Framework for Massively Parallel Testing at Wafer and Package Test

A. Hakan Baba
Stanford University
hakan@stanford.edu

Kee Sup Kim
Intel Corporation
kee.sup.kim@intel.com

Abstract — A novel DFT approach is introduced that enables massively parallel testing of logic devices at both wafer and package test. Parallelism is achieved by utilizing interconnection networks that are built onto a wafer probe or a tester interface unit. The financial benefits of this method in a realistic setting are also presented.

I. INTRODUCTION

With expanding market demand and increasing features and performance, there is rising pressure on the turnaround time of VLSI circuits. Industry trends are pushing the limits of the current VLSI test and validation technologies and practices. Existing product trends create a necessity for more parallelism also in VLSI test methods and flows [Singer 07]. The parts are getting cheaper and becoming more power efficient, especially for mobile and hand-held applications. Reduction in area and power consumption per chip, combined with the recent advances in test hardware allows *massively parallel testing*, *MPT*.

Existing MPT applications can be found in DRAM fabrication, where it is essential to reduce production cost with tight design considerations [Eetasia 04]. This MPT method is ideally also applicable to logic circuits for both wafer probe and package test. Several advancements in test methodology and hardware make MPT feasible for less regular structures than DRAM on a wafer:

A. Smaller Pin Access

There is an increasing number of tests that can be done without full pin access. Scan testing and direct access testing are very good examples. Standard *Test Access Mechanisms* (*TAM*) also allow tests without full pin access even in the case of functional tests [Kim 06, Bhatt 09, Patil 09].

B. Hardware Support for Massively Parallel Testing

There are multiple commercially available settings that would allow for massively parallel testing using a wafer probe. [Electroglas 09, FromFactor 09a, 09b, Tel 09] The high end wafer probe hardware used in current parallel testing efforts provides a portion of the foundation for our current work. Note that, our solution can also be applied to package test with the usage of test boards to enable parallelism.

C. Unique Contribution of this Work

Most of the current multi-site or parallel testing is done with dedicating bit slices of tester pins to each *device under test* (*DUT*). With this approach, the number of tester channels usually limits how many logic devices can be tested in parallel. In the approach introduced in this paper, such a limitation does not exist. The use of dedicated tester signal pins to each DUT has been avoided by limiting the amount of unique messages per DUT and sending identical contents to all DUTs in parallel over an interconnection network. This interconnection network could reside on a wafer probe or on a tester board depending on the application. The main contributions of this paper are:

1. Introduction of the new idea of massively parallel testing and demonstration of its practicality and effectiveness.
2. Detailed implementation of algorithms and system features to enable massive parallel testing in wafer sort and also in package test.
3. Analysis of the trade-offs and return on investment (*ROI*) associated with application of massively parallel testing in future systems.

Unlike BIST-based approaches which can also achieve massive parallelism, the presented approach allows for the use of the tester to deliver carefully generated patterns, which in many cases maybe desirable over randomly generated patterns. Section II discusses related work to parallel testing. We present the components of massively parallel testing methodology in Sec. III. Section IV describes some of the design considerations for this method, expected benefits, cost and trade-off information and simulation results. Section V contains some limitations and directions for future research and Sec. VI concludes this paper.

II. RELATED WORK

In literature there have been some suggestions to wafer level parallel test configurations. In [Chang 91] authors propose a wafer level testing scheme for linear arrays. The suggested method was to put multiplexer rings around the elements to be tested and the test data is propagated through the multiplexers. On the other hand, our parallel testing method relies on the recent developments in wafer probe technology [Mann 04], that allows very large number of contact points for each DUT, to accomplish test data transfer.

In addition, the authors in [Huang 98, Tang 99] introduce a wafer level test structure where the test output of one die is compared to the test output of its neighboring dies. They also extend the analysis to different fault distributions on the wafer and could identify the majority of the dies correctly. In this study we propose a scheme where the expected output is also transmitted to each DUT similar to the way done in [Tripp 05]. The response of each DUT will be compared to the actual expected output instead of a neighboring dies output, which may also be faulty.

Although the MPT might be limited by the complexity of the wafer probe, recently the wafer probe technology has

978-1-4244-5028-2/09/$25.00 ©2009 IEEE

advanced to a degree that a DRAM probe can contain up to 60,000 pins and a 300 mm wafer can be tested with one touch. [Aehr 04, FormFactor 09a, 09b] Comparable performance is also obtained for logic circuits. Therefore we do not expect serious limitation from the wafer probe hardware support. In addition, the MPT scheme is equally applicable to package test with the usage of a test board.

III. Methodology and Flow

A. Configuration

An example configuration of massively parallel testing (MPT) scheme using a wafer probe card can be seen in Fig. 1. The DUT directly connected to the tester is called the *root*. Parallelism is obtained by transmitting the test content to other DUTs on the wafer over an interconnection network, which is shown in red (horizontal/vertical) lines in Fig. 1. The interconnect network is built on the wafer probe card and the connections between the dies and the network are established with wafer probes. In order to connect to the interconnection network, each DUT has a *test router* module (yellow/white rectangles in Fig. 1), which is part of the chip and is located on the wafer. Although Fig. 1 shows a setup where the whole wafer is tested as a whole, depending on the applications and related costs, for example the cost of a complex wafer probe interface, partitioning the wafer and using several parallel test flows is possible.

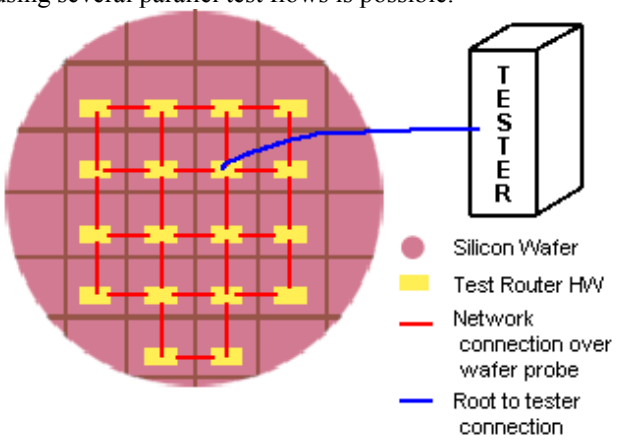


Figure 1: Configuration of MPT using a wafer probe with the network built on the probe card.

Another configuration example of massively parallel test for package test is shown in Fig. 2. Similar to Fig. 1, only one DUT is directly connected to the tester. The DUTs and the interconnection network are located on the tester board. Test router modules in each DUT are not shown in Fig. 2.

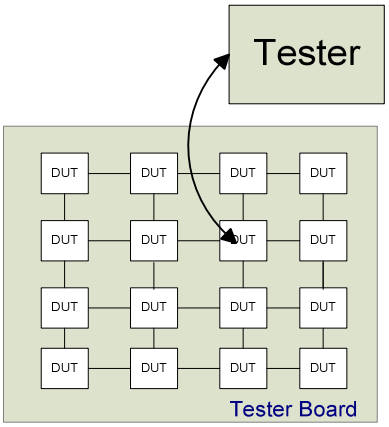


Figure 2: Configuration of MPT during package test

The number of interconnections and pins in the network are closely related to the network topology. With all the DUTs already located on a 2D plane and also considering the ease of implementation, we have decided on a mesh topology [Duato 02] for our initial work. Considerations for other network topology choices have been explored in Sec. IV.A.

In Fig. 3, a high level block diagram of the test router module is shown. The test router has 5 ports that connect to the immediate neighbors and to the tester if needed. The connections to immediate neighbors are established through the ports EAST, NORTH, WEST, and SOUTH. For example, a DUT will be connected to its right neighbor through the EAST port and to its left neighbor through the WEST port and so on. A DUT on the boundary of the wafer will not have all four ports connected to immediate neighbors due to missing dies at the edge of the wafer. Moreover, only the root will have an additional connection to the tester through the tester port. Therefore, a root will be able to identify itself (i.e., if any DUT has a valid connection through the tester port, it is the root) and the MPT flow can begin from the root.

Note that, both configurations in Fig. 1 and Fig. 2 contain only one root. Depending on the application, multiple roots might also be desirable. Further design considerations and trade-off analysis for number and location of the roots can be found in Sec. IV.A.

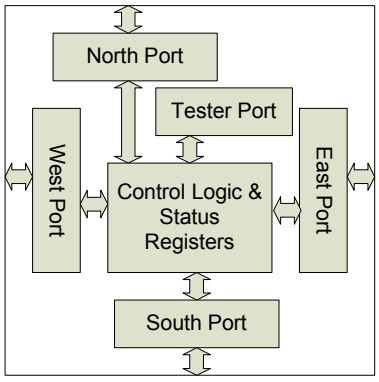


Figure 3: High level block diagram of test router module.

The status registers of the test router are shown in Table 1. Each test router will have information about the DUT where it is located and also about all the neighboring DUTs. The contents and usage of the status registers will be explained in the following sub-sections in detail.

Table 1: Status registers in test router

Status Registers	Content/usage of status registers
ID	Pair of integers for identification
Parent	Pointer to a DUT
Children	Pointer(s) to one or more DUTs
Test Data Buffer	Storage space for test content

In the following sections, we will consider the MPT flow with a wafer probe configuration only as shown in Fig. 1. The concepts and ideas are also applicable to package test using a tester board as in Fig. 2. The primary goal of the MPT flow is to test all the DUTs on the wafer and send all the test results (also test diagnosis information) to the tester over the root. The steps in the presented MPT flow can be grouped under 4 different phases; *Initialization*, *Test Content Delivery and Test*, *Test Result Collection*, and *Diagnosis Information Collection*. In the following subsections we will explain these phases in detail.

B. Initialization Phase

In the initialization phase, all the DUTs must be identified uniquely so that they can be distinguished after the test. Due to the extent of parallelism, identifying each DUT with dedicated tester channels is not a feasible solution. Moreover, each die will be identical and assuming a hardwired unique ID for each die is also not realistic. However, once the tester is connected to a root, all other DUTs can be uniquely identified using their relative positions with respect to the root. As explained in Sec. III.A, the root is self-aware and receives the root ID from the tester, for example (0,0). In the initialization phase, the IDs will be assigned similar to the coordinates in the Cartesian coordinate system. The first integer in the ID corresponds to the x-Axis coordinate and the second integer in the ID corresponds to the y-Axis coordinate. For example, the root die will send the ID (0,0) to all its neighbors. Depending on the origin of the ID message, each DUT will either subtract or add 1 for its X or Y coordinates. The example is shown in Table 2.

Table 2: ID calculation for DUT

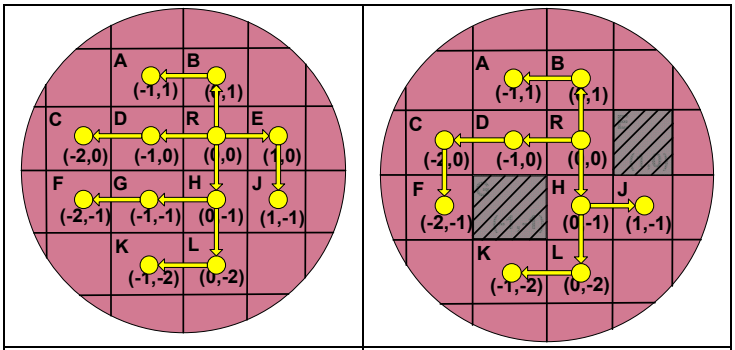
ID message source	DUT ID	example	
		ID msg content	New DUT ID
East	Sub 1 from X	(2,5)	(1,5)
West	Add 1 to X	(2,5)	(3,5)
North	Sub 1 from Y	(2,5)	(2,4)
South	Add 1 to Y	(2,5)	(2,6)

After a DUT receives an ID for the first time, it sets its ID, marks the DUT that sent the ID message first as parent and send out its ID to its neighbors that are not its parent. It is quite likely that each DUT will receive more than one ID message and they are used to validate that there was no error during the initialization phase. Figure 4 outlines the initialization process.

```
Initialization Phase{
  Root sends ID message to everyone;
  Once message is received from neighbor,
    set self-ID based on direction of message;
  Send ID to neighbors who is not the parent;
  Simple & consistent tie breaking algorithm;
}
```

Figure. 4: Initialization of DUTs and message tree

If we compare the wafer containing the DUTs and the interconnections to a graph data structure [Cormen 03], there are evident similarities. Each DUT corresponds to a vertex and each interconnection corresponds to an edge in a graph. Furthermore, the initialization phase can also be considered as a breadth first search on the graph [Stojmenovic 06]. With a breadth first search on a graph data structure, we can generate a breadth first spanning tree of a graph [Cormen 03]. Similarly, we can also maintain a breadth first spanning tree of the DUTs using the *parent* and *children* status registers. This tree will be useful in the subsequent phases of the MPT flow. Each DUT will remember the neighbor from where the ID has been received for the first time in the parent status register. Most DUTs receive several IDs from its neighbors and only one neighbor will be chosen as the parent. The children status registers are assigned similarly. For example, consider the wafer and a breadth first spanning tree rooted at R shown in Fig. 5(a). The DUT H has its parent assigned to NORTH and its children to {WEST, SOUTH}. The DUT E has its parent assigned to WEST and its children assigned to {SOUTH}. The parent status register of the root die points to the tester. A DUT which has received a unique ID and initialized the parent and children status registers is called an *initialized* DUT.



(a) No defective DUTs

(b) E & G have defective test routers

Figure 5: Initialization in presence of defects

In the initialization phase only the test router module of

each DUT is used. So any defects outside of test routers will not be observed in the initialization phase. In order to distinguish between the defect locations, a DUT having a defect at any module except the test router module is called a *defective* DUT. A DUT having a defect in the test router module is called a *test router defective* DUT. Test router defective DUTs can be identified in the initialization phase before the MPT finishes. A build-in self test (BIST) can be implemented only for the new introduced test router ensuring correct operation. Triple module redundancy might be another strategy to increase robustness of the test router. Test router defective DUTs will not connect to the interconnection network and they will appear as non-existing dies on the wafer. A similar disconnection may also occur if several wafer probe connections are faulty for a DUT. For example in Fig. 5(b), G and E have defective test routers. Their neighbors will not be able to identify them and with respect to the interconnection network, they will appear as open spaces on the network. Even in that case, the ID initialization phase has to be able to communicate with all the DUTs except the test router defective DUTs on the wafer.

As mentioned above, the ID initialization phase is equivalent to a breadth first search in a graph data structure. The breadth first search algorithm is also known as a strategy for finding shortest paths in an unweighted graph [Weiss 06]. Therefore, the ID initialization phase will reach all reachable DUTs, and assign correct IDs to them while ensuring that shortest path to each DUT is established in the form of tree network. Once all DUTs except for the test router defective DUTs are assigned a unique ID and the parent and children status registers are initialized, the initialization phase finishes in the MPT flow.

C. Test Content Delivery and Test

After the initialization phase, the test vectors are transferred to all initialized DUTs over the generated breadth first spanning tree in a pipelined fashion. Test is applied and the output is compared on the DUT, similar to [Tripp 05]. Note that, for the test output comparison for scan testing, each partition must also contain the output mask bits in addition to input vectors and expected outputs. The test content is sent to child DUTs almost as soon as they are received. This pipelined testing scheme continues until all the *leaf DUTs* receive the last test data. A leaf DUT is defined as a DUT with an empty children status register and they are also self aware like the root. For example, in Fig. 5(a) A, C, F, K and J are leaf DUTs.

D. Test Result Communication Phase

At the end of the test, each DUT will have test results that need to be communicated back to the tester. In general, it would have three categories of state. *Not Connected* state is indicated by uninitialized ID number within the DUT. *Failed* is another possible state and it can have associated 2 Kbit worth of fail signature information. *Pass* state will have a few bits of information depending on the number of passing bins that the DUT carries.

To maximize the parallel operation, results communication that doesn't include fail diagnosis information will happen in parallel also. To make this happen, during the result collection mode, special DFT feature gets activated to share some existing on-board SRAM as the result map for each die. Depending on the number of bits needed to represent different bins, these result maps would have a few bits reserved for each DUT in the network. Initially each leaf DUT sends test results to its parent DUT. Each non-leaf DUT waits for the result communication from all of its children. As the test results are received from the children, each non-leaf DUT updates its own results map, appends its own test result and sends the test results to its parent. This process continues until all the results are collected to the root DUT and to the tester. This process is outlined in Fig 6.

```
Each DUT {
  {
     Receive message from Child;
        Update fail map w/ new data;
   }until message from all children received;

  Update fail map with own fail info;
  Communicate fail map to parent;
}
```

Figure 6: Test result communication algorithm

Since the result map for many DUTs will only contain information about limited amount of other DUTs, many optimizations can be done to reduce the amount of traffic. These optimizations are not critical to the core idea presented in this paper and not detailed.

E. Diagnosis Collection Phase

If Diagnosis information is still needed during the high volume manufacturing, it is accommodated by point to point connection at the end of the testing. In diagnosis mode, each defective die (in Fail state) will communicate the on-chip diagnosis information to the root, and thus tester, in the following manner.

Each DUT will communicate upward diagnosis message from child on east until it receives done message from that child, then north, west and south, while skipping the direction where there are no connections or connections to the parent. It is done by granting diagnosis message token to its child and by forwarding message from that child to the parent until it receives DONE message from that child. It then grants diagnosis message token to the next child and so on. After sending diagnosis information from all of its children, the DUT will send its own diagnosis information, if any, and send DONE message. If DUT is a leaf node, it will send the diagnosis information as soon as it gets diagnosis token from the parent and then will send the DONE message.

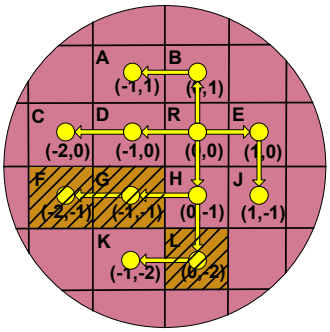


Figure 7: Diagnosis phase. Yellow/striped DUTs are defective with diagnosis information

For example consider the setup shown in Fig. 7. Assume that F, G and L are in the fail state and contain diagnosis information and each DUT searches its children in East, North, West, South order. Initially, East, North and West children of the root sends no diagnosis messages as there are no failures in those branches. H will allow G, which in turn allow F the token next. F will send diagnosis information to the root. Once F is done, G will send its diagnosis information to the root. Once G is done, H grants the token to L, which in turn grants it to K. K doesn't have any diagnosis information to send so L send the diagnosis information next. At the end, all initialized DUTs have been tested and the result messages with diagnosis information have been transmitted to the tester.

IV. DESIGN CONSIDERATIONS AND RESULTS

A. Design considerations

1) Selection of Network Topology

There are a number of network topologies that can be implemented on the wafer probe. With dies already located regularly on a 2D surface, we selected a mesh topology for its ease of analysis and implementation (Sec. III.A). In addition, in a mesh network each die will be connected to all of its four immediate neighbors. Therefore, more than one path exists to connect to the tester to the root and there will be plenty of opportunity for fault tolerance. Some of the other possibilities would be a torus which has a smaller minimum distance but the routing algorithms would be more complicated than a mesh network. Another possibility with substantially higher throughput would be a flattened butterfly topology [Dally 04]. However, this requires excessive amount of routing on the wafer probe and might be infeasible for big networks. If the number of available pins is at premium, other networks that do not use as many pins can be utilized. For example H-network can be built to reduce the number of children of a typical node from 3 to 2.

More interestingly, distinct interconnection networks with different optimizations can be implemented for various phases in the test flow. For example, a bus type structure can be used for the test content delivery while a much narrower and inexpensive network can be implemented for initialization, test result communication and diagnosis collection phases. This technique can be extended further to re-use some of the existing high speed ports such as PCI-E for the test content, namely patterns, expected output and output mask, delivery.

2) Multiple Roots

Based on the expected failure rate of the test routers, it may be advisable to have multiple roots to avoid a single point of failure. That is because, if there is only one root and it has a defective test router, the whole wafer will be in the unconnected state. Since the test router design is likely to have low complexity and contain only synthesized logic, it is possible to design it to be less sensitive to failures. One can estimate the failure rate of the root die based on the failure rate of the logic on a given process.

In addition to the case of a root with a defective test router, the root can also be circled with other dies having defective test routers as shown in Fig. 8. In that case, even if the single root does not have any faults, communication with the majority of the wafer will be impossible. If the process requires a very high yield, multiple root implementations might be explored.

If multiple roots are used, the first root will be used to test the wafer as usual. Once the test is completed, the second root will be activated. This can be accomplished with a switch connected to all candidate roots. If the second root is already initialized with a valid ID, it means that the second root was reached by the first root and no additional test flow from that root is necessary. The third and subsequent roots, if any, will again be checked to see if those roots have been initialized. If they have not been initialized, the whole test flow must be run from the uninitialized root. This multiple root scheme will trade off expected test time against yield and improve the fault tolerance of the system.

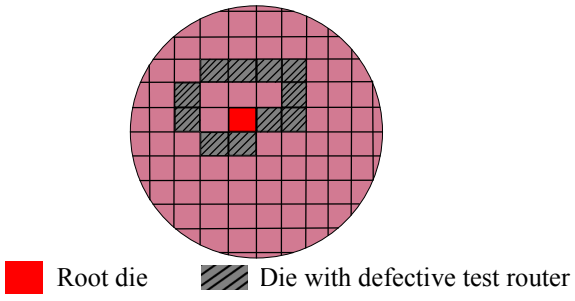


Figure 8: Isolated root die.

3) Vcc to GND Shorts

Separate power pins will be used for each DUT. The reason is that Vcc to GND shorts on any DUT can bring down the whole network unless Vcc are separated

B. Fault Tolerance

The massively parallel testing should work and produce positive return on investment (ROI) in most high volume manufacturing conditions. With the mesh topology, there are number of ways a DUT can be connected to the root even if there are a number of failed connections or failed DUTs. The fault tolerance also increases if multiple roots are used. The likelihood of the good DUT marked as bad can be

calculated. This doesn't have to be zero as long as it can be controlled to be below the threshold cost that still provides enough ROI from the massively parallel testing.

C. Implementation Status and Simulation Results

The described test flow has been modeled in C++ and various simulations have been run on real wafer maps. We take these wafer maps as inputs to our simulations and then compare the output of the parallel test flow with the information on the wafer map. In the root selection process we have only implemented a single root selection until the selected root does not have a defective test router. A multiple root scheme could also be used as described above. However, the root has not been selected completely randomly; instead it was chosen from the center region of the wafer so the distance to the wafer boundaries will be relatively symmetric. Defective dies were assumed to have 5% chance of having its test router defective. After using 50 distinct real wafer maps, the impact on yield was zero under above conditions.

The test router RTL has been completed and simulated under various scenarios. In the initial implementation, the gate count of the test router was 800 not counting the fail map, which will re-use existing SRAM on the design, or buffers for diagnosis information capture.

D. Benefits of Massively Parallel Testing

Massively parallel testing can obviously produce very high return. The actual benefit may very widely depending on the given product. Here is one possible scenario of benefits that uses very conservative estimates. Using the framework given, engineers can estimate the benefits for their own particular cases. If a product has 5mm x 5mm die size and produced on 300 mm wafer, there can be a maximum of 2500 dies per wafer. All of 2500 dies can be tested at once but if there are pin number limitations or power delivery limitations, it may not always be economically feasible to test all 2500 at the same time. Conservatively, we can test 200 at the same time both during wafer probe and package test. We will also assume that highly optimized test only takes 2 seconds to run at 200 MHz, which means 400M cycles of vectors. This test time assumption is mainly for calculating the overhead of serial communication (including diagnosis information) in the worst case. More realistic 10 second test time was used when estimating test cost savings ?

1) Diagnosis overhead

Even if we test 2500 at the same, and very conservatively assume 80% yield, the diagnosis data from 500 die need to be communicated to the root one at a time. With 8 bit width, to communicate 2K bit of diagnosis information assuming negligible communication overhead would be 125K cycles, which is .03% increase in serial test time due to diagnosis. If we test 200 at the same time, 40 die would fail at 80% yield and this would add10K cycles, which is .0025% addition in serial time. Under these scenarios, the overhead of diagnostic information communication is almost negligible and the test time reduction is very close to 200X if 200 die are tested in parallel. More realistic test time would result in even less overhead.

2) Mask Information Communication Overhead

Since mask bits may need to be sent to the DUTs in addition to inputs and outputs, this may increase test time serially by 50%. With 200X parallelism, this 150% test time still can be applied to 200 DUTs at the same time, which still is about 133x reduction in test time.

3) Other Communication Overhead

Since the communication happens in a pipelined fashion and no extensive protocol is needed in this rather controlled environment, there will be about 100 cycle delay due to initial pipeline fill up. At the beginning, the initialization for 2500 die wafer would take about 1K cycles. Other than these, there are no other communication overheads.

4) Die Cost & ROI

Assuming a total of 50K gates for the test router, although the actual implementation took about 800 gates for control, in a reasonable advanced technology, this may cost about in the order of less than .01 cents per die. There are some fixed costs associated with the tests but if test time related test cost is $1 per DUT, reducing it by 133X would save about 99 cents per product. If this technique is utilized on a number of products and if the company manufactures 100M units per year, the saving would be close to $100M.

V. Limitations and Future Work

Depending on the length of the test and the quality of pin connects, there is a slight chance that the network could fail or some of the data registers that contain critical data can fail during test. Analyses need to be done on the likelihood and ways of either recovering or discovering this failure need to be devised. This could include identifying critical areas and devising ways to strengthening them [Duato 94, Gaughan 92, Kim 97]. Currently, the root selection is rather arbitrary in that we only try to avoid defect prone areas. The number of roots, location and overall use of multiple roots can be further investigated. The various topologies for the network, separation of networks for bulk of the test and the ones that need individual communication and ways of optimizing them can be another area that can be explored further. We also adopted a very simple diagnosis approach but more elaborate diagnosis flows and how they can be accommodated in this environment is also an area of further study.

MPT works on tests that do not require direct pin connection to the tester. It is possible to apply digital tests, both structural and functional through this method and many I/O tests with the aid of DFT. If there are still tests that require direct connection to the tester pins, Vih, Voh tests for example, those tests can be performed in a separate, very low cost socket. We believe this is the new area with many avenues to explore and of extremely high impact to the testing cost.

VI. Conclusion

We have introduced a new method for massively parallel testing that can be applied at both wafer probe and package test. Simulations confirmed that the planned approach works well and deals with the existing defect distributions as expected and shows very promising results.

Acknowledgment

We would like to thank Dr. Srikanth Venkataraman at Intel for providing the wafer fail map information.

References

[Aehr 04] "Aehr supplies Fuji Xerox with wafer test system", Asian EE Times March 30, 2004.

[Bhatt 09] S. Bhatt, S Kumar and K. S. Kim, "Cougarpoint Test Controller for Functional Test Reuse," Intel Design and Test Technology Conference, 2009.

[Cormen 03] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*, McGraw-Hill Science/Engineering/Math, 2003.

[Chang 91] M. Chang and W. Fuchs, "Design and parallel testing of wafer scale linear arrays with high harvest rates," *Wafer Scale Integration, 1991. Proceedings., [3rd] International Conference on*, pp. 285-291, 29-31 Jan 1991.

[Dally 04] W.J. Dally and B.P. Towles, *Principles and Practices of Interconnection Networks*, Morgan Kaufmann, 2004.

[Duato 94] J. Duato, B. Dao, P. Gaughan, and S. Yalamanchili, "Scouting: fully adaptive, deadlock-free routing in faulty pipelined networks," *Parallel and Distributed Systems, 1994. International Conference on*, pp. 608-613, 19-22 Dec 1994.

[Duato 02] J. Duato, S. Yalamanchili, and L. Ni, *Interconnection Networks*, Morgan Kaufmann, 2002.

[Eetasia 04] "Aehr supplies Fuji Xerox with wafer test system" http://www.eetasia.com/ART_8800333132_480200_NT_bafe9130.HTM

[Electroglas 09] "Electroglas: Wafer Probers: 300mm" http://www.electroglas.com/products/waferprober_300mm.shtml

[Gaughan 92] P. Gaughan and S. Yalamanchili, "Pipelined circuit-switching: a fault-tolerant variant of wormhole routing," *Parallel and Distributed Processing, 1992. Proceedings of the Fourth IEEE Symposium on*, pp. 148-155, 1-4 Dec 1992.

[FormFactor 09a]"FormFactor, Inc. Technology/Products DRAM HarmonyXP".http://www.formfactor.com/tech_prod/dram/harmony_XP.html

[FormFactor 09b] "FormFactor Reduces Flash Memory Test Cost with New Harmony OneTouch ATC," Press Release,Form Factor, www.formfactor.com/news_events/press/2009/20090209.pdf Feb. 2009

[Huang 98] K. Huang, V. Agarwal, and K. Thulasiraman, "Diagnosis of clustered faults and wafer testing," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol.17, no.2, pp.136-148, Feb 1998.

[Kim 97] J. Kim, Ziqiang Liu, and A. Chien, "Compressionless routing: a framework for adaptive and fault-tolerant routing," *Parallel and Distributed Systems, IEEE Transactions on*, vol.8, no.3, pp.229-244, Mar 1997.

[Kim 06] Kim, K. S., Marty Denham, "NoC DFX" Intel Design and Test Technology Conference, 2006.

[Mann 04] W. Mann, F. Taber, P. Seitzer, and J. Broz, "The leading edge of production wafer probe test technology," *Test Conference, 2004. Proceedings. ITC 2004. International*, 1168-1195, 26-28 Oct. 2004.

[Patil 09] S. Patil, A. Jas, E. Carrieri and K. S. Kim, "An IOSF-based Test Access Mechanism for SoCs," Intel Design and Test Technology Conference, 2009.

[Singer 07] Singer, G. http://www.itctestweek.org/files/2007singerkeynote.pdf Key Note Speech, *IEEE ,Intl Test Conference*, Oct, 2007.

[Stojmenovic 06] M. Stojmenovic and A. Nayak, "Broadcasting and routing in faulty mesh networks," *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International*, 25-29 April 2006.

[Tang 99] Q. Tang, X. Song, and Y. Wang, "Diagnosis of clustered faults for identical degree topologies," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol.18, no.8, pp.1192-1201, Aug 1999.

[Tel 09] "Wafer Probe Systems Tokyo Electron " http://www.tel.com/eng/product/wps/buwps.htm

[Tripp 05] Tripp, M., S. Picano and B. Schnarch, "Drive only at speed functional testing; one of the techniques Intel is using to control test cost," *Test Conference, 2005. Proceedings. ITC 2005. IEEE International* , Nov. 2005

[Weiss 06] M.A. Weiss, *Data Structures and Algorithm Analysis in C++*, Addison Wesley, 2006.