Multilevel Full-Chip Routing with Testability and Yield Enhancement

Katherine Shu-Min Li[1], C.-L. Lee[1], Yao-Wen Chang[2], Chauchin Su[3], Jwu E. Chen[4]

[1]Department of Electronics Engineering, National Chiao Tung University, Hsichu, Taiwan
[2]Department of Electrical Engineering & Graduate Institute of Electronics Engineering, National Taiwan University, Taipei, Taiwan
[3] Department of Department of Electronic Control, National Chiao Tung University, Hsichu, Taiwan
[4]Department of Electrical Engineering, National Central University, Chungli, Taiwan

Abstract

We propose in this paper a multilevel full-chip routing algorithm that improves testability and diagnosability, manufacturability, and signal integrity for yield enhancement. Two major issues are addressed. (1) The oscillation ring (OR) test and its diagnosis scheme for interconnect based on the popular IEEE P1500 are integrated into the multilevel routing framework to achieve testability enhancement. We augment the traditional multilevel framework of coarsening followed by uncoarsening by introducing a preprocessing stage that analyzes the oscillation ring structure for better resource estimation before the coarsening stage, and a final stage after uncoarsening that improves testability to achieve 100% interconnect fault coverage and maximal diagnosability. (2) We present a heuristic to balance routing congestion to optimize the multiple-fault probability, chemical mechanic polishing (CMP) and optical proximity correction (OPC) induced manufacturability, and crosstalk effects, for yield improvement. Experimental results on the MCNC benchmark circuits show that the proposed OR method achieves 100% fault coverage and the maximal diagnosis resolution for interconnects, and the multilevel routing algorithm effectively balances the routing density to achieve 100% routing completion. Compared with [24], the experimental results show that our router improves the maximal congestion by 1.24X--6.11X in runtime speedup by 1.08X--7.66X, and improves the average congestion by 1.00X--4.52X with the improved congestion deviation by 1.37X--5.55X.

1. Introduction

With ever decreasing feature sizes and increasing chip dimensions, the integration complexity in system-on-a-chip (SOC) designs grows dramatically [1]. The high integration complexity is not only caused by the huge number of transistors and interconnects fabricated in a single chip, but also the modern SOC design issues in testability, manufacturability, and signal integrity. In particular, it is well known that interconnect delay dominates the circuit performance for nanometer IC designs. Therefore, it is desirable to handle the large-scale interconnect integration considering testability and diagnosability (defect reduction, yield enhancement, etc), manufacturability (process variation control, optical proximity correction [OPC], etc), and signal integrity (crosstalk minimization, etc) simultaneously.

Testability and diagnosability are very important issues for interconnect design in SOC ICs. Plenty of research works on interconnect testing can be found in the literature. Earlier works on interconnect testing were targeted for board-level testing. However, it is very difficult to apply these interconnect testing methods under the SOC environment without design-for-testability (DFT) support. The popular IEEE P1500 [7] provides a structural support for core testing as well as interconnect testing in SOC. The P1500 SOC test environment consists of a centralized test access mechanism (TAM) and wrappers around cores. The TAM defines the test control, while the wrappers provide a standardized interface for test data transmission. An oscillation ring test (ORT) [9] method for interconnect test was proposed to detect not only stuck-at and open faults, but also delay and crosstalk glitch faults. Many testing and diagnosis problems are incurred by particular interconnect structures, which can be partly solved by carefully determining the interconnect structures. Further, to reduce the probability of multiple faults, it is desirable to reduce wiring congestion in a specific area. This approach is specifically important as the probability of back-end-of-line (BEOL) defects (i.e., high-resistance via and interconnect defects) increases [6]. Therefore, many issues with testability and diagnosability should be addressed during routing.

As technology advances, the manufacturing process increasingly constrains physical layout design and verification [3]. The *chemical-mechanical polishing* (CMP) technology [4],[5] is widely used to increase the metal layers integrated in a single chip. CMP induced variation is kept within acceptable limits by controlling local feature (interconnect) density, relative to a process-specific "window size," to achieve global planarization for manufacturability and performance. Thus, balancing interconnect density minimizes the CMP induced variation, and thus routing plays an important role in determining the variation.

OPC is one of the most effective methods adopted to compensate for the light diffraction effect, typically used as a post layout process to improve manufacturability [10]. Recently, Huang and Wong proposed an algorithm that considers the OPC effect during routing by utilizing a symmetrical property. However, the process is time-consuming, and its results are still limited by the original layout quality. Again, balancing interconnect density can improve the OPC effects efficiently and effectively since the effects are also influenced by neighboring structures and shapes.

Signal integrity is an important factor that affects yield in nanometer IC technology [7]. Crosstalk affects the signal integrity in nanometer IC technology. Two adjacent wires form a coupling capacitor, and a signal changes on an aggressor net can interfere with the signal on a victim net. There are two types of crosstalk effects. One is glitch, which might induce malfunctioning in the logic values of circuit nodes and differ from what we design; the other is crosstalk-induced delay, caused by opposite switching signals in adjacent wires that slow down both signals. Crosstalk is also a crucial issue in modern router design [8].

In this paper, we handle the modern SOC design issues of testability and diagnosability, manufacturability, and signal integrity simultaneously in the routing stage for yield improvement (see Figure 1(a)). Traditionally, those issues are tackled at the post-layout stage. With the increasing design complexity, it is very difficult and even infeasible to handle those issues at the post-layout stage when most interconnect layouts are fixed and not flexible to be changed. In particular, those design issues can all be improves through balancing the routing congestion (see Figure 1(b)). Therefore, we shall present a congestion-driven routing algorithm for yield improvement.

We shall first review some important routing work. Traditionally, the complex routing problem is often solved by using the two-stage approach of global routing followed by detailed

routing. Global routing first partitions the routing area into tiles and decides tile-to-tile paths for all nets while detailed routing assigns actual tracks and vias for nets. Many routing algorithms adopt a flat framework of finding paths for all nets. Those algorithms can be classified into sequential and concurrent approaches. Early sequential routing algorithms include maze-searching approaches [12] and line-searching approaches [13], which route net-by-net. Most concurrent algorithms apply network-flow [13] or linear-assignment formulation [14],[15] to route a set of nets at one time.

The major problem of the flat framework lies in their scalability for handling larger designs. As technology advances, technology nodes are getting smaller and circuit sizes are getting larger. To cope with the increasing complexity, researchers proposed to use hierarchical approaches to handle the problem by dividing a routing region into subregions and routing each subregion independently. Marek-Sadowska [15] proposed a hierarchical global router based on linear assignment. Chang, Zhu, and Wong [14] applied linear assignment to develop a hierarchical, concurrent global and detailed router for FPGA's.

The two-level, hierarchical routing framework, however, lacks information for the interactions among the subregions and is thus still insufficient in handling the dramatically growing complexity in current and future IC designs [16]. Therefore, it is desired to employ more levels of routing for very large-scale IC designs. The multilevel framework has attracted much attention in the literature recently. It employs a two-stage technique: coarsening followed by uncoarsening. The coarsening stage iteratively groups a set of circuit components (e.g., circuit nodes, cells, modules, routing tiles, etc) based on a predefined cost metric until the number of components being considered is smaller than a threshold. Then, the uncoarsening stage iteratively ungroups a set of previously clustered circuit components and refines the solution by using a combinatorial optimization technique (e.g., simulated annealing, local refinement, etc). The multilevel framework has been successfully applied to VLSI physical design. For example, the famous multilevel partitioners, *ML* [17], and *hMETIS* [18] the multilevel placer, *mPL* [19], and the multilevel floorplanner/placer, *MB*-tree* [20], all show the promise of the multilevel framework for large-scale circuit partitioning, placement, and floorplanning. A framework similar to multilevel routing was presented in [21], [22]. Lin, Hsu, and Tsai in [22] and Hayashi and Tsukiyama in [21] presented hybrid hierarchical *global* routers for multi-layer VLSI's, in which both the bottom-up (coarsening) and top-down (uncoarsening) techniques were used for global routing. Recently, Cong, Fang, and Zhang proposed a pioneering multilevel global-routing approach for large-scale, full-chip, routability-driven routing [16]. Cong, Xie, and Zhang later proposed an enhanced multilevel routing system, named MARS [23], which incorporates resource reservation, a graph-based Steiner tree heuristic and a history-based multi-iteration scheme to improve the quality of the multilevel global routing algorithm in [16]. The final tile-to-tile paths for all the nets are then fed into a detailed router to find the exact connection for each net. Lin and Chang also proposed a novel multilevel framework for full-chip routing, which considers both routability and performance [24]. This framework integrates global routing, detailed routing, and resource estimation together at each level, leading to more accurate routing resource estimation during coarsening and thus facilitating the solution refinement during uncoarsening. Their experimental results show the best routability among the previous works. Recently, Ho, et al. proposed yet another multilevel framework by introducing an intermediate layer and track assignment stage between coarsening and uncoarsening to handle crosstalk minimization [26]. A multilevel routing considering antenna effects was recently presented by Ho, Chang, and Chen in [27].

In this paper, we propose a multilevel full-chip routing framework considering testability and diagnosability, manufacturability, and signal integrity simultaneously. Different from the previous works, our approach has the following distinguished features:

- Consider testability and diagnosability, manufacturability, and signal integrity simultaneously in the multilevel routing framework.
- Propose a new testability-driven multilevel routing framework, consisting of a preprocessing stage for oscillation ring test (ORT) generation for interconnect, a coarsening stage, an intermediate stage for optimization, an uncoarsening stage, and a postprocessing stage to process diagnosis patterns for ORT.
- Provide testability and yield enhancement solutions in the routing stage to both diagnose interconnects and improve density flexibility.
- Present heuristics to balance and reduce congestion in routing for yield improvement (by reducing multiple fault probability, CMP variation, OPC effects, and crosstalk).

Experimental results on the MCNC benchmark circuits show that the proposed OR method achieves 100% fault coverage and maximal diagnosis resolution for interconnects, and the multilevel routing algorithm effectively balances the routing density to achieve 100% routing completion. Compared with [24], the experimental results show that our router improves the maximal congestion by 1.24X--6.11X in runtime speedup by 1.08X--7.66X, and improves the average congestion by 1.00X--4.52X with the improved congestion deviation by 1.37X--5.55X.

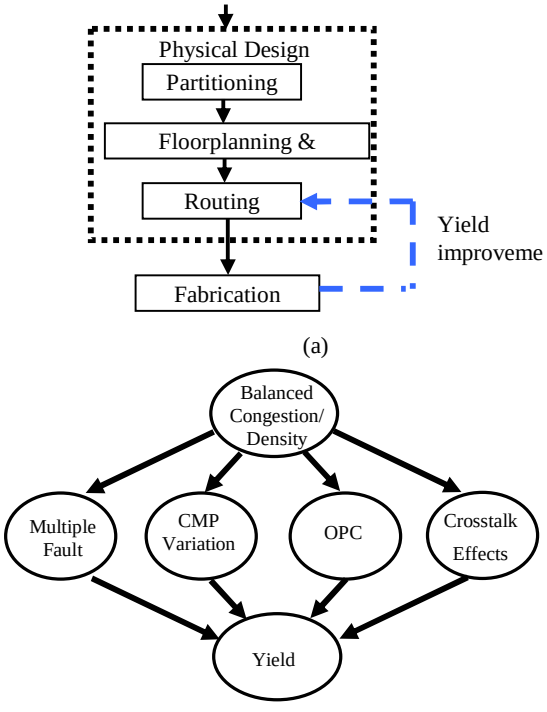


Figure 1: (a) Testability enhancement in routing stage. (b) Balancing routing congestion reduces multiple fault probability, CMP induced variation, OPC, and crosstalk, all of which improve yield.

This paper is organized as follows. Section 2 gives a brief review of oscillation ring test and diagnosis. Section 3 presents the multilevel routing framework. Experimental results are reported in Section 4, and concluding remarks follow in Section 5.

2. Preliminaries

2.1 The OR Test Architecture for Interconnect

In this section, we discuss the oscillation ring test for interconnects. Oscillation ring (OR) test is a useful and efficient method to detect faults in SOC interconnect [9]. An oscillation ring is a closed loop of a circuit under test in which has an *odd number of signal inversions*. Once the ring is constructed during test mode, oscillation signal appears on the ring. Figure 2 illustrates a global counter-based test architecture for both delay and crosstalk glitch detection for SOC ICs. This test architecture implements the IEEE P1500 core test standard, in which each input/output pin of a core is attached with a *wrapper cell*, and a centralized test access mechanism (TAM) is provided to coordinate all test process. In additional to the normal input/output connections, all wrapper cells in a core can also be connected with a shift register, which is usually referred to as a scan path, to facilitate test access. A modified wrapper cell design has been proposed to provide extra connections and inversion control so that the oscillation rings can be constructed through the wires and the boundary scan paths in cores [9]. For example, the ring in Figure 2 consists of one oscillation ring and a neighboring net, and two scan paths in cores C_1 and C_2 form the oscillation ring.

This test architecture can detect stuck-at, open, and delay and crosstalk glitch faults. If an oscillation ring fails to oscillate, it implies that there exists stuck-at or open fault(s) in the oscillation ring. The period of the oscillation signal can also be measured by using a delay counter in a core to test delay faults, and a similar approach can be used for crosstalk glitch detection.

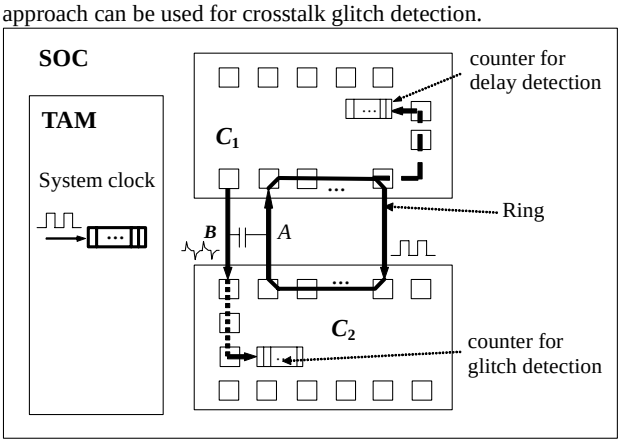


Figure 2. Test architecture for delay and crosstalk detection and delay measurement.

A local counter is included in each core, and a central counter is in the TAM of the chip. The central counter in the TAM is enabled by signal *OscTest* and triggered by the system clock. A local counter is connected to one wrapper cell in each core; however, it can be accessed by every wrapper cell through the wrapper cell chain. When an oscillation ring passes a core, an internal scan path is formed to connect the oscillation signal to the local counter. For example, consider core C_1, in which the oscillation ring pass by (see Figure 2). The oscillation signal is fed to the local counter through a series of modified wrapper cells that are configured as SI→SO. When an oscillation test session starts (*OscTest* = 1), the TAM enables its own central counter as well as all local counters in cores. After the counter in the TAM counts to a specific number n, the oscillation test session terminates and all local counters are disabled (*OscTest* = 0). Then all the local counter contents can then be scanned out to ATE for inspection.

Assume that m oscillation rings are tested. Let the frequency of the system clock be f, and the delay counter contents of the rings be $n_1, n_2, \ldots, n_m$, respectively. An estimation of the i-th ring's oscillation frequency f_i can be approximated by

$$f_i = f \times n_i / n \qquad (1)$$

Since the frequency of each ring is predetermined during the design phase, a delay fault can thus be detected and measured as compared with the result of the counters.

2.2 Process Variation Effects on Oscillation Signals

In order to consider process variation effect on this proposed OR scheme, we conducted an experiment for a ring consisting of 7 inverters (plus transmission gates) and 20μm lines. The Monte Carlo simulation was conducted by changing the W/L ratio of all transistors and the R, C parameters of the nets. The mean was the nominal value, while the distribution was Gaussian with $3\sigma = 20\%$ of the nominal value. In all, 30 simulation runs were performed, and the simulation results are shown in Figure 3, in which all oscillation signals start at time 0. At the end of the first cycle, there is a small variation in the cycle length, and the variations are less than 0.9% of the nominal period of the oscillation signal. The simulation results show that (1) this scheme can oscillate with an odd number of inversions, and (2) the process variation effects with 20% variance contribute to less than 0.9% in the frequency and oscillation period.

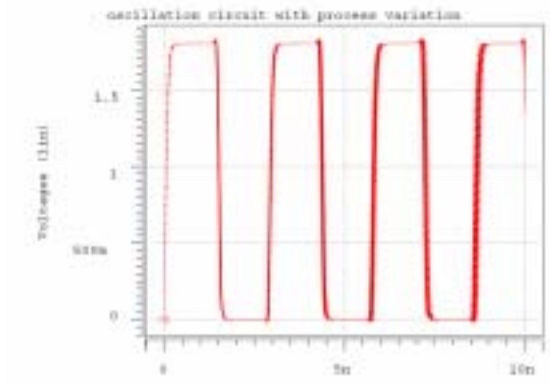


Figure 3. Simulation waveform with process variation effects on the oscillation ring test scheme.

2.3 Interconnect Model in Oscillation Ring Test

A multi-terminal net is usually modeled by a *hypergraph*. The circuit structure of an SOC can be directly transformed into a hypergraph, in which each vertex denotes a pin while each *hypernet* represents a signal net. However, this graph model is not good enough for the OR test problem, as two branches of a net should belong to two different rings, and they cannot be tested simultaneously [9]. Therefore, it would be better to consider each branch of a hypernet separately, instead of treating them as a whole. Each branch of a hypernet thus corresponds to a 2-pin net, which connects the source vertex to one of its sink vertices. An n-terminal

hypernet is thus broken into (n–1) 2-pin nets. The result is a normal graph $G = (V, E)$, where E is the set of 2-pin nets.

A complete test for all interconnections is thus reduced to the problem of finding a set of rings that cover all edges corresponding to the interconnection structure in the graph G. This is equivalent to finding a set of sub-circuits (rings) $R = \{G_1, G_2, \ldots, G_n\}$, such that

- $\forall G_i, G_i \subseteq G$, $G_i = (V_i, E_i)$, G_i is a ring, and
- $\bigcup_{i=1}^{n} E_i = E$.

If delay fault is considered, signal delay on each net along the ring should also be considered. The period of the oscillation signal is thus the summation of the path delay on all wires and scan paths. A large delay on an interconnect wire can be detected by observing the frequency of an oscillation signal that passes the wire under consideration. The detection can be masked by the variation of delays on other wires in the same ring, and thus the control of process variation is crucial for the correct detection.

2.4 Diagnosis with Oscillation Ring Tests

Diagnosis is the process of locating the exact fault site. The oscillation ring test can also be used for interconnect diagnosis. For interconnect diagnosis, the two-pin net model is also not sufficient. Consider the 4-terminal net shown in Figure 4(a), which is divided into five edge segments e_1 to e_5. If edge e_1 is faulty, all three rings will not oscillate correctly. A faulty e_3 affects rings 2 and 3, while faults on edges e_2, e_4, and e_5 affect rings 1, 2, and 3, respectively. For diagnosis purpose, all these five segments are different.

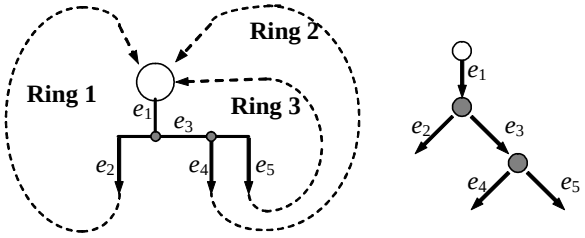


Figure 4. (a) Hypernet, and (b) diagnosis graph model

From the above discussion, it is obvious that hypernets cannot be used for diagnosis. Therefore, the interconnect structure is transformed into a graph model as follows. The scan path and wrapper cells in a core are lumped into a single *terminal node*, as we assume that they are fault-free. The fanout points of a hypernet form dummy *intermediate nodes*, and a wire segment connecting two nodes is an edge. For example, the diagnosis graph model for the hypernet of Figure 4(a) is shown in Figure 4(b), in which the white node is a terminal node and gray nodes are intermediate nodes. An edge is the smallest unit of a wire segment that can be uniquely diagnosed. From the above discussion, it can be seen that any stem affects all the downstream nodes and edges.

3. The Multilevel Routing Framework

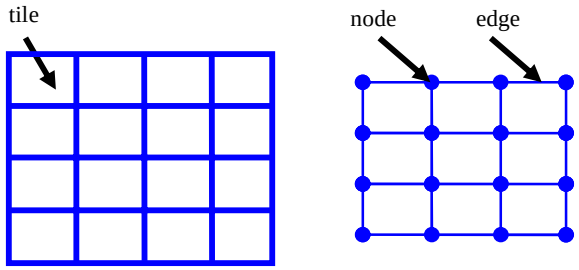
We propose in this section a new multilevel routing framework, as illustrated in Figure 6, that considers routability, performance, testability, diagnosability, process variation, and crosstalk. The oscillation rings for test are based on circuit connectivity, and thus they can be constructed before routing. However, when delay fault is considered, the routing structure must also be considered, since the wire delay is mainly decided by the wire length. On the other hand, the diagnosis process has to consider the actual net layout, and they must be considered after the routing process.

3.1 Routing Model

Our global routing algorithm is based on a graph search technique guided by the congestion information associated with routing regions. The router assigns higher costs to route nets through congested areas (or those of higher delay and/or crosstalk costs) to balance the net distribution among routing regions. Before we can apply the graph search technique to multilevel routing, we first need to model the routing architecture as a graph such that the graph topology can represent the chip structure. Figure 5 illustrates the routing graph model.

For the modeling, we first partition a chip into an array of rectangular subregions. These subregions are called *global cells* (GC). A node in the graph represents a GC in the chip, and an edge denotes the boundary between two adjacent GCs. Each edge is assigned a weight/capacity according to the physical area or the number of tracks of a GC. The graph is used to represent the routing area and is called a *multilevel routing graph*, denoted by G_k, where k is the level ID. A global router finds GC-to-GC paths for all nets on a routing graph to guide the detailed routing. The goal of global routing is to route as many nets as possible while meeting the capacity constraint of each edge and any other constraints, if specified.

As the process technology advances, multiple routing layers are possible. The number of layers in a modern chip can be more than eight. Wires in each layer can run either horizontally (H) or vertically (V) in a grid style.



(a) partitioned layout (b) routing graph

Figure 5. The routing graph.

As illustrated in Figure 6, G_0 corresponds to the routing graph of the level 0 of the multilevel coarsening stage. At each level, our global router first finds routing paths for the *local nets* (or *local 2-pin connections*) (those nets that entirely sit inside a GC). After the global routing is performed, we merge 2×2 of GC into a larger G_i and at the same time perform resource estimation for use at the next level (i.e., level 1 here). Coarsening continues until the number of GCs at a level, say the k-th level, is below a threshold. The uncoarsening stage tries to refine the routing solution of the unassigned segments of the level k. During uncoarsening, the unroutable nets are performed by point-to-path maze routing and rip-up and re-route to refine the routing solution. Then we proceed to the next level (level k–1) of uncoarsening by expanding each G_k to four finer G_{k-1}'s. The process continues until we reach level 0 when the final routing solution is obtained.

3.2 Testability-Aware Multilevel Routing

In the coarsening stage of multilevel routing, shorter nets are routed first, and a congestion-driven heuristic is used to guide a pattern router. For all the nets that can be successfully routed, both global route and detailed route are conducted. All the nets that fail to complete will be handled at the uncoarsening stage. At the uncoarsening stage, the failed nets are routed by a global router with a different cost function to avoid heavily congested area, and a detailed maze router is used to determine the final routing path. In addition to the traditional multilevel framework, we incorporate an oscillation ring test in the preprocessing stage to guide the resource estimation for interconnect and 100% fault detection coverage, an intermediate stage for interconnect optimization, and an oscillation ring diagnosis (ORD) in the postprocessing stage to guarantee maximal interconnect diagnosability (see Figure 6).

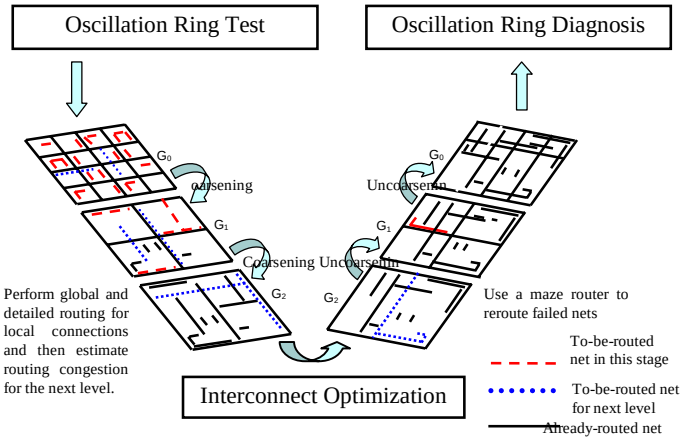


Figure 6. Integrated multilevel routing framework.

3.3 Diagnosability-Aware Routing Structure

The minimum spanning tree (MST) topology leads to the minimum total wire length, and thus congestion is often easier to be controlled for MST than other topologies. This topology may result in longer critical paths and thus degrade circuit performance. In contrast, a shortest path tree (SPT) may result in the best performance, but its total wire length (and congestion) may be significantly larger than that constructed by the MST algorithm.

The diagnosis problem also affects the routing structure. For instance, consider the 4-terminal net example shown in Figure 7. With the *spanning tree* connection given in Figure 7(a), there are three different net segments to be diagnosed. On the other hand, as the diagnosis graph model shown in Figure 4(b), for the *Steiner tree* connection given in Figure 7(b), there are two intermediate nodes (indicated by the two dotted circles) and thus five net segments to be diagnosed. In general, a spanning tree connection employed fewer wire segments to be diagnosed, and thus it is favored in our router. Our algorithm first constructs the minimum spanning tree (MST) structure whenever possible, which is best for diagnosability. Otherwise, it will find a routing tree with the least number of intermediate nodes.

(a) (b)

Figure 7. Two routing trees: (a) a spanning tree with three segments (b) a Steiner tree with the minimum number of intermediate nodes, resulting in five segments.

In order to route a net with the minimum number of intermediate branch nodes and the shortest path, we apply the algorithm shown in Figure 8(a) for the routing tree construction. The algorithm, which is based on Dijkstra's shortest path algorithm, finds a shortest path with the minimum number of intermediate nodes. It associates each basic detailed routing region u with two labels: $d(u)$ and $n(u)$, where $d(u)$ is the distance of the shortest path from source s to u, and $n(u)$ is the minimum number of intermediate nodes along the shortest path from s to u. Initially, $d(u) = \infty$, $n(u) = \infty$, $\forall u \neq s$, $d(s) = 0$, and $n(s) = 0$. The computation of label d's is the same as the original Dijkstra's algorithm. The computation of $n(v)$ is shown in Figure 8(b), where $dist(u, v)$ and $node(u, v)$ are the distance and the number of intermediate nodes between nodes u and v, respectively.

```
SPA(G, dist, s)
1. Initialize_Source(G, s);
2. S ← ∅;
3. Q ← V[G];
4. while Q ≠ ∅
5.   u ← Extract_Min(Q);
6.   S ← S ∪ {u};
7.   for each v ∈ Adj[u]
8.     Count_Nodes(u, v, dist);
```

(a)

```
Count_Nodes(u, v, dist)
1. if d(v) ≥ d(u) + dist(u, v)
2.   d(v) ← d(v) + dist(u, v);
3.   if n(v) ≥ n(u) + node(u, v)
4.     n(v) ← n(u) + node(u, v);
5.     record u as the predecessor
       routing region of v.
```

(b)

Figure 8. (a) Shortest path algorithm, (b) $n(v)$ computation.

3.4 Cost Metric for Routing Density Control

A router that incurs imbalanced routing density may degrade system performance in many ways.

- Crosstalk effects are the results of signal coupling between adjacent wires, and the coupling capacitance is usually inversely proportional to the distance between wires. In a heavily congested area, the distance between adjacent wires is small and thus the probability of crosstalk faults is increased.
- Physical defects in a congested area may create multiple faults, which are difficult to be detected and diagnosed.
- Process variation due to CMP is usually caused by unbalanced routing congestion/density.

Therefore, it is desirable to balance routing congestion/density in all areas for router design. Given a netlist, we first run the minimum spanning tree (MST) algorithm to construct the topology for each net, and then decompose each net into 2-pin connections, with each connection corresponding to an edge of the minimum spanning tree. Our multilevel framework starts from coarsening the finest tiles of level 0. At each level, tiles are processed one by one, and only local nets (connections) are routed. At each level, the two-stage routing approach of global routing followed by detailed routing is applied. The global routing is based on the approach

used in the pattern router [25] and first routes local nets on the tiles of level 0. Let the multilevel routing graph of level i be $G_i = (V_i, E_i)$. Let $R_e = \{e \in E_i \mid e$ is the edge chosen for routing$\}$. In order to balance the routing density, we use the cost function α: $E_i \rightarrow R$ to guide the routing:

$$\alpha(R_e) = \sum_{e \in R_e} c_e \qquad (2)$$

where c_e is the congestion of edge and it is defined as

$$c_e = \begin{cases} \dfrac{1}{2^{\lceil (p_e/t) - d_e \rceil}} & d_e < (p_e/t) \\ 1 & d_e \geq (p_e/t) \end{cases}$$

where p_e and d_e are the capacity (p_e) and the number of nets assigned to edge e (d_e), respectively. The parameter t is used to define the target level of the maximum density, and it can be determined either by the user or by averaging over all routing areas. For example, if the goal is to make the average routing density to be half of the maximum acceptable density, then t is set to 2.

After the global routing is completed, we perform detailed routing with the guidance of the global-routing results and find a real path in the chip. Our detailed router is based on the maze-searching algorithm. Pattern routing uses an L-shaped or a Z-shaped route to make the connection, which gives the shortest path length between two points. Therefore, the wire length is minimized, and we do not include wire length in the cost function at this stage. We measure the routing congestion based on the commonly used channel density. After the detailed routing finishes routing a net, the channel density associated with an edge of a multilevel graph is updated accordingly.

Our global router first tries L-shaped pattern routing. If the routing fails, we try Z-shaped pattern routing. If both pattern routes fail, we give up routing the connection, and an overflow occurs. We refer to a *failed net* (*failed connection*) as that causes an overflow. The failed nets (connections) will be reconsidered (refined) at the uncoarsening stage.

The uncoarsening stage starts to refine each local failed net (connection), left from the coarsening stage. The global router is now changed to the maze router with the following cost function β: $E_i \rightarrow R$:

$$\beta(R_e) = \sum_{e \in R_e} (a \cdot c_e + b \cdot o_e) \qquad (3)$$

where a, b, are user-defined parameters, and $o_e \in \{0,1\}$. If an overflow happens, o_e is set to 1; otherwise, it is set to 0.

There is a trade-off between minimizing congestion and overflow. At the uncoarsening stage, we intend to resolve the overflow in a tile. Therefore, we make b much larger than a. Also, a detailed maze routing is performed after the global maze routing. Iterative refinement of a failed net is stopped when a route is found or several tries have been made. Uncoarsening continues until the first level G_0 is reached and the final solution is found.

4. Experimental Results

The multilevel routing system was implemented in the C programming language on a 900 MHz SUN Blade 2500 workstation with 1GB memory. We conducted two sets of experiments: (1) testability enhancement, and (2) congestion control for routing considering multiple faults, manufacturability, and crosstalk. Three types of benchmarks were used in our experiments: the first type is for inter-module interconnects only (see Table I); the second is the full-chip benchmarks (only mcc1 and mcc2), which include both inter-module interconnections and intra-module interconnections; the third type contains only intra-module interconnections which are local interconnections within standard-cell modules. The results of the experiments based on type-2 and -3 benchmarks are given in Table II.

4.1. Testability Enhancement

For testability enhancement, the experimental results of the embedded OR scheme in the proposed multilevel routing framework are reported in Table I. We have presented both a detection (the preprocessing stage) and a diagnosis schemes (the postprocessing stage) as shown in Figure 6 for oscillation ring based interconnect testing in SOC in a predetermined design flow. Thus, $f_{min} \leq f_i \leq f_{max}$ gives the timing specification for this scheme, where f_i is the estimated oscillation frequency for the i-th ring. Since our target of this OR scheme is for interconnect among modules, our experiments were conducted based on the MCNC benchmark circuits with inter-module connections.

Table I gives the name of the circuit, the statistics for the circuits (the number of cores, #core; the number of pads, #pad; the number of hyperedges, #hyp; the number of 2-pin nets), the number of rings constructed for detection, $|R_t|$, and the number of rings constructed for diagnosis, $|R_d|$. Thus, $|R_t|$ is the testability-driven cost in the preprocessing stage, and $|R_d|$-$|R_t|$ is the additional cost for the postprocessing stage. In addition to the 100% fault coverage of the oscillation ring detection scheme, we also obtained 100% net segment diagnosability.

To show the feasibility of this scheme, we include the actual estimated ATE measurement times in the parentheses in Table I. Since the frequency of each ring is predetermined during the design phase, a delay fault can thus be detected and measured by inspecting the contents of the local core counters (see Figure 2). Let the oscillation frequency of the rings, according to the timing specification, be $f_{min} \leq f_i \leq f_{max}$, with the unit time of measuring T_0 ($= n/f$). Thus, we have delay the counter contents of $n_{min} \leq n_i \leq n_{max}$, where $n_{min} = f_{min} \times T_0$ and $n_{max} = f_{max} \times T_0$. Let ξ be the resolution of delay measurement, and ε be the maximum measurement error. Since a counter's maximum measurement error is ± 1, the requirement for ε should be the reciprocal of f_{min} times T_0.

$$\varepsilon = \frac{1}{f_{\min} \times T_0} \leq \zeta \qquad (4)$$

We show an example of the delay measurement. Let the frequency specification of the oscillation rings be 4 MHz to 400 MHz, and ξ is 0.001, which implies that the counter content d_{min} is at least 1000. From Equation (4), we have the required T_0 250μs. Thus, we get the estimated detection and diagnosis times in the parentheses. For example, for the ac3 circuit, we need 133 rings to detection and 374 rings to diagnose; therefore 133 x 250μs = 33.25 ms for interconnect detection, and 374 x 250μs = 93.5 ms for interconnect diagnosis. This shows the effectiveness and efficiency of the testability enhancement.

4.2. Congestion Control for Multi-objective Optimization

Table III reports the results for multilevel routing considering multiple faults, manufacturability, and crosstalk. We compared three different routing algorithms: (A) performance-driven MR [24], (B) routability-driven MR [24], and (C) our proposed method (with MST routing and balanced density).

Table I: Experimental results based on the MCNC benchmarks for testability enhancement of interconnect detection and diagnosis

Circuit	Statistics				#rings constructed for testability $\|R_t\|$ & diagnosis $\|R_d\|$	
	#core	#pad	#hyp	#2-pin	$\|R_t\|$	$\|R_d\|$
ac3	27	75	211	416	133(33.3ms)	374(93.5ms)
ami33	33	42	117	343	242(60.5ms)	303(75.8ms)
ami49	49	22	361	475	156(39ms)	386(96.5ms)
apte	9	73	92	136	73(18.3ms)	122(30.5ms)
hp	11	45	72	195	81(20.3ms)	164(41ms)
xerox	10	2	161	356	218(54.5ms)	342(85.5ms)

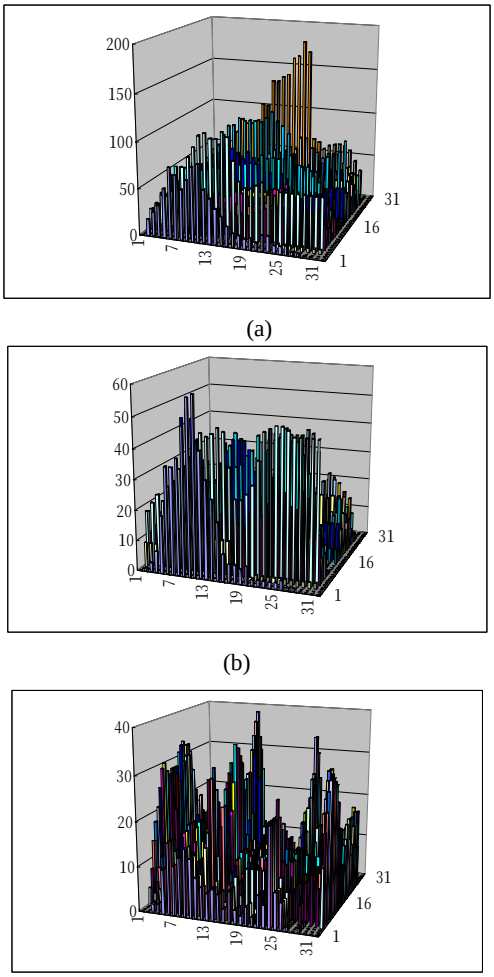
Table II: The routing benchmark circuits.

Circuit	Size (μm)	#Layers	#Nets	#Pins
Mcc1	39000×45000	4	1694	3101
Mcc2	152400×152400	4	7541	25024
Struct	4903x4904	3	3551	5717
Primary1	7552x4988	3	2037	2941
Primary2	10438x6468	3	8197	11226
S5378	4330x2370	3	3124	4734
S9234	4020x2230	3	2774	4185
S13207	6590x3640	3	6995	10562
S15850	7040x3880	3	8321	12566
S38417	111430x6180	3	21035	32210
S38584	12940x6710	3	28177	42589

In each case, we give the maximum (critical path) delay d_{max}, average delay d_{avg}, and the maximum number of nets crossing a level-0 tile #Net$_{max}$, which is a good estimate for the maximum routing density. In our experiment, we set the parameter t = 4 for the ISCAS89 circuits, while for other benchmarks were set to t = 2. The completion rate is 100% for all cases. It can be seen that the proposed method achieves about the same level of performance as the routability-driven method does by up to 0.2% increase in d_{max} and d_{avg},, but the maximum density is much smaller. Compared with [24], the experimental results show that our router improves the maximal congestion (#Net_max) by 1.24X--6.11X in runtime speedup by 1.08X--7.66X.

In Table IV, we show some statistical density results. The average number of nets crossing a level-0 tile is denoted by #Net$_{avg}$, and we also list those of vertical tiles and horizontal tiles #Net$_{avg_v}$ and #Net$_{avg_h}$ respectively. Also, σ_v is denoted for the standard deviation from the vertical tile prospect and σ_h for that of the horizontal tile prospect. The results show that our scheme is more effective for the full-chip benchmarks mcc1 and mcc2. For other intra-module routing, our scheme also improve the results for most cases. Compared with [24], the experimental results show that our router improves the average congestion by about 1.00X--4.52X, and improves the balanced congestion (σ _v and σ _h, standard deviation respective for vertical and horizontal tiles) by 1.37X--5.55X.

To demonstrate the effectiveness of the proposed algorithm in balancing the routing density, the number of horizontal wires crossing each level-0 tile for benchmark mcc1is shown in Figure 9 for the three algorithms. It can be seen that the performance-driven MR results in the least balanced routing, and the peak congestion is 181 (#Net$_{max}$) in mcc1. The routability-driven MR tries to avoid congested area to improve the probability of successful routing, and thus reduces the maximum density; its peak congestion is 61. With the proposed algorithm, the maximum density is further reduced to 45, and thus the manufacturability effects, the probability of multiple faults, and crosstalk effects are reduced accordingly. Mcc1 shows the maximal congestion improvement in our proposed algorithm by 1.36X compared to the routablility-driven MR and by 4.02X compared to the performance-driven MR. For mcc1, our proposed algorithm improves the average congestion by 1.01X--1.02X compared to the routablility-driven MR and 2.81X--2.85X compared to the performance-driven MR. For balanced congestion on mcc1, our proposed algorithm improves the result by 1.38X--1.48X compared to the routablility-driven MR and by 2.72X-3.32X compared to the performance-driven MR. For runtime speedup, our approach improves by 1.06X compared to routabillity-drive MR and by 3.08X compared to performance-driven MR.



(a)

(b)

(c)

Figure 9. Routing density distribution for mcc1 for (a) the performance-driven MR, (b) the routability-driven MR, (c) and the proposed algorithm.

Further, the interconnection congestion, as evident in the inter-module connections in mcc1 and mcc2, demonstrates the respective maximal and average congestion improvements by 1.39X--3.23X and 1.27X--2.36X with the congestion balance improvement (σ _v and σ _h, standard deviation respective for vertical and horizontal tiles) by 1.37X--2.76X.

5. Concluding Remarks

We have shown that the embedded oscillation ring test and diagnosis scheme is feasible based on the simulation results with TSMC .18 μm process technology. Also, this OR scheme achieves 100% fault detection coverage and maximal diagnosability. We have also presented an effective multilevel routing framework that applies a congestion-driven routing algorithm to reduce the multiple-fault probability, CMP and OPC induced effects, and crosstalk effects for yield enhancement.

REFERENCES

[1] Semiconductor Industry Association (SIA), International Technology Roadmap for Semiconductors (ITRS), 2001.

[2] V.K.R. Chiluvuri, "Yield optimization in physical design: a review," In Proc, of the Fifth ACM/SIGDA Physical Design Workshop, pages 198–206, 1996.

[3] W. Maly, "Moore's Law and Physical Design of ICs," (special address), in *Proc. ISPD*, 1998.

[4] G. Nanz and L. E. Camilletti, "Modeling of chemical-mechanical polishing: a review," *IEEE Trans. Semiconductor Manufacturing*, vol. 8, no. 4, pp. 382-389, 1995.

[5] Y. Chen, A. B. Kahng, G. Robins, and A. Zelikovsky, "Practical Iterated Fill Synthesis for CMP Uniformity," in *Proc. DAC*, pp. 671-674, 2000.

[6] A.B. Kahng, B. Liu, and I.I. Mandoiu, "Non-Tree Routing for Reliability and Yield Improvement," Proc. ICCAD, pp. 260-266, November, 2002.

[7] IEEE P1500 Website, http://grouper.ieee.org/groups/1500/.

[8] K.S.-M. Li, C.-L. Lee, C.-C. Su and J.-E. Chen, "A unified approach to detecting crosstalk faults of interconnects in deep sub-micron VLSI," Proc. ATS, pp. 145-150, November, 2004.

[9] K. S.-M. Li, C.-L. Lee, C. Su, and J.E. Chen, "Oscillation ring based interconnect test scheme for SOC," to be presented in ASPDAC 2005.

[10] L.-D Huang, M.D.F.Wong, "Optical Proximity Correction (OPC)-Friendly Maze Routing," in *Proc. DAC*, pp. 812-817, Jun. 2003."

[11] Lee, "An algorithm for path connection and its application," *IRE Trans. Electronic Computer*, EC-10, 1961.

[12] D. Hightower, "A solution to line routing problems on the continuous plane," in *Proc. DAW*, pp. 1-24, 1969.

[13] C. Albrecht, "Global routing by new approximation algorithms for multicommodity flow," *IEEE Trans. on CAD*, vol. 20, no. 5, pp. 622-632, May 2001.

[14] Y.-W. Chang, K. Zhu and D. F. Wong, "Timing-driven routing for symmetrical-arraybased FPGAs," *Trans. on Design Automation of Electronic Systems*, vol. 5, no. 3, pp. 433-450, July 2000.

[15] M. Marek-Sadowska, "Router planner for custom chip design," in *Proc. ICCAD*, Nov. 1986.

[16] J. Cong, J. Fang and Y. Zhang, "Multilevel approach to full-chip gridless routing," in *Proc. ICCAD*, pp. 396-403, Nov. 2001.

[17] C. J. Alpert, J.-H. Huang, and A. B. Kahng, "Multilevel circuit partitioning," *IEEE Trans. on CAD*, vol. 17, no. 8, pp. 655–667, Aug. 1998.

[18] G. Karypis, R. Aggarwal, V. Kumar, and S. shekhar, "Multilevel hypergraph partitioning: Application in VLSI domain," *IEEE Trans. VLSI Systems*, Vol. 7, pp. 69–79, Mar. 1999.

[19] T. F. Chan, J. Cong, T. Kong, J. R. Shinnerl, "Multilevel optimization for large-scale circuit placement," in *Proc. ICCAD*, pp. 171–176, Nov. 2000.

[20] S.-C. Lee, Y.-W. Chang, J.-M. Hsu, and H. Yang, "Multilevel large-scale module floorplanning/placement using B*-trees," in *Proc. DAC*, pp. 812-817, Jun. 2003.

[21] M. Hayashi and S. Tsukiyama, "A hybrid hierarchical global router for multi-layer VLSI's," *IEICE Trans. Fundamentals*, Vol. E78-A, No. 3, pp. 337–344, 1995.

[22] Y.-L. Lin, Y.-C. Hsu, and F.-S. Tsai, "Hybrid routing," *IEEE Trans. on CAD*, Vol. 9, No. 2, pp. 151–157, Feb. 1990.

[23] J. Cong, M. Xie and Y. Zhang, "An enhanced multilevel routing system," *Proc. ICCAD*, pp. 51-58, Nov. 2002.

[24] S.-P. Lin and Y.-W. Chang, "A novel framework for multilevel routing considering routability and performance," *Proc. ICCAD*, pp. 44-50, Nov. 2002.

[25] R. Kastner, E. Bozorgzadeh and M. Sarrafzadeh, "Predictable routing," in *Proc. ICCAD*, pp. 110-114, Nov. 2000.

[26] T.-Y. Ho, Y.-W. Chang, S.-J. Chen, and D.-T. Lee "A fast crosstalk- and performance-driven multilevel routing system," in *Pro. ICCAD*, pp. 382--387, San Jose, Nov. 2003.

[27] T.-Y. Ho, Y.-W. Chang, and S.-J. Chen, "Multilevel routing with antenna avoidance," in *Proc.ISPD-2004*, pp.34-40, Phoenix, Arizona, April 2004.

Table III: Comparison of routing results of maximum density with both maximum delay and average delay

Circuit	(A) Performance-Driven [24]				(B) Routability-Driven [24]				(C) Proposed Balanced Density with 100% routability			
	d_{max}	d_{avg}	#Net_max	CPU	d_{max}	d_{avg}	#Net_max	CPU	d_{max}	d_{avg}	#Net_max	CPU
Mcc1	4.65e+7	1.08e+7	181	223.68	2.03e+8	3.32e+7	61	77.11	2.03e+8	3.33e+7	45	72.63
Mcc2	7.26e+7	5.07e+6	274	5964.2	8.46e+7	5.12e+6	135	2855.5	8.51e+7	5.11e+6	96	2592.34
Comparison	0.413	0.413	3.227	2.322	0.998	0.998	1.390	1.10	1	1	1	1
Struct	1.13e+6	6.93e+4	32	307.91	1.52e+6	7.13e+4	9	56.33	1.52e+6	7.13e+4	7	56.53
Primary1	3.01e+5	3.33e+4	51	241.96	7.00e+5	5.51e+4	17	63.9	6.99e+5	5.50e+4	15	64.36
Primary2	3.91e+6	2.08e+5	91	1808.56	3.92e+6	2.09e+5	28	298.17	3.91e+6	2.09e+5	25	295.32
S5378	8.89e+4	6.38e+3	49	23.28	8.91e+4	6.39e+3	17	4.13	8.94e+4	6.41e+3	15	4.29
S9234	1.02e+5	9.4e+3	61	16.78	2.53e+5	1.19e+4	15	2.91	2.53e+5	1.19e+4	14	2.9
S13207	3.96e+5	2.04e+4	114	65.45	4.64e+5	2.04e+4	30	14.44	4.64e+5	2.03e+4	27	14.57
S15850	6.03e+5	2.89e+4	140	181.82	2.66e+6	6.68e+4	30	22.04	2.66e+6	6.67e+4	26	21.77
S38417	5.22e+5	2.93e+4	272	741.53	8.52e+6	3.94e+5	27	50.02	8.52e+6	3.94e+5	23	50.08
S38584	1.64e+6	5.83e+4	295	1453.8	1.76e+8	1.25e+7	31	127.8	1.76e+8	1.25e+7	29	122.5
Comparison	0.448	0.347	6.105	7.656	0.999	0.998	1.238	1.083	1	1	1	1

Table IV: Comparison of routing results of statistical density

Circuit	(A) Performance-Driven [24]				(B) Routability-Driven [24]				(C) Proposed Balanced Density with 100% routability			
	#Net_avg_v	#Net_avg_h	σ_v	σ_h	#Net_avg_v	#Net_avg_h	σ_v	σ_h	#Net_avg_v	#Net_avg_h	σ_v	σ_h
Mcc1	28.19	31.78	20.59	24.35	10.03	11.50	10.45	10.82	9.91	11.33	7.58	7.33
Mcc2	39.35	44.05	37.26	46.98	19.39	21.65	23.53	25.80	18.74	20.88	17.30	18.54
Comparison	2.357	2.354	2.325	2.757	1.027	1.029	1.366	1.416	1	1	1	1
Struct	4.97	4.86	4.62	5.03	1.42	1.41	1.24	1.67	1.42	1.41	1.07	1.59
Primary1	2.29	1.74	3.00	5.67	0.70	0.60	1.05	1.95	0.70	0.60	1.20	1.80
Primary2	7.22	7.49	5.56	18.23	2.05	1.85	1.59	4.57	2.05	1.85	1.56	4.45
S5378	12.53	13.46	9.16	8.40	4.38	3.44	3.45	2.13	4.40	3.46	3.44	2.10
S9234	14.16	9.99	12.91	7.04	3.95	2.56	3.25	1.62	3.95	2.56	3.24	1.60
S13207	28.43	20.49	18.40	11.08	9.30	5.93	5.77	2.76	9.29	5.92	5.23	2.81
S15850	36.61	34.48	23.89	20.42	10.29	7.41	5.63	2.92	10.31	7.41	5.39	2.91
S38417	44.58	27.38	37.36	27.94	7.31	4.27	4.75	2.17	7.3	4.27	4.44	2.18
S38584	43.99	30.53	35.93	20.12	9.06	5.80	5.74	2.86	9.05	5.79	5.43	2.88