01 Aug 2008

Post-Configuration Testing of Asynchronous Nanowire Crossbar Architecture

Sriram Venkateswaran

Minsu Choi
Missouri University of Science and Technology, choim@mst.edu

Recommended Citation

S. Venkateswaran and M. Choi, "Post-Configuration Testing of Asynchronous Nanowire Crossbar Architecture," *Proceedings of the 8th IEEE Conference on Nanotechnology (2008, Arlington, TX)*, pp. 899-902, Institute of Electrical and Electronics Engineers (IEEE), Aug 2008.
The definitive version is available at https://doi.org/10.1109/NANO.2008.269

Post-Configuration Testing of Asynchronous Nanowire Crossbar Architecture

Sriram Venkateswaran and Minsu Choi
Dept of ECE, Missouri University of Science and Technology, Rolla, MO 65409-0040, USA
{svf44, choim}@mst.edu

***Abstract*— An asynchronous nanowire crossbar architecture has been recently proposed to eliminate the clock distribution network from conventional clocked counterpart. The proposed clock-free architecture is envisioned to enhance the manufacturability with simpler periodic structure and to improve the robustness by removing various timing-related failure modes. Even though the proposed clock-free architecture has numerous merits over its clocked counterpart, it is still not free from high defect rates induced by nondeterministic nanoscale assembly. In order to address this issue, our research team has been working on developing test schemes for effective mapping of threshold gates onto Programmable Gate Macro Blocks (PGMB). We have come up with a novel functional test approach which uses prioritized input tuples to effectively stimulate coinciding defects in configured PGMB. Numerous preliminary plots and results obtained till date prove that this scheme can be used to achieve high test efficiency for any threshold gate. The main motivation behind this research is to propose a comprehensive test scheme which can achieve high enough test coverage with acceptable test overhead. Parametric simulation results using MATLAB have been used to show potential performance of this testing scheme.**

I. Introduction

The recently proposed asynchronous nanowire crossbar architecture is based on the delay insensitive data encoding and self timed logic - therefore it is totally clock-free [1]. This helps eliminate all the failure nodes related to timing. The other potential benefits of using this architecture include enhanced manufacturability, scalability, robustness and defect and fault tolerance [2]. The proposed asynchronous nanowire crossbar architecture is based on a delay-insensitive logic paradigm known as Null Conventional Logic (NCL) [3]. NCL logic can be realized using 27 threshold gates [3]. These gates can be used to implement any expression involving upto four variables.

In the proposed architecture, every threshold gate macro that can be programmed on to a PGMB has a certain predefined pattern of crosspoint placement that would give the corresponding functionality of the gate. For instance, a TH23 gate can be expressed as $F = AB + BC + AC + AF' + BF' + CF'$, where A, B, C are the primary inputs and F’ is the output feedback. The first three product terms in this Boolean equation are for the threshold behavior of the gate since the quorum of this gate is 2. Also, the last three product terms (which is also equivalent to $(A+B+C)F'$) are for the hysteresis behavior. Once the output F is asserted, the only way to make it back to zero is reset all primary inputs. Figure 1 shows a TH23 gate configured on a PGMB.

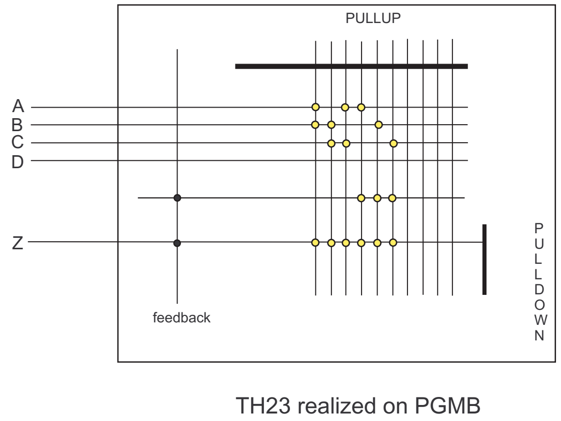

Fig. 1. TH23 gate realized on PGMB.

Defect rates arising due to fabrication vary on an average from 0% to 10% [4]. Researchers are still not able to accurately predict the defect rate in these PGMBs. The effects of these defects on the logical operation of the circuit needs to be scrutinized. These defects have to be tolerated to maintain proper functionality of the circuit.

II. Functional Test Approach

The most primitive way of testing a nanowire crossbar is to test individual crosspoints one by one by sequentially scanning through them and generate a defect map. This is not only a very laborious scheme, but also introduces a considerable amount of testing overhead in time/space complexity [2]. The functional test scheme proposed in our paper is designed to test maximum number of programmable crosspoints using the minimal number of test inputs. The test inputs are nothing but logical inputs based on the logical expression realized by any THmn gate. As shown in the algorithm, the first step is to map the THmn gate onto the PGMB following which the truth table for the specific gate is generated. A list of prioritized inputs are generated for testing the ON crosspoints. In case our objective is to scan the PGMS for defects, then inputs are applied in order of decreasing priority. In this manner, the entire ON programmable space is successfully scanned. In case locating the defect is essential, then partial isolation and location can be achieved. This is however confined only to the OR plane crosspoints. The reason being they have direct correspondence with the test tuples.The fault count thus

978-1-4244-2104-6/08/$25.00 ©2008 IEEE.

generated from either of the approaches specifies the number of defective crosspoints generated. Another feature of our approach is that the functional test scheme being proposed in this work avoids the issues associated with this raw crossbar testing. The crosspoints under test are limited by the number of ON-inputs (i.e., crosspoints that should be programmed as ON) of the given threshold gate macro. Minimizing the test space helps reduce the test time. In addition, since Boolean inputs are used to check for defects, these programmable inputs can be prioritized according to the number of ON-inputs they can cover. The other advantage of this approach is the minimal number of test inputs it takes to cover the test space. On close comparison of desired functional output due to defect free mapping and one generated due to defective crosspoints at programmable locations, prioritized input tuple levels have been set for each threshold gate. These prioritized test tuples can be applied sequentially to validate the programmed gate function.

Test Tuple	# of programmable locations tested
1 Test Tuple	3 crosspoints
2 Test Tuples	6 crosspoints
3 Test Tuples	9 crosspoints
4 Test Tuples	12 crosspoints
5 Test Tuples	13 crosspoints
6 Test Tuples	14 crosspoints
7 Test Tuples	15 crosspoints
8 Test Tuples	16 crosspoints
9 Test Tuples	17 crosspoints
10 Test Tuples	18 crosspoints

TABLE I

TEST TUPLES AND THEIR CORRESPONDING NUMBER OF TESTPOINTS FOR TH23 GATE

		Prioritized Test Tuple Count				
Gate	N_{on}	3	4	5	6	7
TH23	18	50%	66.67%	72.22%	77.8%	83.3%
TH24	30	40%	53.33%	66.67%	70.0%	73.3%
TH34	28	28.57%	35.71%	42,86%	50.0%	57.17%
TH33w2	15	53.3%	66.67%	73.3%	80.0%	86.67%
TH44w3	21	47.6%	57.14%	66.67%	71.42%	76.19%

TABLE II

TOTAL CROSSPOINTS TESTED VS. PRIORITIZED TEST TUPLE COUNT.

Let's consider TH23 gate. The three primary inputs will generate 8 input bit patterns ranging from 000 to 111. Figure 1 shows a TH23 gate configured on a PGMB. We can see that there are 18 ON-inputs represented by highlighted dots. Imperfect assembly may cause any one or more of these points to be OFF. For example, a defect at the left-most crosspoint in the first row results in a faulty function of $F^* = B + BC + AC + AF' + BF' + CF'$. Notably, one or more test input tuples can be found by comparing output columns of F and F^* in their truth table. The proposed functional test scheme also applies input tuples in the order of their priority level and validates outputs from those input tuples. As the number of applied test tuples increases, the total number of testable ON-input crosspoints increases, too. Figure I shows the number of testable ON-input crosspoints as a function of test tuple count. The first sets of 4 inputs test 12 out of the 18 possible programmable locations for defects. In case a particular test input results in an undesired output, then the ON-crosspoints under test are tested as bad. In case of the TH23 gate, using the first 3 most highly ranked input tuples cover 50% of the total test space. Using another input increases this to 66.67% . This rate rises to 72.2, 77.8, 83.3% respectively with each additional input. Table II shows the coverage values for all 5 gates under consideration. This is a very important point especially when we have a large input sample space. For example, in order to test 75% of ON-crosspoints, 6 input tuples should be applied. With this set level, we can achieve a relative testability (i.e., # of total tested-good crosspoints/# of total crosspoints tested) of greater than 90% on average. Figure 2 shows a plot of relative testability for 5 different threshold gates. These 5 threshold gates have been considered in the following plots because they cover the maximum possible input combinations and can be considered as representatives of the several other types of gates. The TH23 gate has only two priority levels as shown in fig. 3. TH34, on the other hand, has 3 input priority levels with the first tuple testing 4 ON-crosspoints, the second highest set testing 2 points each and finally the lowest level providing one to one correspondence.

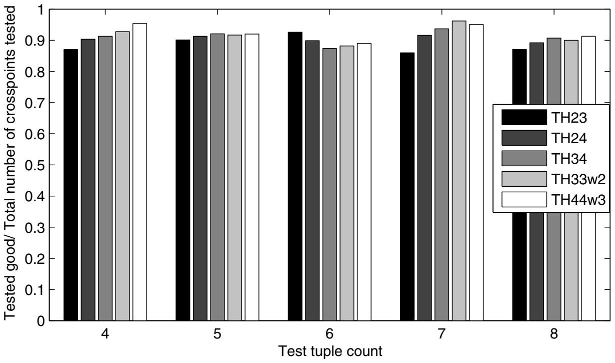

Fig. 2. Relative testability of THmn gates

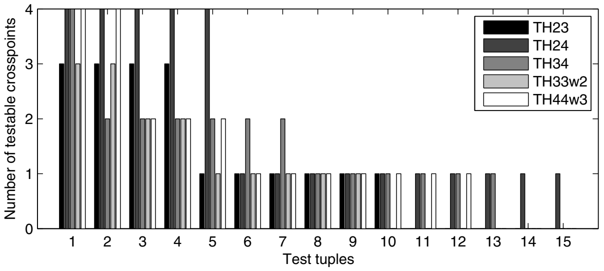

Fig. 3. Testable crosspoints with each input for THmn Gates

Consider Figure 3 which gives the tested good over the tested bad PGMB ratio. This plot helps us understand the relative distribution of the two types of PGMB in the sample. The nature of the plots show that as the defect rate decreases and as the number of crosspoints under test increases, the ratio of tested good over tested bad falls considerably. This count is of extreme significance especially when we require the distribution of bad crosspoints for the purpose of repair. In case of repair being the priority, the inputs are applied in the order of increasing priority. This will enable maximum one to one correspondence to be achieved. In the set of programmable crosspoints, the OR plane has highest priority. In order to account for any potential failure in any programmable OR crosspoint, we have proposed a unique solution. Our solution suggests implementing OR plane redundancy. A parallel OR plane can be introduced. Fig 4 represents the distribution of bad PGMBs due to at least one defect in any of the programmable OR crosspoint locations. TH24 gate has the highest number of defective PGMBs since it uses all the 10 programmable crosspoints. TH33w2 on the other hand has only 5 out of the available 10 which are programmed. This concentration of defects over a single OR plane especially for higher defect rates suggests the need to focus on the OR plane. For minimizing the defective PGMBs due to defective OR plane, we need to test this plane by using low priority inputs. This can help locate the defects which can be repaired or corrected accordingly in future.

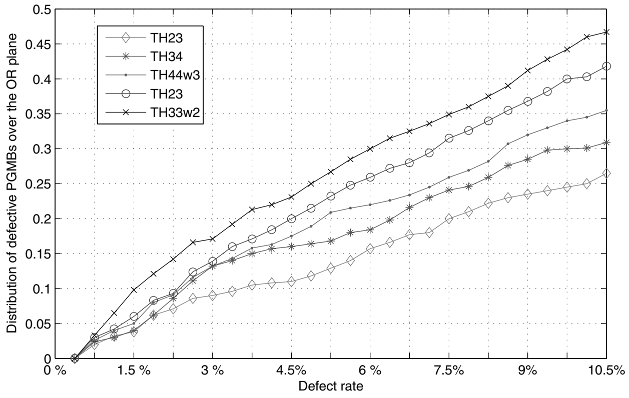

Fig. 4. distribution of defective PGMBs due to defective OR plane crosspoints

Consider a 6x10 grid and a TH24 gate is to be implemented on this. We have 10 programmable OR locations in this gate. By introducing a redundant wire we increase the PGMB dimensions to 7x10. In case the OR crosspoint of the j_{th} row is defective, we can program the crosspoint on the j-1 th row and corresponding to the same column number. Only if both the points are defective simultaneously will there be a manipulation in the desired output. In case one of them is defective, we can still achieve efficient programmability with this approach.

The plots and results have been obtained considering the defect rate of 10%, which is the worst case under the current prediction.

Accuracy is a figure of merit which has been used to quantify our test approach. Accuracy of the functional test scheme can be defined as the ratio of number of tested as bad PGMBs over the total number of bad PGMBs. It is evident that the accuracy ratio increases with increase in defect rate and the number of test tuples covered. For lower defect rates and lesser number of test tuples, the number of bad crosspoints are few. Of the two dependent parameters, only the number of test tuples applied can be varied. Hence, test tuple count should be suitably selected with due consideration to required accuracy. In fig.6 and fig.7, accuracy plots for TH23 gate with varying number of test tuples and increasing defect rates have been generated. It is interesting to note that in both the plots, the accuracy rates increase with defect rate. This is due to the increase in total number of bad PGMBs with increase in defect rate. When the prioritized inputs are applied in reverse order, the accuracy is very low and increases slowly with each test tuple. For test tuples applied in order of decreasing priority, we can achieve higher accuracy for comparatively lesser number of tuples applied. Having said that, if location of defect is essential, then a compromise needs to be made on the accuracy front. This is a necessary tradeoff. Another complementary factor that can be generated is escape factor. It is the ratio of actually bad PGMBs over total identified bad PGMBs. Actual bad ones are those which have been subject to all the test tuples possible to cover the entire programmable space. Total identified bad PGMBs are those which have been identified as bad when a reduced set of test tuples have been applied. This reduced set, called as N_{test} is a subset of the total test points, denoted by N_{on}. It is clear from definition that accuracy and escape factor are complementary to each other. Escape factor is greater when lesser number of test tuples are applied.For increasingly larger number of test tuples, the number of indeed bad PGMBs are lesser, bringing down the escape factor. A low value for escape factor means lesser the chances of an indeed bad PGMB escaping as a tested good one.

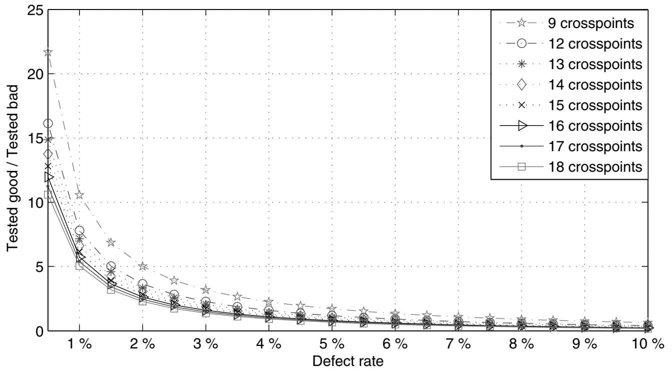

Fig. 5. Tested good over tested bad PGMB ratio for varying defect rates and variation in number of crosspints

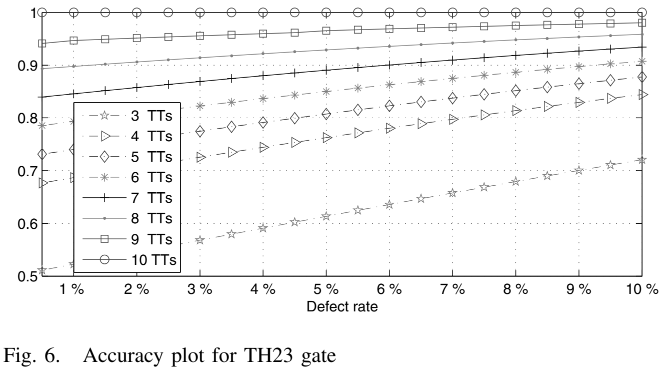

Fig. 6. Accuracy plot for TH23 gate

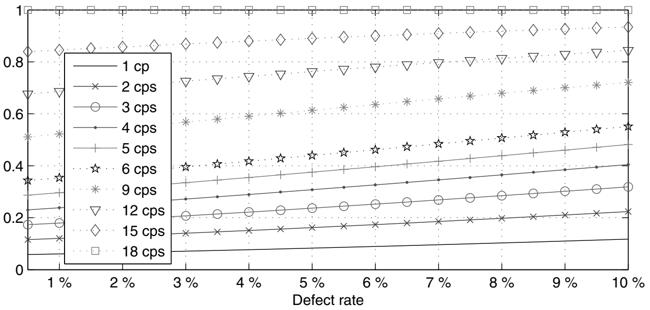

Fig. 7. Accuracy plot for TH23 with test tuples applied in order of increasing priority

III. Conclusion and Future work

The complete sequential scan testing of nanowire crossbar guarantees the perfect test coverage. However, this scheme is rather laborious in terms of time/space complexity. Thus, we have proposed a novel test approach for the recently proposed asynchronous nanowire crossbar architecture. The proposed testing scheme is to functionally test ON-crosspoints solely by applying a number of input tuples. Notably, some of the input tuples may be used to cover more than one ON-crosspoint. Thus, it is possible to prioritize them to achieve the desired combination of test coverage and overhead. The trade-off between the performance (i.e., test coverage) and the overhead (i.e., # of total input tuples applied) is shown in preliminary simulation results in this paper. Having said that, in case of locating defects being our priority, we lose one-to-one correspondence with the input tuples with increasing priority. We will hence no longer be able to directly isolate AND plane defects. We will have to use combination of inputs to locate faults. In future , we plan to extend our functional test algorithm to accommodate this. All these approaches are aimed at maximizing the utility of PGMBs in spite of the inherent fabrication defects.

References

[1] R. Bonam, S. Chaudhary, Y. Yellambalase and M. Choi, "Clock-Free Nanowire Crossbar Architecture based on Null Convention Logic (NCL)", 7th IEEE International Conference on Nanotechnology (IEEE-Nano), Apr 2007.

[2] Ravi Bonam, Yong-Bin Kim, Minsu Choi, "Defect-Tolerant Gate Macro Mapping and Placement in Clock-Free Nanowire Crossbar Architecture", 22nd IEEE International Symposium on Defect and Fault-Tolerance in VLSI Systems, 2007. DFT '07. 26-28 Sept. 2007 Page(s):161 - 169

[3] S. C. Smith, R. F. DeMara, J. S. Yuan, D. Ferguson, and D. Lamb, " Optimization of NULL Convention Self-Timed Circuits", "Integration, The VLSI Journal, Vol. 37, No. 3, pp. 135-165, 2004.

[4] J.Huang, M.B. Tahoori and F. Lombardi, " On the defect tolerance of Nano-Scale Two Dimensional Crossbars"' IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, pp 96-104,Oct 2004.