Monte Carlo Algorithm for Leakage Optimization Based on Input Vector Control

Yuan Cao

Input Vector Control (IVC) is a simple yet effective way of optimizing the leakage power of a given digital circuit [1, 2]. Compared with other methods such as reducing power supply voltage or MTCMOS technology, IVC can optimize the leakage power to a fraction, without adding extra circuitry or introducing complexity. For an arbitrary combinational logic circuit, finding the input vector that consumes the least leakage power has been proved to be an NP-complete problem, which means any correct algorithm for this problem has a complexity that is exponential with the input vector size.

Several approaches have been proposed to solve this problem, either accurately or approximately. These algorithms include those based on 2-SAT solver, dynamic programming, linear programming, etc. Another class of algorithms uses random search to find the best input vector. Exact solutions such as those based on 2-SAT turned out to be extremely slow on large circuits, while the faster algorithms usually give poorer results.

In this work, an improved random algorithm based on simulated annealing for finding optimal input vector is proposed. Simulated annealing is an effective approach for minimizing the energy of a system with extremely large degrees of freedom, and it can avoid falling into local minimums which is usually encountered by simpler hill climbing algorithms. In our problem, the 'energy' of the system is the leakage current of any combinational digital logic, and we want to minimize the 'energy' with respect to the input vector.

I. Model for simulation

To simulate any gate-level combinational circuit, a hierarchical model is adopted to describe the logic connections. Figure 1 shows the hierarchy. A circuit has some input nodes, output nodes and components, which are connected together by internal nodes. A component can be another circuit, or a low-level gate, that is the leaf node of the hierarchy. A gate computes some outputs and a leakage current based on provided inputs. The inner details of the gate can be viewed as a black box which depends on the implementation. Usually a gate has only a few inputs, and the outputs and leakage current is pre-computed and stored as a look-up table. Figure 2 gives an example of the look-up table for a 28-T CMOS full adder gate, which has three inputs and two outputs. Figure 4 shows the dependence of the leakage current on the input vector of a 4-bit ripple-carry adder, and the histogram of the leakage current. It is seen that IVC can reduce the leakage up to 35% for this circuit.

With the information from every single gate, the output as well as the leakage current of the entire circuit can be assembled hierarchically, for any given input vector to the top-level circuit. However, the top-level circuit can have tens or even hundreds of inputs, which makes enumerating all input vectors impossible for these circuits. The algorithm to be proposed overcomes this problem by doing a Monte Carlo annealing process, which vastly reduces computation time but still maintaining a good result.

II. Proposed Algorithm

In our proposed annealing approach, the optimal input vector is obtained by optimizing a random initial input vector in the following way. A global parameter *T* is defined as the 'temperature' of the system. In each iteration,

1. One of the inputs in the input vector is randomly chosen and flipped (i.e. 0 to 1, 1 to 0).
2. The new leakage current is evaluated and compared with the previous leakage current.
   a. If the new current is smaller, the change will be kept and a new iteration starts from step 1.
   b. If the new current is larger, the change is retained only at a probability of $\exp(-\Delta I / T)$. Otherwise, the change is discarded and we return to step 1.
3. After a certain number of iterations, *T* is decreased by a fraction. The iteration continues with the new *T*.
4. The leakage current will finally converge to the minimum.

III. Heuristics

The simulated annealing algorithm described above might give poor results randomly in some cases, due to the following observation: since we only flip one bit in the input vector at a time, if there is a lower leakage input vector that is only two bits different from the current one, the algorithm might still stuck in the local minimum until the end. Therefore to prevent this, we do 'relaxation' on the present best state after some number of iterations, which is to enumerate all possible two-bit flipping and see whether the leakage can be lowered. Since this is a O(n^2) operation where *n* is the input size, we do not want to do the relaxation too often that impacts the performance. It is observed that by doing this relaxation process, the probability of finding the global optimal leakage can increase by up to an order of magnitude.

IV. Results and Discussion

In order to test the above described algorithm based on simulated annealing, the algorithm is implemented by Java and tested on several widely used benchmark circuits from ISCAS85. These benchmark circuits range from tens of gates to several thousands of gates, therefore providing a comprehensive evaluation on the effectiveness of our algorithm. The leakage of the benchmark circuits are computed using parameters extracted from Cadence GPDK 90nm technology library. Computations are performed on Intel Core i7-4790@3.6GHz.

Figure 5 shows the typical iteration process on a combinational circuit. As *T* cools down, the system is harder and harder to flip to a state that has larger leakage current. Eventually, the system should converge to a minimum when *T* goes to zero. If the cooling process is slow enough, it should always end up in the global minimum. Even if the cooling is not slow, the result obtained is usually good enough.

Figure 3 lists the results of using the described algorithm with heuristics on benchmark combinational circuits. It is seen that our algorithm has an O(n^2) complexity due to the heuristics. The average result of our algorithm is very close to the true minimum leakage regardless of the size of the circuit, while the calculation time is moderate compared to other existing algorithms.

V. Conclusion

An algorithm for optimizing input vector to reduce leakage power consumption based on simulated annealing is proposed, and its validity and effectiveness is tested on various benchmark combinational circuits of different sizes.

Acknowledgements:

The authors gratefully acknowledge Prof. Vivienne Sze and Yu-Hsin Chen for helpful discussion.

References:

[1] Halter, J. P. and Najm F. N., "A Gate-Level Leakage Power Reduction Method for Ultra-Low-Power CMOS Circuits"

[2] Feng G., and Hayes J. P., "Exact and heuristic approaches to input vector control for leakage power reduction." Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 25.11 (2006): 2564-2571.

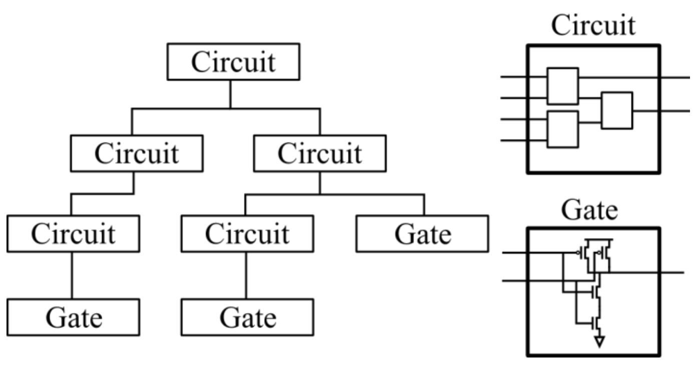

Figure 1: Hierarchical simulation model

Input vector(A,B,Ci)	Output(S,Co)	Leakage/nA
000	00	76.97
001	01	104.60
010	01	126.48
011	10	124.12
100	01	146.35
101	10	126.51
110	10	146.13
111	11	105.73

Figure 2: Leakage table of a full adder

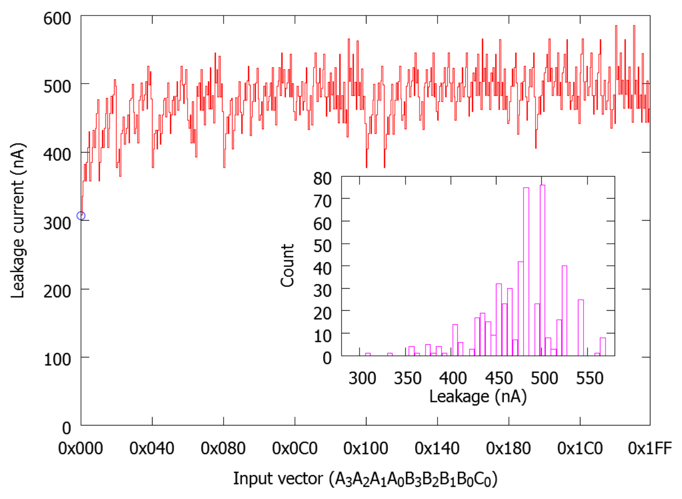

Figure 4: Leakage of a 4-bit ripple carry adder

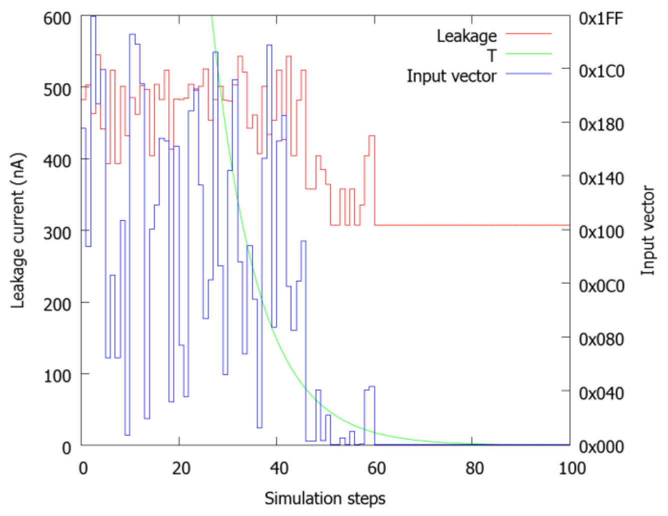

Figure 5: Typical annealing process of the 4-bit adder

Circuit	Inputs	Gate count	Min. Leakage(nA)	Time(s) (By bruteforcing)	Avg. annealing Leakage(nA)	Time(s) (By annealing)	Ratio
Adder-4	9	4	307.9	0.1	307.9	0.1	1.00
C17	5	6	35.4	0.1	35.4	0.1	1.00
C432	36	160	3360.6	-	3389.1	1.0	1.008
C880	60	383	9716.7	-	9815.9	6.0	1.010
C1355	41	546	11081.9	-	11081.9	5.0	1.00
C3540	50	1669	52588.5	-	52588.5	25.0	1.00
C6288	32	2416	75939.3	-	75939.3	22.0	1.00
C7552	207	3512	1.05×10^5	-	1.07×10^5	439.0	1.019

Figure 3: Leakage calculation for various benchmark circuits