A Heuristic to Determine Low Leakage Sleep State Vectors for CMOS Combinational Circuits

Rahul M. Rao[1], Frank Liu[2], Jeffrey L. Burns[2], Richard B. Brown[1]

[1]Department of EECS
University of Michigan, Ann Arbor
MI, USA 48109
{rmrao,brown}@eecs.umich.edu

[2]Austin Research Labs
IBM, Austin
TX, USA 78758
{frankliu,jlburns}@us.ibm.com

ABSTRACT

Input vector control has been used to minimize the leakage power consumption of a circuit in sleep state [1]. In this paper, we present a novel heuristic for determining a low leakage vector to be applied to a circuit in sleep state. The heuristic is a greedy search based on the controllability of nodes in the circuit and uses the functional dependencies among cells in the circuit to guide the search. Results on a set of ISCAS and MCNC benchmark circuits show that in all cases our heuristic returns a vector having a leakage within 5% of that of the vector obtained using an extensive random search, with orders of magnitude improvement in computational speed.

1. INTRODUCTION

The increasing complexity and density of integrated circuits has resulted in an alarming increase in their power consumption. Power dissipation of integrated circuits has emerged as one of the most critical design metric since it dictates several system issues such as packaging and cooling costs, battery life for portable systems, etc. Aggressive scaling of channel length, oxide thickness and threshold voltage have resulted in an exponential increase in the leakage power of integrated circuits in recent technology generations [2]. Leakage power is thus becoming an increasingly important fraction of the total power consumption of integrated circuits across the entire design spectrum and is especially important for devices that spend a significant percentage of time in sleep state.

Several circuit techniques have been proposed to minimize leakage power dissipation. One possible approach is to use higher threshold voltage devices along non-critical paths [3]. Multi-threshold voltage CMOS (MTCMOS) in which high threshold-voltage sleep transistors are inserted in series with low-threshold voltage circuitry has been used in conjunction with sleep signals to reduce the leakage power in sleep mode [4]. Techniques for run-time modification of device threshold voltage using body bias have also been suggested [5-6]. All of these techniques require additional processing. In [7], the authors evaluated several run-time techniques for leakage power reduction and found input-vector control to be a very effective approach for leakage reduction without significant performance overhead. Input-vector control is based on the fact that the leakage of a circuit is dependent on the input vector applied to the circuit [1]. This technique is applicable irrespective of the additional process capabilities required for the above mentioned approaches but requires the knowledge of a low leakage vector that can be applied to the circuit during sleep mode to obtain savings in leakage power.

It can be shown that determining the minimum leakage input vector is an NP-hard problem. Many approaches for determining the minimum leakage vector have been suggested. In [1], the authors presented a formulation for determining the minimum leakage vector using a random search method with a specified statistical confidence and tolerance. A genetic algorithm was used to determine a low leakage vector in [8], while a greedy heuristic based on leakage observability metrics was used in [9]. In [10], the authors presented an incremental SAT solver that improves on the previously used random search method. None of these techniques explicitly uses the underlying circuit topology and dependency information. Also, some of these require multiple SPICE simulation runs for determining the leakage of the circuit, which can be time consuming, especially for large circuits.

In this paper, we propose a novel heuristic for determining low leakage sleep state vectors for a given circuit. This novel approach is based on the functional dependencies in the circuit and the controllability of its nodes. In Section 2, we define a few basic terms. Section 3 presents the heuristic with an example for a simple circuit. The results for a set of ISCAS and MCNC benchmark circuits are presented in Section 4 and the contributions are summarized in Section 5.

2. TERMINOLOGY

We assume that any circuit can be partitioned into smaller components in the form of gates, channel-connected components or other primitives (which we shall refer to as cells in the remainder of this document). We further assume that these cells have been pre-characterized for their leakage for all possible input combinations. This is similar to the process of delay characterization. It is also assumed that the node variables at the cell boundaries attain full logic values (V_{DD} or 0). The total leakage power of the circuit can then be determined as the sum of the leakage power of the individual cells.

2.1 Node Controllability and Controllability Lists

The controllability of each node in the circuit is defined as the minimum number of inputs that have to be assigned to particular states in order to force the node to a specific state. This is based on the concepts used in automatic test pattern generation for fault detection. Thus, two values are assigned to each node, namely CC0 (controllability-at-0) and CC1 (controllability-at-1). The controllabilities of the primary inputs are assumed to be 1. The controllabilities of each of the internal nodes in the circuit can be computed using the functionality of the cell that drives the node and the

Net	CC0	CC1
N_5	2: (1 x 1 x x)	1: (0 x x x x)
N_6	2: (x x 1 1 x)	1: (x x x 0 x)
N_7	2: (x 1 x 0 x)	1: (x 0 x x x)
N_8	2: (x x x 0 1)	1: (x x x x 0)
Z_0	2: (0 0 x x x)	2: (1 x 1 x x)
Z_1	2: (x 0 x x 0)	2: (x x x 0 1)

Table 1: Controllability lists for all nodes in c17.

controllabilities of its input nodes [11]. Controllability lists can be generated for each node, which specify the constraints on the input vector necessary to force the node to a specific state. Consider a simple benchmark ISCAS circuit, c17, shown in Fig. 1. Table 1 contains the controllability and the controllability lists for each node in the circuit. For instance, consider node N_5. The inputs to cell C_0 are primary inputs and hence have a controllability of 1. For the node N_5 to be at logic 0 state, it is necessary to have both the inputs to cell C_0 at logic 1 state. Hence, CC0 (N_5) = 2 and CC0_list (N5) is $<P_0P_1P_2P_3P_4>$ = *<1x1xx>*, where *x* represents a do not care condition. Similarly, for the node N_5 to be at logic 1 state, one of the inputs to cell C_0 should be at logic 0 state and thus CC1(N_5) = 1. In order to generate its controllability list, we determine the fan-out of each of the input nodes and select the node with a lower fan-out, which in this case is the primary input P_0. Thus, the controllability-at-1 list for node N_5 can be determined to be CC1_list (N_5) = *<0xxxx>*. The controllabilities and the controllability lists for all the nodes can be generated in a similar manner.

2.2 Best Input Condition (BIC) and Worst Input Condition (WIC)

The leakage of each cell in the circuit depends on the input pattern applied to that cell. Table 2 lists the total leakage for all possible input combinations for a Nand2 cell. It is seen that L_{00} and L_{10} are much smaller than L_{01} and L_{11}, where L*ij* refers to the total leakage (i.e., sub-threshold plus gate leakage) in the *input 1 = i* and *input 2 = j* state. Hence the leakage of a Nand2 cell can be minimized by forcing its inputs to the *x0* state. Similarly, constraints can be determined that minimize the leakage for each type of cell used in the design.

These constraints can be mapped onto the primary inputs using the controllability lists. Thus, the best input condition for each

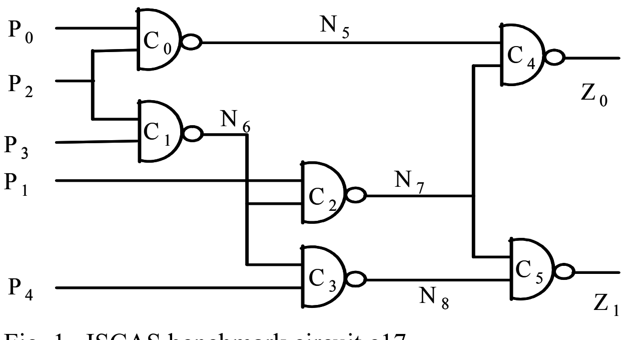

Fig. 1. ISCAS benchmark circuit c17.

cell represents the minimum number of primary inputs (and their specific values) that forces the cell to its low leakage state. For instance, the constraint for cell C_5 can be written as $<N_7N_8>$ = *x0*. Hence, its best input condition can be determined as follows:

BIC (C_5) = Constraint (N_7 = x) & CC0_list (N_8)

BIC (C_5) = *<xxxxx>* & *<xxx01>*

BIC (C_5) = = *<xxx01>*

Similarly, a list for the worst leakage state of each cell can be generated. For a Nand2 cell, the worst leakage state is the *11* state. Thus, the worst state condition for cell C_5 can be written as $<N_7N_8>$ = *11* which corresponds to a worst input condition WIC (C_5) = *<x0x0x>*. Table 3 lists the best and worst input condition for each of the cells in c17.

2.3 Cell Leakage Penalty (CLP) and Worst Leakage Penalty (WLP)

If the BIC for a cell cannot be satisfied, the cell may be in a higher (undesirable) leakage state. The increase in leakage of the cell can be quantified by a metric called Cell Leakage Penalty (CLP) given by the difference in the mean leakage of the undesirable states and the mean leakage of the desirable (low leakage) states. Thus, for a Nand2 cell the cell leakage penalty is given by

$$CLP(Nand2) = 0.5*(L_{01} + L_{11} - L_{00} - L_{10})$$

Similarly, if the worst input condition for the cell is satisfied, the cell is in its worst leakage state and this increase in leakage can be quantified using another metric called worst leakage penalty. The Worst Leakage Penalty (WLP) for a cell is given by the difference between the worst case leakage of the cell and the mean leakage of the desirable (low leakage) states.

$$WLP(Nand2) = L_{11} - 0.5* (L_{00} - L_{10})$$

2.4 Conflicting and Dominated Cells

The functionality of the cells in the circuit determines the states of the internal nodes for any given input vector. As a result, satisfying the best input condition for any cell will result in certain nodes in the circuit being forced to particular states. Hence, forcing C_A cell into its lowest leakage state may result in a violation of the input constraint of cell C_B. In such a situation, cell C_B is said be a conflicting cell for cell C_A. Similarly, it may be possible that forcing cell C_p into its lowest leakage state results in cell C_Q being forced into its lowest leakage state. In this case, cell C_Q is said to be dominated by cell C_p. This is similar to the definition of dominant faults used by the testing community [11]. For instance, for the circuit c17, cells C_0 and C_2 are conflicting cells (since they have opposing requirements for primary input P_2), while cell C_1 is

Input 1	Input 2	L_{ij} (mA)
0	0	6.10 E-05
0	1	1.53 E-04
1	0	5.61 E-05
1	1	5.54 E-04

Table 2: Total leakage current for all possible input patterns for a Nand2 cell.

Cell	Best Input Condition	Worst Input Condition
C_0	$(P_2 = 0)$ => x x 0 x x	$(P_0 = 1)$ & $(P_2 = 1)$ => 1 x 1 x x
C_1	$(P_3 = 0)$ => x x x 0 x	$(P_2 = 1)$ & $(P_3 = 1)$ => x x 1 1 x
C_2	$(N_6 = 0)$ => x x 1 1 x	$(P_1 = 1)$ & $(N_6 = 1)$ => x 1 x 0 x
C_3	$(P_4 = 0)$ => x x x x 0	$(N_6 = 1)$ & $(P_4 = 1)$ => x x x 0 1
C_4	$(N_7 = 0)$ => x 1 x 0 x	$(N_5 = 1)$ & $(N_7 = 1)$ => 0 0 x x x
C_5	$(N_8 = 0)$ => x x x 0 1	$(N_7 = 1)$ & $(N_8 = 1)$ => x 0 x x 0

Table 3: Constraint list for all cells in circuit c17.

dominated by cell C_4 (since input constraint for cell C_1 is a subset of the input constraint for cell C_4). Thus, for each cell a list of conflicting and dominated cells can be generated as in Table 4.

2.5 Cost Function

When the BIC of a cell is satisfied, the BIC of its conflicting cells are violated, while the BIC of its dominated cells are satisfied. Hence the cost of satisfying the best input condition of a cell(i) can be calculated using its list of conflicting and dominated cells and their cell leakage penalties as:

Cost (C_i) = Σ (CLP (conflicting cells(C_i))) - Σ (CLP (dominated cells(C_i))) - CLP (C_i).

3. HEURISTIC FOR DETERMING LOW LEAKAGE VECTOR

The total leakage of the circuit is the sum of the leakage of its constituent cells. Thus, the leakage of the whole circuit can be minimized by minimizing the leakage of the individual cells, i.e., by forcing the individual cells to their low leakage states. Theoretically, the leakage of a circuit will be minimum if all the cells are in their minimum leakage state. However, due to the functional dependencies between the cells in the circuit, it may not be possible to force all the cells into their low leakage states. In our approach, we attempt to satisfy the best input condition for cells in the circuit while minimizing the leakage penalty that results due to the violation of the best input condition for other cells. Our heuristic to determine the low leakage vector is as follows:

Cell	Dominated Cells	Conflicting Cells
C_0	-	C_2
C_1	-	C_2
C_2	-	C_0, C_2, C_4, C_5
C_3	-	C_1
C_4	C_1	C_2
C_5	C_1	C_2,C_3

Table 4: Conflicting and dominated cell list for all cells in circuit c17.

Iteration	Selected Cell	Input Vector	Selection List
0	-	x x x x x	C_0, C_1, C_2, C_3, C_4, C_5
1	C_4	x 1 x 0 x	C_0, C_3, C_5
2	C_0	x 1 0 0 x	C_3, C_5
3	C_3	x 1 0 0 0	-

Table 5: Selected cell and input vector after each iteration of Step 4 for c17.

```
1. For each node in the circuit:
    • Determine the controllability and controllability lists
2. For each cell in the circuit
    • Generate the input constraint lists
    • Determine the list of conflicting and dominated cells
3. Put all cells in the selection list
4. While selection list is not empty
    • Compute Cost function for each cell in the selection list
    • Select the cell with the least cost function
    • Satisfy its input constraint
    • Remove the selected cell and its list of dominated cells
      from the selection list
    • Remove the list of its conflicting cells from the selection
      list and put them in the violated list
    • Update the conflicting and dominating cell list for each
      cell
5. If any input is undefined
    • Set undefined input to 1 and determine Cost_Input(1) = $\Sigma$
      (WLP (Violated cells that get their WIC satisfied)
    • Set undefined input to 0 and determine Cost_Input(0) = $\Sigma$
      (WLP (Violated cells that get their WIC satisfied)
    • Assign input to suitable value based on the Cost_Input (0)
      and Cost_Input (1)
6. Done
```

The criteria of selecting the cell with the lowest cost function ensures that in each iteration the most profitable input constraint (minimum leakage penalty) is satisfied first. The undefined inputs can be assigned a suitable state in an attempt to minimize the number of violated cells that have their WIC being satisfied. The time complexity of the heuristic is $O(n^2)$, where n is the number of cells in the circuit, determined by Steps 2 and 4 of the heuristic. The final vector is a low leakage vector that can be applied to the circuit in sleep state. For instance, to determine the low leakage vector for circuit c17, the controllability lists, input constraint lists and the list of dominating and conflicting cells can be generated (as shown in Tables 1, 3 and 4 respectively). The initial selection list contains all the cells in the circuit and initial vector is completely unspecified. In the first iteration of the selection loop, cell C_4 is selected as the best cell since it has one dominated cell (C_1) and one conflicting cell (C_2) (and all the cells are identical and hence have the same CLP). Hence, the input vector is updated to satisfy the BIC for cell C_4 and the selection list is pruned by deleting the cells C_4, C_1 and C_2. Step 4 is then repeated till the selection list is empty as shown in Table 5.

4. RESULTS

We have implemented our heuristic in C and tested it using a set of ISCAS and MCNC benchmark circuits and the results are listed in Table 6. The heuristic results are compared against the best vector obtained from a random search across 10000 input vectors using SPICE. A choice of 10000 vectors gives us over 99% confidence that less than 0.5% of the vector population has a leakage lower than the minimum leakage value observed from the random search [1]. For circuits with less than 13 inputs, we compared our results against an exhaustive search over the input vector space. The runtime savings compares the heuristic with just a single SPICE leakage estimation for the circuit.

The best vector obtained by our heuristic has a leakage of within 5% as compared to the best vector obtained from random search in all cases. In some cases, the heuristic returns a lower leakage vector compared to that obtained from a random search. For instance, the best vector obtained for the MCNC circuit i4 has a 10% lower leakage than the random search vector. The heuristic can be easily altered to determine a high leakage vector by suitably changing the input constraints for the constituent cells. Thus, our heuristic can be used to estimate the approximate bounds on the leakage power consumed by the circuit.

	Random	Heuristic		
Ckt	Minimum Leakage (mA)	Minimum Leakage (mA)	Diff %	Runtime Savings (X)
c17	0.000927	0.000927	0.00	2420.00
c432	0.053687	0.054487	-1.49	247.67
c499	0.148894	0.14869	0.14	1098.22
c880	0.085622	0.08504	0.68	134.52
c1355	0.119469	0.118096	1.15	161.04
c1908	0.182616	0.178759	2.11	63.15
c2670	0.290749	0.290168	0.20	22.55
c5315	0.597011	0.608196	-1.87	24.48
c6288	0.589224	0.571751	2.97	458.81
c7552	0.836394	0.830377	0.72	34.03
i4	0.043078	0.038726	10.10	52.36
i5	0.073675	0.068402	7.16	69.83
i6	0.109648	0.114423	-4.35	320.41
i7	0.148743	0.149879	-0.76	129.22
i8	0.580241	0.597182	-2.92	48.84
i10	0.718149	0.721337	-0.44	35.24
my_add	0.064005	0.064766	-1.19	1318.00
c8	0.066114	0.067407	-1.96	378.40
cc	0.018429	0.018809	-2.06	2182.00
cht	0.053588	0.053577	0.02	724.22
clip	0.129456	0.134287	-3.73	1012.00
cm138a	0.004540	0.004563	-0.51	5100.00

Table 6: Results for a set of ISCAS and MCNC benchmark circuits compared with random search across 10000 vectors.

5. CONCLUSIONS

The leakage power consumed by a circuit in sleep state can be reduced by applying a low leakage vector. In this paper, we have presented a novel heuristic to determine such a low leakage vector. The heuristic uses the concept of controllability of nodes in the circuit and utilizes the functional dependencies of the cells in the circuit to guide the input vector search. The results show that in most cases the heuristic can determine a vector that results in a leakage very close to that obtained using an extensive random search, at much lower computational cost.

6. REFERENCES

[1] J. Halter and F. Najm, "A Gate-Level Leakage Power Reduction Method for Ultra Low Power CMOS Circuits," *Proc. of CICC,* pp.475-478, 1997.

[2] S. Borkar, "Design Challenges of Technology Scaling," *IEEE Micro,* Vol. 19, no. 4, pp. 23-29, Aug. 1999.

[3] L. Wei, et.al, "Design and optimization of low voltage high performance dual threshold CMOS circuits," *Proc. of DAC,* pp. 489-494, Jun. 1998

[4] S. Mutoh, et.al, "1-V Power Supply High-Speed Digital Circuit Technology with Multi-Threshold Voltage CMOS", *IJSSC,* vol. 30, no. 8, pp. 847-854, Aug. 1995.

[5] T. Kurado, et.al, "A 0.9V, 150MHz, 10-mW, 4mm2, 2-D Discrete Cosine Transform Core Processor with Variable Threshold (Vt) Scheme," *IEEE Journal of Solid State Circuits,* vol. 31, no. 11, pp. 1770 - 1779, Nov. 1996.

[6] F. Assaderaghi, "DTMOS: Its Derivatives and Variations, and their Potential Applications," *Proc. of 12th Intnl. Conference on Microelectronics,* pp. 9-19, 2000.

[7] D. Duarte, Y. Tsai, N. Vijaykrishnan and M. Irwin, "Evaluating Run-Time Techniques for Leakage Power Reduction," *Proc. of 15th Intnl. Conference on VLSI Design,* pp. 31-38, 2002.

[8] Z. Chen, M. Johnson, L. Wei and K. Roy, "Estimation of Standby Leakage Power in CMOS Circuits Considering Accurate Modeling of Transistor Stacks," *Proc. of ISLPED,* pp. 239-244, 1998.

[9] M. Johnson, D. Somasekhar and K. Roy, "Models and Algorithms for Bounds on Leakage in CMOS Circuits," *IEEE Trans. on CAD,* vol. 18, no. 6, pp. 714-725, Jun. 1999.

[10] F. Aloul, S. Hassoun, K. Sakallah, D. Blaauw, "Robust SAT-Based Search Algorithm for Leakage Power Reduction," PATMOS, 2002.

[11] M. Abramovici, M. Breuer, A. Friedman, "Digital Systems Testing and Testable Design," *IEEE Press,* 1995.