55	0.318
Brzozowski et al.	←	does	not	converge	fully	→
Menon et al. – Fig. 3(b) [11]	2.387	42	21	3.59	2.06	0.354
Menon et al. – Fig. 4(b) [11]	2.730	54	27	3.81	2.28	0.442
Proposed (OPO)	1.367	18	9	1.59	0.56	0.255

In Table IV, in the first two rows, TNI is not half of DC. This is because the circuits have been realized using gates instead of cells for some sub-functions in [11].

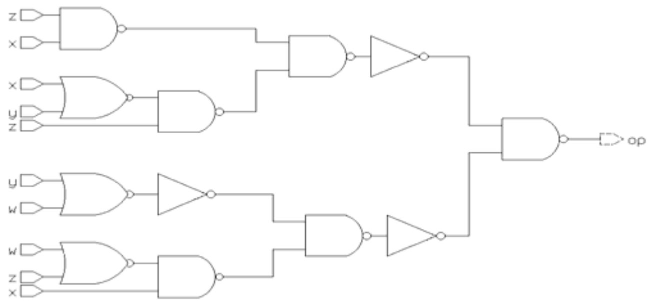

Fig. 8 Menon et al. DAG aware realization for Fig. 3(b) [11]

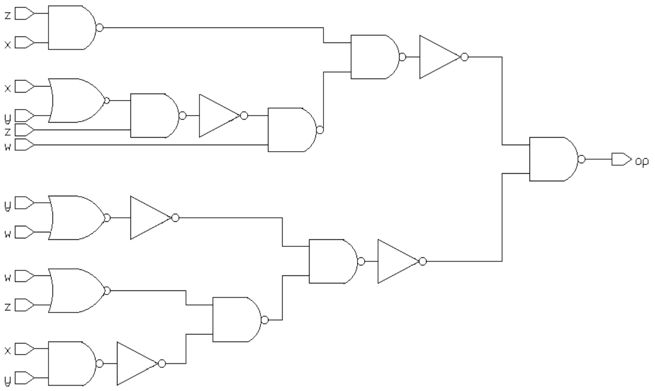

Fig. 9 Menon et al. DAG aware realization for Fig. 4(b) [11]

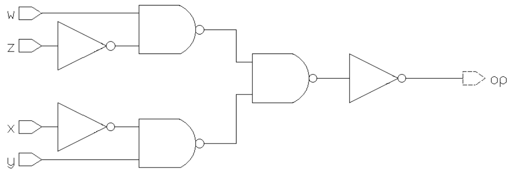

Fig. 10 Proposed circuit realization with OPO

VI. PROBLEM CASES

We now consider examples to highlight scenarios where Brzozowski et al. and Menon et al. algorithms would not properly converge to yield a power optimized solution. In other words, we make it clear that there are a number of function cases, where primary input inverters cannot be eliminated based on the methods suggested in [8 [9] [10] and [11]. A couple of them are listed here and are described by the following examples.

A. Problem 1: F(a,b,c,d) = ∑m (2,3,7,9,11,13) +d(1,10,15)

We consider the above incompletely specified logic function to illustrate a situation, where Brzozowski et al., algorithm could not eliminate the primary input inverter. The simulation results for the circuits realized according to the algorithm in [11] and the proposed heuristic are mentioned in Table V.

$$F_{\text{Brzozowski et al.}} = b'c+cd+ad \quad (3)$$

$$F_{\text{Menon et al.}} = ad+cd+(b+d)'c \quad (4)$$

$$F_{\text{Proposed}} = ad+c(b.d')' \quad (5)$$

As seen from (3), it is clear that [8] and [9] would not be able to eliminate the inverter associated with the primary input ‘b’. There are many functions for which [8] and [9] would not converge to obtain a solution free of primary input inverters.

TABLE V
COMPARISON OF DESIGN METRICS FOR THE TWO SOLUTIONS

Realization	TGSA	DC	TNI	P_{avg} Without C (nW)	P_{avg} With C (nW)	Delay (ns)
Menon et al.	1.414	26	13	1.495	1.092	0.318
Proposed (SOP)	1.109	18	9	1.060	0.973	0.308

B. Problem 2: F(a,b,c,d) = ∑m (0,1,3,4,7,12,14,15)

In this case, Menon et al. algorithm [11] would not be able to provide a power optimized solution without increasing the number of groups. The resulting reduced expressions are,

$$F_{\text{Brzozowski et al.}} = (a+b+c)'+b(c+d)'+(a+(cd)')'+abc \quad (6)$$

$$F_{\text{Menon et al.}} = a'b'c'+bc'd'+a'b'd+a'cd+abc \quad (7)$$

$$F_{\text{Proposed}} = [(ab)' + (c+(bd)')' + (a+d)'c]' \quad (8)$$

As seen from (7), [11] does not converge in obtaining a fully optimal solution as is evident from the 4th term in (7). However, a solution with minimum switching activity could still be obtained using [11], only by increasing the number of groups, which is contrary to the actual methodology of [11].

TABLE VI
COMPARISON OF DESIGN METRICS FOR THE TWO METHODS

Realization	TGSA	DC	TNI	P_{avg} Without C (nW)	P_{avg} With C (nW)	Delay (ns)
Brzozowski et al.	2.749	56	28	3.83	2.74	0.421
Proposed (SOP)	1.726	34	17	2.66	1.21	0.359

TABLE VII
BOOLEAN FUNCTION SPECIFICATION

Function ID	Logic function specification
1	F = ∑m (5,7,8,9,10,15)
2	F = ∑m (0,10,12,15) + d (4,11)
3	F = ∑m (1,4,5,12,13,14,15)
4	Z_{ON} = {000222,222000}
5	Z(a,b,c,d,e,f,g) = ∑m (0,3,13,15) + d (1,2,7,9,14) + e(m_6 + m_8) + f(m_{12}) + g(m_5)
6	F = ∑m (1,4,6,10,20,22,24,26) + d (0,11,16,27)
7	F = ∑m (0,2,14,16,24,26) + d (18,28,30)
8	Z_{ON} = {000222222,222000222,222222000}
9	Z_{ON} = {000222222222,222000222222, 222222000222,222222222000}
10	Z_{ON} = {000222222222222,222000222222222, 222222000222222,222222222000222, 222222222222000}
11	Z_{ON} = {000222222222222222,222000222222222222, 222222000222222222,222222222000222222, 222222222222000222,222222222222222000}

Table VII lists the other problems cases considered to validate our argument through simulation results.

In this table, functions 4, 8, 9, 10 and 11 are represented by their ON-set, where ‘0’ indicates a variable present in complementary form and ‘2’ indicates the absence of a corresponding input variable in the term. Average power dissipation comparison for different methods without correlation between the inputs is shown in fig. 11. Average power dissipation for the various methods with spatio-temporal correlation existing between the inputs [15] is shown in fig. 12. The worst case delay metric for the different schemes is highlighted in fig. 13.

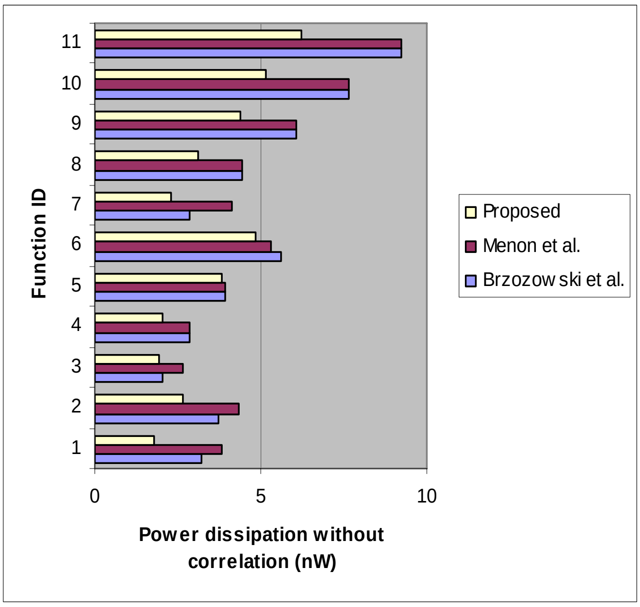

Fig. 11 Average power dissipation comparison without input correlation

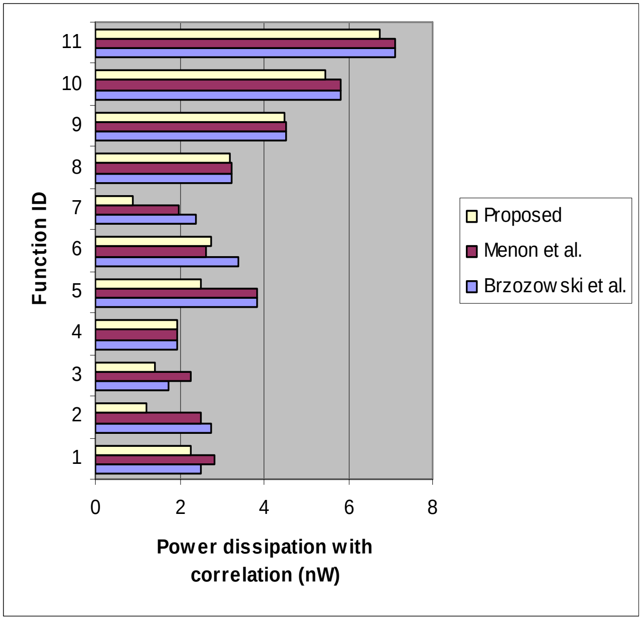

Fig. 12 Average power dissipation comparison profile with spatio-temporal correlation between inputs

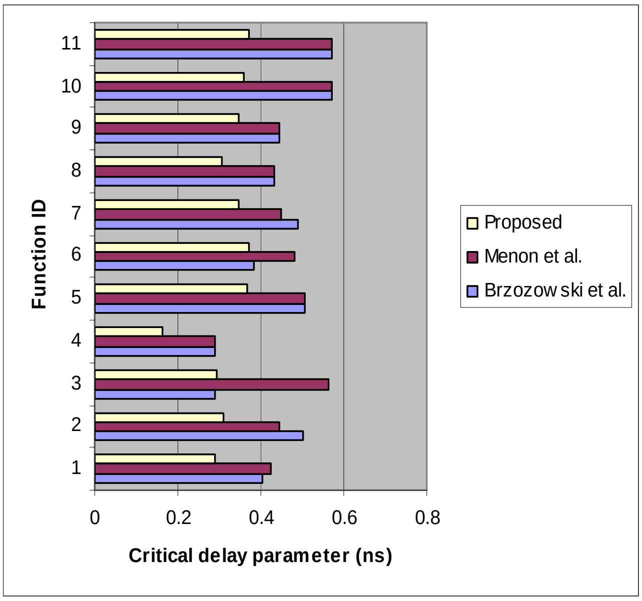

Fig. 13 Critical path delay comparison for different methods

Total gate switching activity (TGSA), which is computed as sum of the switching activity at all the gate output nodes for a uniform input distribution, for all the methods is depicted by fig. 14. The comparison of device count (in terms of the number of transistors required for implementation with static CMOS logic) and total number of interconnections (determined as the sum of all the fan-in of all the gates in the Boolean network), for the different methods is depicted by fig. 15 and fig. 16 respectively.

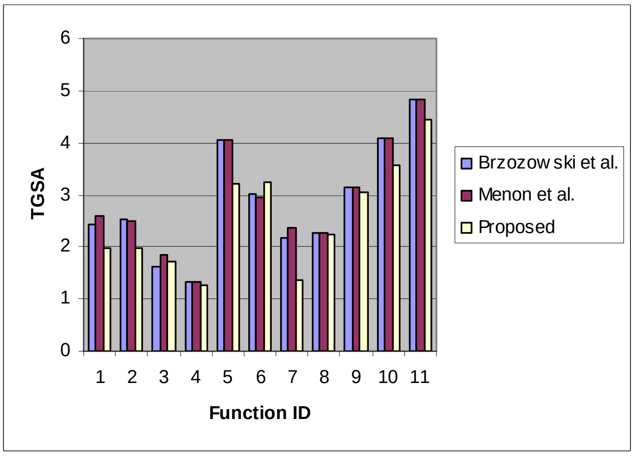

Fig. 14 Total gate switching activity comparison for various methods

VII. Discussion of Results and Conclusion

Significant number of combinatorial logic functions has been considered for simulation studies and a representative sample has been listed in Table VII. At the technology-independent stage, the DAG aware realizations for the minimized solutions based on three different methods were obtained. At this stage, the proposed methods resulted in average savings in TGSA by 10.8% over the best of other methods; while in terms of DC and TNI, mean savings of 19.28% was reported. The savings for DC and TNI are found to be the same, as TNI is half of the device count for a DAG aware logic circuit realization.

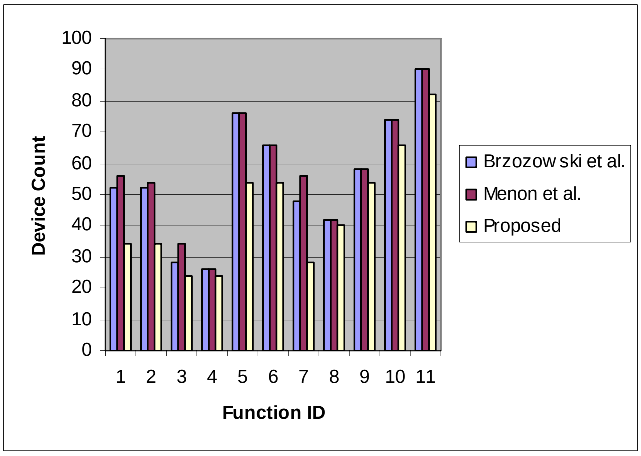

Fig. 15 Device count comparison for the different methods

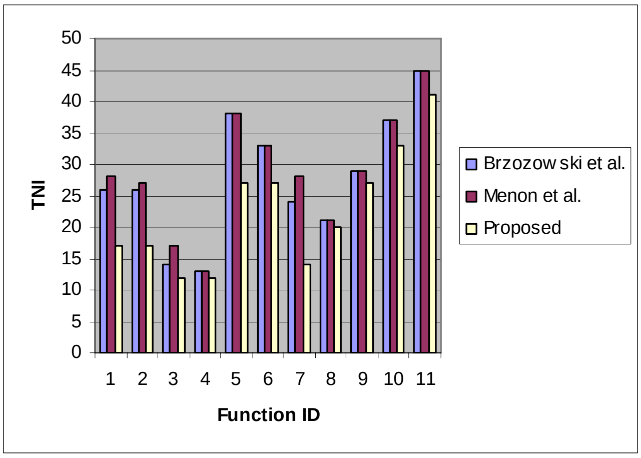

Fig. 16 Total number of interconnects for the circuits synthesized according to the various methods

After technology mapping with a 0.35 micron TSMC CMOS process for a supply of 3.3V and an input frequency of 100MHz, an overall savings in average power dissipation by 25.46% was obtained for the circuits synthesized according to the proposed method in comparison with the best of the values obtained for those of the other methods [9] [11], without input correlation. Considering spatio-temporal correlation to exist between the inputs, the proposed heuristic effected power savings of 13.22% over the other methods. The percentage improvement in worst case delay is also considerable, which has been 26.52%, overall. The simulation results have been obtained using Mentor Graphics tools on a Linux platform.

In this paper, we have proposed a heuristic for low power design of digital combinatorial circuits. The approach is versatile and valid for any arbitrary non-regenerative logic function(s). The approach is generic and does not claim eliminating all the inverters that could be associated with the primary inputs of a circuit, which may not be possible for many functions. Instead, it first tries to optimize the logic at the technology-independent stage and then further optimizes after translating the designs in terms of DAG specifications. De-Morgan's laws are applied recursively so that the final circuit realization is interpreted only in terms of cells, compatible with a standard technology library. The proposed solution is also in favour of NAND logic instead of NOR logic in order to ensure good performance, whilst reducing power dissipation. In addition, the procedure does not deviate from the conventional two-level logic minimization schemes for single/multiple outputs, but only emphasizes on transforming it in order to enable a power aware synthesis solution at the logic level.

REFERENCES

[1] T. Kuroda, "Low-power high-speed CMOS VLSI design," *Proc. of IEEE International Conf. on Computer Design*, 2002, pp. 310-315.

[2] Semiconductor Industry Association (SIA) ITRS Roadmap, Available: http://www.sia-online.org/backgrounders_itrs.cfm

[3] A.P. Chandrakasan, S. Sheng, and R.W. Broderson, "Low-power CMOS digital design," *IEEE Journal of Solid-State Circuits*, vol. 27(4), April 1992, pp. 473-484.

[4] O. Coudert, "Two-level logic minimization: an overview," *Integration-The VLSI Journal*, vol. 17(2), October 1994, pp. 97-140.

[5] S. Devadas, A. Ghosh, K. Kuetzer, *Logic Synthesis*, Mc-Graw Hill Series, 1994.

[6] R. Burch, F.N. Najm, P. Yang, and T.N. Trick, "A Monte Carlo approach for power estimation," *IEEE Transactions on VLSI Systems*, vol. 1(1), March 1993, pp. 63-71.

[7] A. Ghosh, S. Devadas, K. Kuetzer, and J. White, "Estimation of average switching activity in combinational and sequential circuits," *Proc. of 29th ACM/IEEE Design Automation Conference*, 1992, pp. 253-259.

[8] I. Brzozowski, and A. Kos, "Minimization of power consumption in digital integrated circuits by reduction of switching activity," *Proc. of 25th Euromicro Conference,* 1999, vol. 1, pp. 376-380.

[9] I. Brzozowski, and A. Kos, "Energy consumption minimisation with new synthesis method," *Proc. of 7th IEEE International Conf. on Electronics, Circuits and Systems*, 2000, vol. 1, pp. 605-608.

[10] R.V. Menon, S. Chennupati, N.K. Samala, D. Radhakrishnana, and B. Izadi, "Power optimized combinational logic design," *Proc. of 2003 International Conf. on Embedded Systems and Applications*, 2003, pp. 223-227.

[11] R.V. Menon, S. Chennupati, N.K. Samala, D. Radhakrishnana, and B. Izadi, "Switching activity minimization in combinational logic design," *Proc. of 2004 International Conf. on Embedded Systems and Applications*, 2004, pp. 45-51.

[12] R. Zimmermann, and W. Fichtner, "Low-power logic styles: CMOS versus pass-transistor logic," *IEEE Journal of Solid-State Circuits,* vol. 32(7), July 1997, pp. 1079-1090.

[13] P.C. McGeer, J.V. Sanghavi, R.K. Brayton, and A.L. Sangiovanni-Vincentelli, "ESPRESSO-SIGNATURE: a new exact minimizer for logic functions," *IEEE Trans. on VLSI Systems*, vol. 1(4), December 1993, pp. 432-440.

[14] G. De Micheli, *Synthesis and Optimization of Digital Circuits*, Mc-Graw Hill, 1994.

[15] C.-S. Ding, C.Y. Tsui, and M. Pedram, "Gate-level power estimation using tagged probabilistic simulation*," IEEE Transactions on CAD of Integrated Circuits and Systems*, vol. 17(11), November 1998, pp. 1099-1107.

[16] N. Hanchate, and N. Ranganathan, "LECTOR: a technique for leakage reduction in CMOS circuits," *IEEE Trans. on VLSI Systems*, vol. 12(2), February 2004, pp. 196-205.

Padmanabhan Balasubramanian received his B.E degree in Electronics and Communication Engineering discipline from the University of Madras, TN, India in 1998 and his M.Tech in VLSI System from National Institute of Technology, Tiruchirappalli, TN, India in 2005. He was earlier Lecturer in the School of Electrical Sciences at Vellore Institute of Technology (University and IET, UK Accredited), Vellore, TN, India. He is working towards his PhD in the School of Computer Science at The University of Manchester, Lancashire, UK. His research interests are in logic synthesis for low power, asynchronous design; CMOS based design and timing optimization issues.

C. Hari Narayanan completed his B.Tech in Computer Science and Engineering from School of Engineering, Cochin University of Science and Technology, Cochin, Kerala, India in 2004. He is currently pursuing his final year M.Tech in VLSI Design from Vellore Institute of Technology (University and IET, UK Accredited), Vellore, TN, India. His interests are in digital IC design and VLSI CAD tool development.

Karthik Anantha completed his B.Tech in Electronics and Communication Engineering from Jawaharlal Nehru Technological University, Hyderabad, AP, India in 2005. He is currently pursuing his final year M.Tech in VLSI Design from Vellore Institute of Technology (University and IET, UK Accredited), Vellore, TN, India. His current areas of interests include logic synthesis, reconfigurable architecture, software programming and computer graphics.