Chip Level Power Supply Partitioning for I_{DDQ} Testing Using Built-In Current Sensors

Abhijit Prasad, D. M. H Walker

Dept. of Computer Science
Texas A&M University
College Station TX 77843-3112
Tel: (979) 862-4387
Fax: (979) 847-8578
Email: {abhijitp, walker}@cs.tamu.edu

Abstract

The International Technology Roadmap for Semiconductors projects that I_{DDQ} levels will rise rapidly with each technology node. In addition, manufacturing variations in the I_{DDQ} level will be difficult to control. The combination will make it increasingly difficult to distinguish defect-free from defective chips via I_{DDQ} tests. Built-in current sensors (BICSs) have been proposed to increase test resolution by virtually partitioning the supply mesh, so that each partition has a relatively small defect-free I_{DDQ} level. In the future such a scheme would require 100 000 or more BICSs and thus the partitioning task needs to be automated. This paper presents a practical methodology to do this power supply partitioning.

1. Introduction

In the Very Deep Submicron (VDSM) era, the MOSFET leakage current (i.e. defect-free I_{DDQ}) is rising rapidly with each technology generation. An additional difficulty in these technologies is the significant variation in effective channel length, transistor threshold voltage and the gate oxide thickness. Combined with a higher background current, this makes it increasingly difficult to distinguish faulty I_{DDQ} from process variation [1][2][3][4]. I_{DDQ} testing has been a very effective test method, offering high fault coverage with a small set of vectors [5]. Besides this, it plays a vital role in defect characterization and failure analysis. Hence, it would be desirable to extend its usefulness into the VDSM technology era.

Several approaches have been proposed to extend the life of I_{DDQ} testing. Delta-I_{DDQ} [6], Current Ratio [7], and Current Signature [8][9] methods can increase I_{DDQ} test resolution by 30-100 times, however these only extend the usefulness of I_{DDQ} testing for a few more technology nodes. Recent work [1][10][11][12][13] also examines ways to lower the intrinsic leakage current: temperature reduction, substrate backbiasing, lowered quiescent V_{DD}, multiple transistor thresholds, stacked transistors, and Silicon on Insulator (SOI). These approaches suffer from drawbacks like the use of a specific technology, design modifications, or process modification. They may also be insufficient to keep the background leakage low enough to permit effective I_{DDQ} testing.

Power supply partitioning has been proposed to increase test resolution by partitioning the power supply network, such that each partition has a relatively small defect-free I_{DDQ} level. Analysis in [14] shows that the only feasible long-term I_{DDQ} test approach would be to combine power supply partitioning with resolution enhancement methods.

Projections from the International Technology Roadmap for Semiconductors [15] show that I_{DDQ} for high performance microprocessors will rise to over 100 A in the 35 nm technology node. The total chip I_{DDQ} values are computed by assuming half the transistors are leaking, and the W/L of the transistors is 3. These are reasonable if one assumes that the leakage is dominated by SRAM arrays.

In order to be feasible as a test method, the total BICS area must be kept to 1% of the chip area. The analysis in [14] showed that for good I_{DDQ} resolution the sensor area is infeasibly small for the high performance technologies, particularly when one considers the need to access large numbers of sensors via scan chains. Therefore, BICS must be combined with a variance reduction method to reduce the required number of sensors by about 100 times, requiring about 100 000 sensors in the 35 nm technology node.

The requirements for a practical BICS were described in [14] and a prototype sensor was described in [16]. This sensor is about 500 transistors in size, including self-calibration and scan chain readout, which has 400 transistors. Since most transistors are in the scan chains, if they are shared across multiple sensors the area overhead can be reduced at the expense of test time. For example, sharing a scan chain with 4 sensors reduces the per-sensor transistor count to 200 from 500 while increasing test time four times.

Partitioning at the logic level is easiest to implement. It has the inherent drawback that the place/route tools would need to be modified to be able to handle chunks of logic in a way that all the power to a given partition be monitored by a sensor on a single branch. In such a case, the number of partitions would be the number of sensors required. Such a routing may not meet all IR-drop constraints of the power network and thus in general the only way to do power supply partitioning is once cells have been placed and routed. Power gating [17] to reduce power dissipation provides a natural partition to insert a BICS, but such partitions may be too large or too small for BICS insertion and hence is not considered here. There has been other prior work done to estimate the partition sizes [18][19][20], but there is no prior work on a practical partitioning strategy.

An alternative scheme that identifies a set of branches on the power supply network that have current less than the acceptable background leakage needs to be developed. This minimal set of branches must also have the property that monitoring current on these branches enables us to monitor all the leakage current on the chip. A BICS can then be used to monitor current on each of these branches. In this paper we present a methodology that can generate such a set optimally in polynomial time. We assume that BICSs must be inserted without modifying the supply network topology. We also do not consider the problem of fitting the sensors into the network. This is part of our future work.

The rest of the paper is organized as follows. In Section 2 we formulate the power supply partitioning problem. We explain how the power supply network is transformed into a flow network in Section 3. Section 4 explains the experiments we conducted, and the results obtained. We give our conclusions and list future work in Section 5.

2. Power Supply Partitioning Problem

There can be multiple variants of the power supply partitioning problem depending on the constraints and the objective function. We first enumerate the variables in this problem:

- I_m - The maximum current being monitored by any sensor on a single branch.
- n - The number of sensors used to monitor the current.
- I_u - The total current that is not being monitored by the sensors.

I_m addresses the problem of reduction in test resolution as background leakage current increases. The higher this value, the lower the resolution of the sensor in detecting faults. n is the number of sensors used on the chip. It addresses the issue of area overhead due to the additional sensor circuitry on the chip, and increased routing complexity. I_u helps in deciding the test escape probability. The greater the current that is not monitored, the higher the probability of missing detection of a defect using I_{DDQ} test, and thus the lower the quality of the solution. There can be three formulations:

Minimize I_u	Minimize I_m	Minimize n
Such that $n <= n_{max}$;	Such that $n <= n_{max}$;	Such that $I_m <= I_{Dmax}$;
And $I_m <= I_{Dmax}$.	And $I_u = 0$.	And $I_u = 0$.

The problem formulation we have considered in this paper is to minimize n. It means that we need to cut the power network into two parts, with the power pads in one partition and all the cells in the other partition. All the edges on the cut must have a current less than I_{Dmax}. It is easy to see that such a solution meets both constraints as described above. If we call the number of edges along the boundary of the two partitions the cut size, our objective is to find the partition with the minimum cut size.

So far in our problem formulation, the constraint on the current limit I_{Dmax} has been fixed. However, this can be relaxed to allow a small increase in I_m if it results in a large reduction in the number of sensors n. For example, if I_{Dmax} is 10 µA and there is an 11 µA branch feeding 22 branches each carrying a current of 0.5 µA, then it might be preferable to have a single sensor monitoring the 11 µA line rather than 22 sensors monitoring each of the 0.5 µA lines. Such fine-tuning of the algorithm can be done easily in the transformed max-flow problem.

3. Power Supply as a Flow Network

We model the power network as a directed graph $G=(V, E)$, where V is the set of nodes (i.e. fan-out points in the resistive network), and E is the set of branches. The weight of a branch is the nonzero current flowing on the branch. If there is no current flow, or if it is in the reverse direction, the branch is assigned zero weight.

The first step is to obtain the topology of the power supply network. This can be easily obtained from the parasitic resistance extraction of the power network. We also need to know the leakage current estimates of the cells. The technology design parameters give us this data. The extracted parasitic resistances form a resistive network and the leakage current of the cells can be modeled as current sources hanging off the nodes in the network. This linear circuit problem can be solved to obtain the fault free leakage current magnitude and direction for each branch.

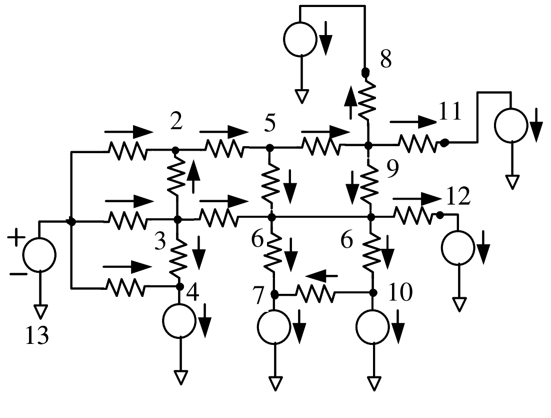

Figure 1. A resistive network

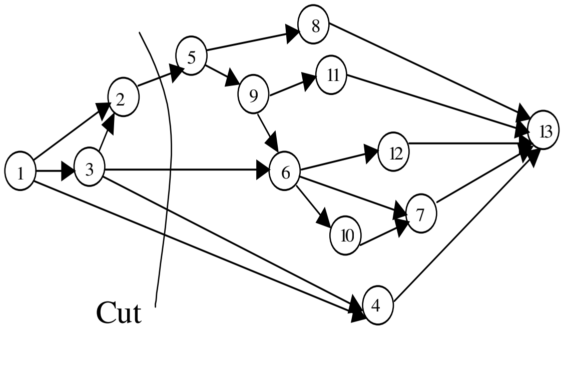

Figure 2. Abstracted as a flow network

We use the above data to transform G into a flow network $G' = (V, E')$ with each edge e in E associated with an edge e' in E'. If s and t are the source and target in V, then an s-t cut (or simply *cut*) (V_1, V_2) is a bipartition of V into sets V_1 and V_2 such that $s \in V_1$ and $t \in V_2$. An edge whose starting node is in V_1 and ending node is in V_2 is defined as a forward edge from V_1 into V_2. The capacity of the cut (V_1, V_2) is the sum of capacities on the forward edges from V_1 to V_2.

We then reduce the problem to that of finding an *s-t* (source to target) cut of minimum capacity with appropriate *s* and *t* values in the transformed flow network. Figure 2 shows the transformation of the resistive network in Figure 1 to its equivalent flow network. The numbers shown on the circuit are the nodes and the arrows show the direction of current flow. The values of *s* and *t* in this particular case are 1(V_{DD}) and 13(GND) respectively.

Each edge *e'* in *E'* has two values associated with it, a flow *F*(*e'*) and a capacity *C*(*e'*). The capacity *C*(*e'*) is assigned by comparing the weight on the corresponding edge *e* with I_{Dmax}. If the weight of *e* is less than or equal to I_{Dmax}, we set *C*(*e'*) to 1, otherwise it is set to infinity. The value of *F*(*e'*) is initially assigned 0 for all edges.

The reason behind the capacity labeling scheme is as follows. All branches that have current less than I_{Dmax} are potential sites for the sensor placement and are thus potential cut edges. Since it is the number of such cut edges that we want to minimize, we normalize all the edges such that all potential cut edges have the same probability of being chosen. Correspondingly, all edges that are not potential edges are given a high capacity that forces the min-cut algorithm to avoid choosing them. These values can either be set to a single infinitely high value or scaled up according to the edge weight. As an example, all edges with weight from I_{Dmax} to 110% of I_{Dmax} can be assigned weight 20. This would mean that exceeding I_{Dmax} by 10% would only be acceptable if it requires 19 fewer sensors than otherwise. This capacity scaling scheme is flexible, allowing us to set tradeoffs in the solution.

This scheme can also be extended to eliminate those branches on which it would be difficult to monitor current. For example, in a mesh type power network, it might be hard to monitor current on a branch that is in the higher metal layers and thus this branch could be given a higher weight.

The max-flow min-cut theorem [21] states that the value of a maximum flow is equal to the capacity of a minimum cut. More formally stated, given a max flow *f* in *G*, let $V_1 = \{v \in V\text{: } \exists$ an augmenting path from *s* to *v* in *G*$\}$, and let $V_2 = V - V_1$, then (*V1*, *V2*) is a cut of minimum capacity (which is equal to $|f|$), and *f* saturates all forward edges from V_1 to V_2. To find the min-cut once the flows are known, we start from *s* and traverse the network until we reach a saturated edge. The set of saturated edges found this way gives us the min-cut. The edges on this min-cut are the set of branches on the power network where sensors need to monitor current.

There are numerous polynomial time algorithms that exist to find the max-flow in a network. The fastest such algorithm is the Goldberg-Tarjan algorithm [22] that has a time complexity of $O(|V|^3)$, which we have implemented in this work.

4. Experiments and Results

We implemented the partitioning scheme described above using the C language and ran it on a Pentium 4, 1 GB RAM, 2.26 GHz PC running RedHat Linux 7.3. Since most large chip designs use a mesh type supply network, these are the type we consider (Figures 3 and 4). The shaded lines are the top-level power lines and the un-shaded lines are the ground lines. The horizontal and vertical lines are connected together using vias as shown. To show the effect of the power network on the number of sensors we consider two strategies for pad placement. The first strategy is shown in Figure 3, where the power pads are an array on the surface of the chip. The second strategy shown in Figure 4 has the power pads on the periphery.

In the case when the current at the power pads is less than the maximum acceptable background current, the solution to the problem is trivial, i.e. to measure the current at each power pad. However, it is expected that the current being drawn at each pad is larger than I_{Dmax} for any reasonable size circuit.

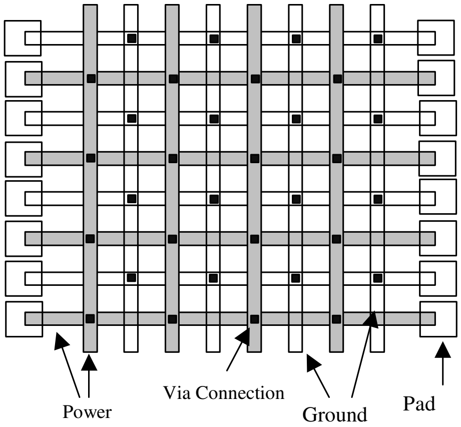

Figure 3. Mesh network with power pads as an array

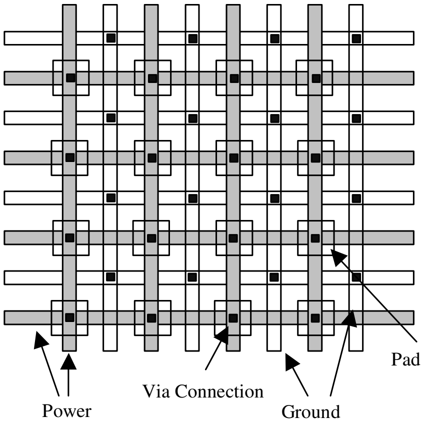

Figure 4. Mesh network with power pads at periphery

We generated a set of power networks and use these as our test data. The number of rows, columns and cells is varied to get different power supply topologies. At the lower level of the power network, each branch on the main mesh feeds a group of cells, which itself could be connected to other branches. The number of cells in each region is randomly distributed. Hence certain regions of the chip will be denser than other regions, so the current at some pads may exceed I_{Dmax} even if the average pad current is less than I_{Dmax}. This effect can be seen in Figure 5 and Figure 6 where the average current per pad has been shown as dotted lines (at 232.5 μA for pads as an array and 465 μA for pads at periphery). If cell distribution were uniform, the number of sensors would be equal to the number of pads at these I_{Dmax} values. However, due to non-uniform distribution of cells, and therefore branch currents, the number of sensors is greater than the number of power pads.

We assume 10 transistors per cell. The estimate of leakage current of the transistors is made from the ITRS predictions for the 130 nm technology (i.e. 2.55 nA). The average leakage per cell assumes 50% of the transistors are leaking or 12.75 nA per cell. It is assumed that the mesh structure of the power network is on Metal 5 and 6, and the lower level connections to the cells are done on Metal 4. Sheet resistance values are obtained for the respective metal layers of 130 nm technology. The power grids considered are presented in Table 1. Column 2 gives the number of cells in the chip. The next two columns are the number of rows and columns of the power mesh. Columns 5 and 6 are the number of power pads for the two different power pad configurations, and the last column is the total I_{DDQ} of the chip.

Table 1. Test Cases

Chip No.	No. of Cells	Rows on Mesh	Columns on Mesh	No. of Power Pads using Periphery	No. of Power Pads using Array	Chip I_{DDQ} (mA)
1	150 900	20	20	40	80	2
2	1 302 348	60	60	120	240	17
3	3 732 039	100	100	200	400	48
4	7 202 460	140	140	280	560	92
5	11 830 482	180	180	360	720	151
6	14 595 449	200	200	400	800	186

Table 2. Pads at periphery, I_{Dmax} = 100 µA

Chip No.	Minimum Number of Sensors	Number of Sensors given by our Algorithm			% Area Overhead without Scan Optimization	% Area Overhead with Scan Optimization
		On the mesh	Not on the mesh	Total		
1	40	40	0	40	1.6	0.6
2	170	815	177	992	3.9	1.5
3	480	3 513	1 274	4 787	6.4	2.5
4	920	7 596	3 436	11 032	7.6	3.0
5	1 510	13 299	6 503	19 802	8.4	3.4
6	1 860	16 528	8 558	25 086	8.6	3.5

Table 3. Pads as an array, I_{Dmax} = 100 µA

Chip No.	Minimum Number of Sensors	Number of Sensors given by our Algorithm			% Area Overhead without Scan Optimization	% Area Overhead with Scan Optimization
		On the mesh	Not on the mesh	Total		
1	80	110	3	113	3.7	1.5
2	240	303	13	316	1.2	0.5
3	480	1 826	160	1 986	2.7	1.0
4	920	3 980	618	4 598	3.2	1.3
5	1 510	7 267	1 573	8 840	3.7	1.5
6	1 860	8 867	2 246	11 113	3.8	1.5

Our algorithm was run with I_{Dmax} of 100 µA for both the power pad configurations. We chose this number since many chips with leakage of 100 µA are successfully screened with I_{DDQ} test today [23]. The results are presented in Tables 2 and 3. The columns for these tables are explained as follows. We calculate the lower bound on the number of sensors by dividing the total chip I_{DDQ} by I_{Dmax}, i.e. if each sensor monitors exactly I_{Dmax} then this will be the number of sensors required to monitor all the current on the chip. The actual minimum number of sensors required is obviously bounded by the number of power pads on the chip since in the best case we will require at least those many sensors. Hence the greater of the calculated lower bound and the number of power pads is the value in column 2. Columns 3, 4 and 5 are the number of sensors given by our algorithm and the breakdown of the sensor location. Columns 6 and 7 are the area overhead of the sensors. Consider the case of chip 6 from Table 3 - 11 113 sensors are required for the entire chip with 14 595 449 cells. We know that a cell has 10 transistors and the area for a sensor is equivalent to the area of about 500 transistors (100 in the sensor circuit and 400 in the scan chain readout). The area overhead for the sensors can be calculated as (500 * 11 113)/(14 595 449 * 10). This works out to a 3.8% overhead. These are the values in column 6. For the numbers presented in column 7, it is assumed that four sensors share a single scan chain, giving an average of 200 transistors per sensor. This method gives us a 2.5 times reduction in area overhead with a 4 times test time penalty. Hence, for the case considered above the area overhead with scan chain optimization is 1.5%.

In addition we also experiment with various values of I_{Dmax} for Chip 6. In Figure 5 we observe that as I_{Dmax} is increased, sensors locations are more on the mesh than on the lower levels until they are finally all on the mesh. This is expected, as the current at the mesh level is greater than at the lower levels and would thus give a better cut. At lower values of I_{Dmax}, the algorithm is forced to look for cuts at a lower level, which increases the number of cuts required.

In Figure 6 we show the change in the total number of sensors required for different values of I_{Dmax}. It is clear that the mesh type power supply network always requires fewer sensors than the peripheral configuration. We observe that for the periphery topology the percentage of sensors on the mesh is fewer than in the array topology. This is due to the structure of the periphery power network that causes non-uniform distribution of currents in the branches of the mesh. The middle regions of the chip carry small currents whereas those closer to the periphery carry larger currents. As a consequence many sensors will be underutilized measuring currents less than I_{Dmax}. By making the pads as an array, current flow in the branches of the power network becomes more uniform. As a result there are fewer branches carrying very low currents and sensors can be pushed to their limits, thus decreasing the total number of required sensors.

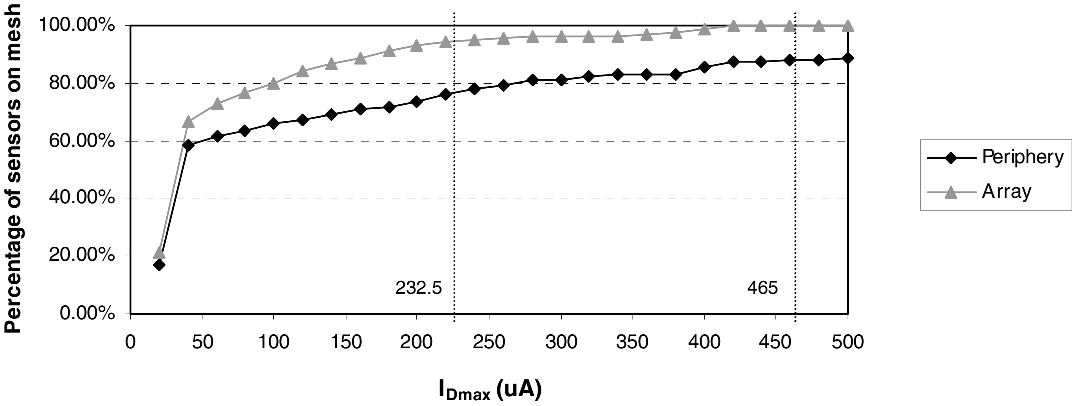

Figure 5. Percentage of sensors on the mesh

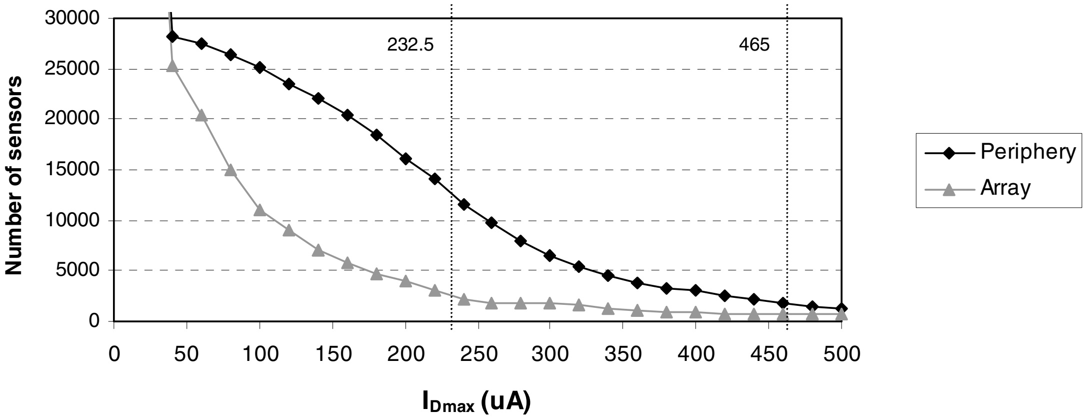

Figure 6. Total sensors for Array and Periphery

5. Conclusions and Future Work

We present a practical methodology to partition the power supply network of a chip for on-chip current sensing. Experimental results show that combined with resolution enhancement methods like Delta-IDDQ, this strategy generates solutions with acceptable overhead and is the first such methodology of its kind. Our future research is on the following areas:

- Work on the different forms of the power supply partitioning problem.
- Identify methods to modify the power network so that suitable cuts can be obtained.
- Consider the power and ground network together for potential sensor locations.
- The problem of inserting the sensors into the power supply network.

Acknowledgements

This research was funded by the Semiconductor Research Corporation under contract 2001-TJ-954.

References

[1] M. Sachdev, "Deep Sub-micron I_{DDQ} Testing: Issues and Solutions", *European Design and Test Conference,* Paris, France, 1997.

[2] T. Williams et al., "I_{DDQ} Test: Sensitivity Anlysis of Scaling", *International Test Conference*, Washington DC, 1996, pp.786-792.

[3] A. Ferre and J. Figueras, "I_{DDQ} Characterization in Submicron CMOS", *IEEE International Test Conference*, Washington DC, 1997, pp. 136 -145.

[4] P. Nigh et al., SEMATECH Study, "An Experimental Study Comparing the Relative Effectiveness of Functional Scan, I_{DDQ}, and Delay Fault Testing", *IEEE VLSI Test Symposium*, Washington DC, 1997, pp. 459.

[5] S.D. McEuen, "I_{DDQ} Benefits", *IEEE VLSI Test Symposium,* Atlantic City NJ, 1991, pp.34-39.

[6] C. Thibeault, "On the Comparison of Δ I_{DDQ} and I_{DDQ} Testing", *VLSI Test Symposium*, 1999, Dana Point CA, pp. 143-150.

[7] P. Maxwell, et al., "Current Ratios: A Self-scaling Technique for Production I_{DDQ} Testing", *IEEE International Test Conference,* Atlantic City NJ, 1999, pp. 738-746.

[8] A. Gattiker and W. Maly, "Current Signatures: Application", *IEEE International Test Conference*, Washington DC, October 1997, pp. 156-165.

[9] C. Thibeault, "A Novel Probabilistic Approach for IC Diagnosis Based on Differential Qiescent Current Signatures", *IEEE VLSI Test Symposium*, Monterey CA, 1997, pp. 80-85.

[10] M. Sachdev, "Deep Sub-micron I_{DDQ} Test Options", *IEEE International Test Conference*, Washington, DC, 1996, pp. 942.

[11] A. Keshvarzi, K. Roy and C.F. Hawkins, "Intrinsic Leakage in Low Power Deep Submicron CMOS ICs", *IEEE International Test Conference*, Washington DC, 1997, pp. 146-155.

[12] T. Karnik, et al., Total power optimization by simultaneous dual-Vt allocation and device sizing in high performance microprocessors", *Design Automation Conference*, New Orleans LA, 2002, pp 486-491.

[13] G. Sery, S. Borkar and V. De, "Life Is CMOS: Why Chase the Life After?", *Design Automation Conference*, New Orleans LA, 2002, pp. 78-83.

[14] D. M. H. Walker, "Requirements for Practical I_{DDQ} Testing of Deep Submicron Circuits", *IEEE International Workshop on Defect and Current Based Testing*, Montreal, Canada, April 2000, pp.15-20.

[15] International Technology Roadmap for Semiconductors 2001, Semiconductor Industry Association, available at http://public.itrs.net.

[16] H. Kim and D. M. H. Walker, "A Practical Built-in Current Sensor for I_{DDQ} Testing", *IEEE International Test Conference*, Baltimore MD, Oct 2001, pp. 405-414.

[17] F. Li and L. He, "Maximum Current Estimation Considering Power Gating", *International Symposium on Physical Design*, 2001, Sonoma CA, pp. 106-111.

[18] W. Maly and M. Patyra, "Built-In Current Testing", *IEEE Journal of Solid State Circuits*, Vol. 27, No.3, Mar 1992.

[19] S. M. Menon and M. Palmgren, "Estimation of Partition Size for I_{DDQ} Testing using Built-In Current Sensing" *IEEE International Workshop on I_{DDQ} Testing*, Washington DC, Nov 1997, pp. 68-72.

[20] Y. K. Malaiya, et al., "Enhancement of resolution in supply current based testing for large ICs", *VLSI Test Symposium*, Atlantic City NJ, 1991, pp. 291-296.

[21] J. R. Ford and D. R. Fulkerson, "Flows in Networks", *Princeton University Press*, Princeton, 1962.

[22] A. V. Goldberg and R. E. Tarjan, "A New Approach to the Maximum Flow Problem", *Proceedings of the 18th ACM Symposium on Theory of Computing*, 1986, pp.136-146.

[23] W. R. Daasch, K. Cota, J. McNames and R. Madge, "Neighbor selection for variance reduction in I/sub DDQ/ and other parametric data", *IEEE International Test Conference*, Baltimore MD, Oct 2002, pp. 1240-1248.