# Joint Minimization of Power and Area in Scan Testing by Scan Cell Reordering


Shalini Ghosh[1], Sugato Basu[2] and Nur A. Touba[1]

[1]Dept. of Electrical and Computer Engineering
University of Texas, Austin, TX 78712-1084

[2]Dept. of Computer Sciences
University of Texas, Austin, TX 78712-1084



## Abstract

*This paper describes a technique for re-ordering of scan cells to minimize power dissipation that is also capable of reducing the area overhead of the circuit compared to a random ordering of the scan cells. For a given test set, our proposed greedy algorithm finds the (locally) optimal scan cell ordering for a given value of $\lambda$, which is a trade-off parameter that can be used by the designer to specify the relative importance of area overhead minimization and power minimization. The strength of our algorithm lies in the fact that we use a novel dynamic minimum transition fill (MT-fill) technique to fill the unspecified bits in the test vector. Experiments performed on the ISCAS-89 benchmark suite show a reduction in power (70% for s13207, $\lambda$ = 500) as well as a reduction in layout area (6.72% for s13207, $\lambda$ = 500).*


## 1. Introduction

Two drawbacks of scan testing are the power dissipation and the area overhead. In CMOS circuits, power dissipation is proportional to the amount of switching activity that takes place [3]. During scan testing, a much larger percentage of the scan cells will change value in each clock cycle than during normal operations. This excessive switching activity during scan testing can cause power dissipation in the circuit to be very high. Another drawback of scan testing is the area overhead. As suggested in [8], one of the biggest components adding to the area overhead is the stitching wire between consecutive cells in the scan chains. Different techniques for controlling power dissipation during testing have been proposed in [2], [4], [6], [11], while various techniques for reducing the area overhead have been proposed in [7], [8].

In this paper, we describe a technique for minimizing power dissipation during scan testing. Our technique is also capable of reducing the area overhead of the circuit, with respect to random ordering of the scan cells. For a given set of test-vectors, we find the (locally) optimal re-ordering of the scan cells that minimizes a score function, which is a linear combination of the power and the area overhead. The score function has a trade-off parameter $\lambda$ that can be used by the designer to specify the relative importance of area overhead minimization and power minimization – increasing $\lambda$ causes a decrease in the power dissipation in the circuit, at the cost of increased area overhead. We propose a greedy algorithm for finding the best ordering for a given value of $\lambda$. The strength of our algorithm lies in the fact that we use a novel dynamic minimum transition fill (MT-fill) of the 'X' (i.e. unspecified) bits in the test vector. The method of doing "on-the-fly" MT-fill of the test vector matrix while calculating the optimal ordering gives us a better power reduction in the re-ordered matrix.

We ran experiments on standard benchmark circuits and show power versus area overhead trade-off plots. These plots provide the designer with the flexibility of giving more importance to minimizing power or area overhead, according to the design requirements, by choosing a suitable value of the design parameter $\lambda$. Our experiments show that with a proper choice of the parameter $\lambda$, our algorithm is quite effective in reducing the power in a circuit. For example, power in the circuit s13207 was reduced by 70%, for $\lambda$ = 500. It is also capable of reducing the area overhead of the circuit, with respect to random ordering of the cells. For example, layout area overhead in the circuit s13207 was reduced by 6.72%, for $\lambda$ = 500.

## 2. Estimation of Power

In CMOS circuits, the predominant fraction of power is dissipated when circuit elements switch from logic 0 to 1 or vice versa. We assume that if the primary inputs of the circuit under test (CUT) are directly controllable from the chip pins, then they are held constant during scan-in, so that all switching activity during scan-in is due to the transitions in the scan chain.

The best way to estimate power during scan testing would be to do actual circuit simulation to find the number of circuit elements that switch when a vector is scanned in. However this procedure is very time-intensive. Instead, we use the weighted transitions metric proposed by [9] to estimate the scan-in or scan-out power. According to this model, the sum of average weighted scan-in transitions and scan-out transitions is fairly closely correlated to the average number of circuit elements that make transitions in the CUT, if the weight assigned to a transition is the difference between the size of the scan chain and the position in the vector in which the transition occurs [9].

## 3. Estimation of Area overhead

In an algorithm that re-orders the scan chains to reduce the power dissipation, the main concern is whether the re-ordering increases the area overhead of the circuit. We have used two terms that give a measure of the area overhead. One measure of the area overhead is the layout area, which measures the overall area of the chip. Another heuristic measure of area that we use is the approximate area overhead, which is the sum of the Manhattan distances between two consecutive cells in the scan chain. This is an estimate of the stitching-wire length between consecutive cells in the scan chain, and is one of the biggest components to the area overhead [8]. Manhattan distance gives the estimate of the routing complexity – the longer the Manhattan distance, the greater is the routing complexity and the designer cannot compact area as much.

## 4. Minimum-transition Fill (MT-fill)

One important feature of our optimization algorithm is the fact that we do MT-fill of test vectors "on-the-fly", which we call dynamic MT-fill. Details of this method will be explained in Section 6. Here we give a background of the MT-fill technique.

Consider a test vector matrix that has 0, 1 and X entries, where each row of the matrix corresponds to a test vector for the circuit. X is an unspecified value and can be filled with either 0 or 1. The conventional approach for filling the X's in the test cube is to do random fill (R-fill) in which the X's are randomly replaced by 0's or 1's. In R-fill, the idea is that it increases the chance that a single test cube would detect additional faults and hopefully the other test cubes would not be required and can be eliminated during reverse fault-simulation. However, since we are considering power, which involves the number of weighted transitions in the test vector, it is best to consider Minimum Transition Fill (MT-fill). In MT-fill, a series of X entries in the test vector are filled with the same value as the first non-X entry on the right side of this series. This minimizes the number of transitions in the test vector when it is scanned in. For example, consider the test vector: 100**XX**010**X**1**X**0. This vector, after MT-fill, would become 100**00**010**1**1**0**0. If the test vector has a string of X bits that is not terminated by a non-X bit on the right side, then it should be filled by the bit value to the left of the sequence. For example: 1000001011**XX** should be 1000001011**11** after MT-fill.

## 5. Scan Reorder Methodology

In the conventional approach, a gate-level netlist is generated after design synthesis. After that, scan insertion is done and then placement and routing is performed. In our scan methodology, after scan insertion we do placement to get the initial co-ordinates of the scan cells in the circuit. These co-ordinates are used for finding the best ordering of the scan cells using our proposed algorithm, which tries to do joint minimization of power and area overhead. The scan chain cells are re-connected according to this new ordering, after which placement and routing is done again to get the new co-ordinates of the re-ordered scan cells. The test vectors are also reordered according to the new ordering of the scan cells.

## 6. Algorithm for Ordering Scan Cells

In our algorithm (shown in Fig. 1), we start with a test vector matrix that has 0, 1 and X entries, where each row of the matrix corresponds to a test vector for the circuit. Each test vector column corresponds to a scan cell in the scan chain, such that ordering the columns in the test vector matrix is equivalent to ordering the scan cells in the scan chain. To jointly minimize power and area overhead, we define a score function between two columns $i$ and $j$ of a test vector matrix: $score(i,j)=distance(i,j)+\lambda*power(i,j)$, where $\lambda$ is the trade-off parameter between distance and power, $distance(i,j)$ is measured by the Manhattan distance between the scan cells $i$ and $j$, while $power(i,j)$ is measured by weighted transitions as explained in Section 2.

For a given value of $\lambda$, the algorithm starts by choosing the two columns in the test matrix that have the minimum score between them, in the function *findBestSeeds*. Considering that the test vectors are inserted into the scan chain from the left, the transitions between the two columns on the rightmost side of the re-ordered test matrix would have the maximum weight. So, we greedily seed the re-ordering process by assigning the two columns with minimum score between them as the rightmost columns in the re-ordered matrix. After that, the column before the rightmost one is MT-filled with respect to the rightmost column. We refer to this process of doing MT-fill "on-the-fly" as *dynamic MT-fill*. In the main loop of the function *findBestOrdering*, we consider every column $i$ in the matrix from right to left (due to the weighting scheme), starting from the column before the rightmost one. The function *greedyColumnToSwap* in the algorithm finds the column $j$ in the matrix, to the left of $i$, which has the minimum score with column $i$. It then swaps column $i-1$ with column $j$. After swapping, the new column $i-1$ is dynamically MT-filled with respect to column $i$.

The dynamic MT-fill done at every step of the re-ordering process is a novel technique, which greatly contributes to the improved performance of our algorithm. Instead of doing the dynamic MT-fill at every step, if we had performed the MT-fill on the total test vector matrix at the beginning of the algorithm, then we would have lost degrees of freedom in choosing the best values with which to fill up the X values in the test vectors in order to minimize the number of transitions. If, on the other hand,

we had chosen to do a MT-fill at the end of the algorithm, then we would have too many options for selecting the best column at every step of the algorithm. Dynamic MT-fill gives a good compromise between these two extremes, by selecting the X values in the current column so as to minimize the number of transitions at every step of the algorithm, while at the same time keeping enough degrees of freedom for selecting the best columns in the future steps. At every step of the algorithm, the overall score (power and distance) between the test matrix columns is greedily minimized. At the end of the swaps, the new column ordering is output. This process is repeated for all values of the trade-off parameter λ that are specified by the designer.

```
Input: - TestMatrix[rowSize,columnSize]
          // Test vector matrix
        - LocationsVector[columnSize]
          // Scan cell co-ordinates
        - LambdaVector[numLambdas]
          // Different values of lambda
Output: Best column ordering for each lambda value

findBestOrdering()
  for each lambda in LambdaVector {
    findBestSeed(lambda);
    Dynamic MT-fill column (columnSize-1) w.r.t.
      column columnSize;
    for (i=columnSize-1; i>=2; i--) {
      bestColumn =
        greedyBestColumnToSwap(i,lambda);
      Swap column bestColumn with column (i-1);
      Dynamic MT-fill column (i-1) w.r.t. column i; }
    Output the column ordering }
findBestSeed(lambda)
  Find the pair of columns [a,b] from TestMatrix
    that have minimum score(a,b,lambda)
  Swap column pair (a,b) with the two rightmost
    columns of TestMatrix
greedyBestColumnToSwap(i,lambda)
  Find the column j in the TestMatrix that has
    the minimum score(i,j,lambda)
score(i,j,lambda)
  Between columns i and j in TestMatrix, compute
    [wireLength(i,j)+lambda*transitions(i,j)]
wireLength(i,j)
  Compute Manhattan distance between cell i
    and cell j, using LocationsVector
transitions(i,j)
  Compute weighted 0->1 or 1->0 transitions between
    column i and column j of the TestMatrix
```

**Figure 1.** Algorithm for Scan Cell Reordering

## 7. Experimental Methodology

For a given circuit, we first insert the scan chain into it. We then run a placement and routing tool to get the locations of the scan cells and the total area overhead of the circuit. Using the test vectors and the locations of the cells, we run the greedy algorithm to get the (local) optimal ordering of the cells for different values of the scaling parameter λ. The values of approximate area overhead and estimated power of the circuit, corresponding to the ordering for a particular value of λ, are used to plot the power and the approximate area overhead for each value of λ. After looking at the plots, we choose a particular value of λ that gives a good trade-off between power and area overhead. For this value of λ, we stitch the scan-chain according to the ordering found by the algorithm. Running an area-measurement tool gives the layout area overhead for this λ.

## 8. Experimental Results

We performed experiments on the following circuits from the ISCAS-89 benchmark suite [1]: *s5378*, *s9234*, *s13207*, *s15850* and *s38417*. We used the *wolfe* tool [10] in OctTools for routing, placement, and calculating the area overhead of the circuit after placement.

The results on *s13207, s15850* and *s38417* are shown in Figs. 2, 3 and 4. The figures for the other benchmark circuits are given in [5]. In each figure, the initial estimated area of the circuit refers to the approximate area overhead of the circuit with the default ordering of the scan cells. The approximate area overhead is calculated as the sum of the lengths of the wires connecting the scan cells, i.e., the Manhattan distance between scan cells. The initial estimated power is the power estimated from the number of transitions in the test vectors with the default ordering. For each value of λ, our algorithm finds the (local) optimal ordering by minimizing the combined power and area overhead metric. For each λ, the final estimated area is the approximate area overhead of the circuit after layout and placement, with the scan cells being ordered according to the optimal ordering, while the final estimated power is the power estimated from the number of transitions in the test-vector matrix after the ordering of the scan cells.

As can be seen from the graphs, increasing the value of λ increases the final approximate area overhead and decreases the final estimated power, after the scan chain has been re-ordered using the ordering output by the algorithm. For all of our circuits, the designer can choose the value of λ to trade-off the savings in power with increase in approximate area overhead by looking at the graphs.

For each circuit, we chose a particular trade-off value of λ. For that value of λ, the percentage reduction of the actual layout area overhead of the circuit after scan-chain

re-ordering was calculated, by running a placement and routing tool on the circuit with the reordered scan cells. For the trade-off value of λ chosen by us corresponding to each circuit, the percentage reduction in estimated power and actual layout area is shown in Table 1. Our algorithm achieves very good reduction in estimated power and approximate area overhead for all the benchmark circuits.

**Table 1.** Results showing reduction in estimated power and layout area for optimal chosen values of λ

| Bench-mark circuits | Size of scan chain | Chosen λ value | Estimated power reduction | Actual area reduction |
|---|---|---|---|---|
| s5378 | 164 | 100 | 48.9% | 4.8% |
| s9234 | 211 | 100 | 47.2% | 3.8% |
| s13207 | 638 | 500 | 70.2% | 6.7% |
| s15850 | 534 | 500 | 58.3% | 5.4% |
| s38417 | 1636 | 1000 | 61.5% | 5.0% |

## References


[1] F. Brglez, D. Bryan, and K. Kozminski, "Combinational profiles of sequential benchmark circuits", *Proc. of Intl. Symposium on Circuits and Systems*, pp. 1929-1934, 1989.

[2] V. Dabholkar, S. Chakravarty, I. Pomeranz and S.M. Reddy,"Techniques for Minimizing Power Dissipation in Scan and Combinational Circuits During Test Application", *IEEE Trans. on Computer-Aided Design*, Vol.17, No. 12, pp. 1325-1333, 1998.

[3] S. Devadas and S. Malik, "A Survey of Optimization Techniques Targeting Low Power VLSI Circuits", *Proc. of Design Automation Conference*, pp. 242-247, 1995.

[4] S. Gestendorfer and H.J. Wunderlich, "Minimized Power Consumption for Scan-Based BIST", *Proc. of Intl. Test Conference*, pp. 77-84, 1999.

[5] S. Ghosh, S. Basu and N.A. Touba, "Joint Minimization of Power and Area in Scan Testing by Scan Cell Reordering", CERC Technical Report, UT-CERC-TR-NAT02-1, University of Texas, Austin, 2002.

[6] P.Girard, L. Guiller, C. Landrault and S. Pravossoudovitch, "A Test Vector Inhibiting Technique for Low Energy BIST Design", *Proc. of VLSI Test Symposium*, pp. 407-412, 1999.

[7] K.H. Lin, C.S. Chen and T.T. Hwang, "Layout-driven Chaining of Scan Flip-Flops", *IEE Proc. Comput..Digit.Tech.*, Vol. 143, No. 6, 1996.

[8] S. Makar, "A Layout-Based Approach for Ordering Scan Chain Flip-Flops", *Proc. of Intl. Test Conference*, pp. 341-347, 1998.

[9] R. Sankaralingam, R. Oruganti and N. Touba, "Static Compaction Techniques to Control Scan Vector Power Dissipation", *Proc. of IEEE VLSI Test Symposium*, pp. 35-42, 2000.

[10] C. Sechen and A. Sangiovanni - Vincentelli. "The TimberWolf Placement and Routing Package", *IEEE Journal of Solid State Circuits*, vol. 20, pp. 510-522, 1985.

[11] S. Wang and S.K.Gupta, "ATPG for Heat Dissipation Minimization for Scan Testing", *Proc. of Design Automation Conf.*, pp. 614-619, 1997.


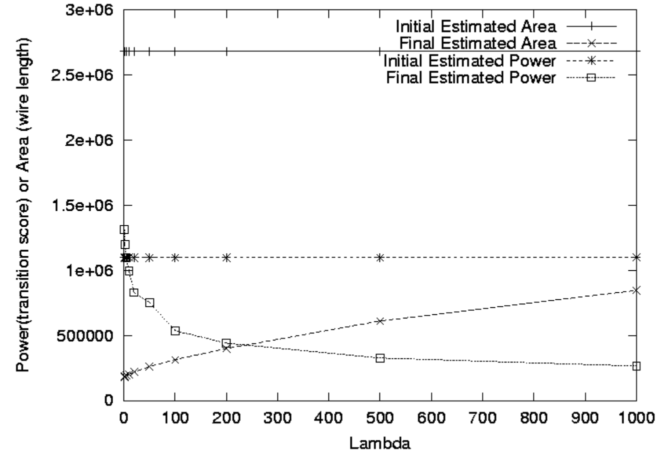


**Figure 2.** Results for s13207

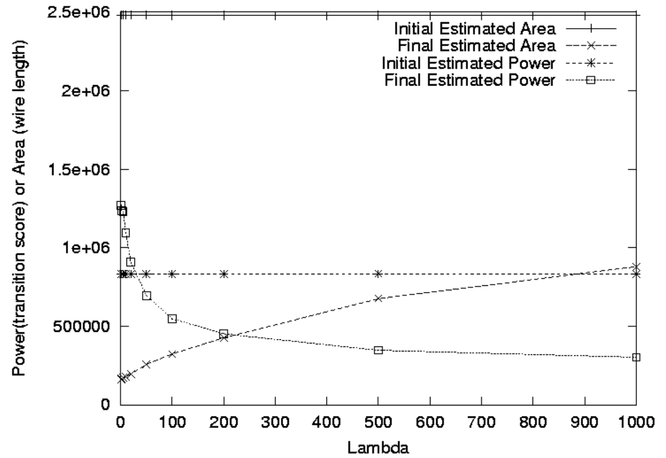


**Figure 3.** Results for s15850

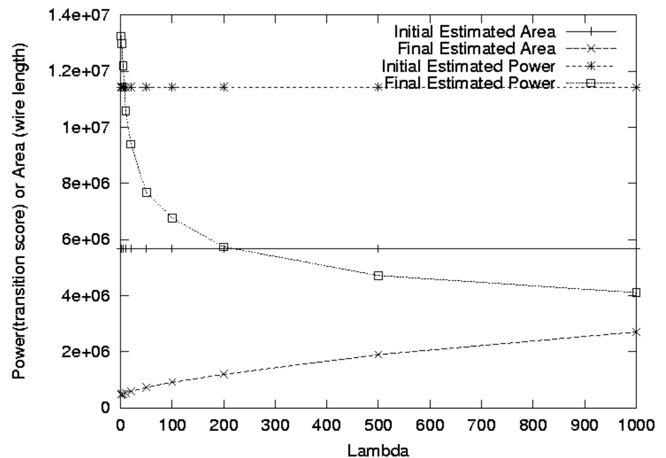


**Figure 4.** Results for s38417