Defect-aware Logic Mapping for Nanowire-based Programmable Logic Arrays via Satisfiability

Yexin Zheng and Chao Huang
Bradley Department of Electrical and Computer Engineering
Virginia Tech, Blacksburg, VA 24061, USA
{yexin, chaoh}@vt.edu

***Abstract*—Programmable logic arrays (PLAs) using self-assembly nanowire crossbars have shown promising potential for future nano-scale circuit design. However, due to the density and size factors of nanowires and molecular switches, the fabrication fault densities are much higher than those of the conventional silicon technology, and hence pose greater design challenges. In this paper, we propose a novel defect-aware logic mapping framework via Boolean satisfiability (SAT). Compared with the prior works, our technique considers PLA defects on both input and output planes at the same time. This synergistic approach can help solve logic mapping problems with higher defect rates. The proposed method is universally suitable for various nano-scale PLAs, including AND/OR, NOR/NOR structures, *etc*. The experimental results have shown that it can efficiently solve large mapping problems at a total defect rate of 20% or even higher. We further investigate the impact of different defects on PLA mapping, which helps set up an initial contribution for yield estimation and utilization of partially-defective PLAs.**

I. Introduction

Novel nano-scale electronic devices have been proposed to enhance or possibly replace the conventional complementary metal-oxide semiconductor (CMOS) technology for future circuit system design [1], [2]. Given these emerging opportunities, the traditional top-down fabrication process becomes difficult and costly to deal with such increasingly size-shrinking nano-devices. The alternative approach, bottom-up self-assembly process, demonstrates promising potential to fabricate nano-electronic circuits more economically and precisely [3], [4]. The lithography-independent self-assembled process features fabrication regularity, which is well suited to implement reconfigurable structures [5]. Architectures such as nanowire-based programmable logic array (PLA) thus have become active research topics most recently [6], [7], [8].

Due to the density and size factors of nanowires and molecular switches, however, the PLA circuit defects are inevitable during the non-deterministic self-assembly process. The defect rates could be relatively high – about 15% defective crosspoints of using molecular switches have been observed in a recently fabricated 8×8 crossbar [9]. This figure is orders of magnitude larger than conventional silicon technologies. Therefore, advanced fault-tolerant design methods are essential to fully take advantage of emerging nano-technologies.

For defective molecular PLAs, the malfunctioning crosspoints and broken nanowires impose a great deal of topological constraints in logic synthesis. Therefore, mapping logic functions onto a defective PLA is nontrivial and can be difficult to find a feasible solution if the defect rate is high. In [10], a greedy algorithm of bipartite matching is proposed to map logics around crosspoint defects for nanoPLA structures, which assumes that the PLA inputs have been previously assigned. The work in [11] models the PLA synthesis problem as embedding a logic function bipartite graph into a crossbar bipartite graph, and develops heuristics to help prune impossible mappings. A defect-unaware method is presented in [12] through identifying defect-free subsets in a defective crossbar. In this paper, we present a framework for defect-aware logic mapping on nanowire-based PLAs via satisfiability (SAT). Similar to [10], [11], our method is on a per-array basis.

SAT is the problem of deciding if there exists an assignment for the variables in a propositional formula that makes the formula true. These problems are usually formulated in the conjunctive normal form (CNF), which consists of the conjunction (logical AND) of several clauses and each clause is a disjunction (logical OR) of one or more literals. SAT-based methods have been widely used to solve complex problems in electronic design automation, including combinational equivalence checking [13], model checking [14], routing [15], *etc*. Thanks to the achievements of current SAT solvers in terms of efficiency and scalability [16], [17], [18], it is beneficial to employ SAT-based methods in the context of emerging nano-technology design.

We have developed techniques to efficiently formulate PLA logic mapping into Boolean CNF formulas. The PLA defects are integrated as covering and closure constraints, including switch stuck-open fault, switch stuck-closed fault, nanowire broken fault, and other faults that result in unusable nanowires. Compared with the prior works on defect-tolerant mapping which tackles a single crossbar each time [11], [12], our technique considers defects on both input (AND) and output (OR) planes at the same time, and generates mapping on these two crossbars synergistically. This comprehensive approach can help to solve mapping problems with higher defect rates. This work is also among the first to consider assignments on broken nanowires and stuck-closed switches, which are treated as completely defective/unusable in the prior works. However, the proposed method does not limit itself to the AND/OR PLA structure, but is inherently suitable for other nano-scale structures, such as the NOR/NOR PLAs [6]. We further investigate the impact of different defects on PLA mapping, which helps set up an initial contribution for yield estimation and utilization of defective PLAs.

978-3-9810801-5-5/DATE09 © 2009 EDAA

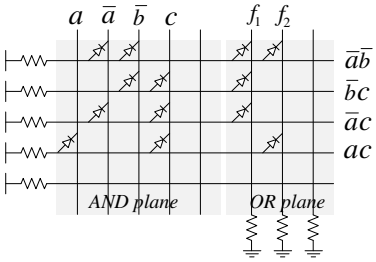
The rest of this paper is organized as follows. Section II introduces the background materials of the PLA architecture and defect models. Section III describes our defect-aware mapping methodology in detail. Section IV demonstrates the experimental results. Finally Section V concludes.

II. Background

In this section, we describe the preliminary concepts on molecular PLA and defect models.

A. Nanowire-based PLA

The nanowire-based PLA architecture [19], [20] mainly uses molecular crossbar as building blocks. Generally speaking, a crossbar contains two groups of parallel nanowires, which are perpendicular to each other, and molecules at the intersections (crosspoints) as programmable switches. By using pullup and/or pulldown resistors to configure the crosspoint switches, the crossbars can implement logic AND and OR functions.



$$f_1 = \bar{a}\bar{b} + \bar{b}c + \bar{a}c$$
$$f_2 = \bar{a}\bar{b} + ac$$

Fig. 1. Logic function mapping on defect-free PLA

A nanowire-based PLA circuit is shown in Fig. 1. It is an AND/OR PLA implementing Boolean functions in the form of sum of products. The AND plane selectively generates the product terms of the desired Boolean functions, while the OR plane chooses to sum the corresponding product terms, thereby, to result in the circuit outputs. This selection process, *i.e.*, the PLA mapping process, is accomplished by programming the nanodevices at the crosspoints. Mapping an arbitrary set of Boolean functions on a defect-free PLA is straightforward. Fig. 1 demonstrates an example of realizing functions $f_1 = \bar{a}\bar{b} + \bar{b}c + \bar{a}c$ and $f_2 = \bar{a}\bar{b} + ac$.

B. Defect model

Due to the density and small size factors of nanowire and molecular switch, the circuit defects become unfortuantely inevitable, which pose great challenges to designers. In general, the molecular PLA defects are listed as follows.

- *Switch stuck-open fault:* The switch connecting two perpendicular nanowires is stuck-open – the switch is preprogrammed as "OFF". Although the configurability of this specific junction is lost, two related nanowires can still be used as if the faulty switch is left open during the mapping. A stuck-open example, shown with sign $\times$ in Fig. 2(a), represents the faulty switch at the crosspoint of horizontal nanowire h_2 and veritical nanowire v_1. We can still map a product term and a variable to nanowires h_2 and v_1, respectively, if the variable mapped to v_1 does not belong to the product term mapped to h_1.
- *Switch stuck-closed fault:* The switch connecting two perpendicular nanowires is stuck-closed – the switch is preprogrammed as "ON". In this case, we can either generate a mapping with this switch turning on, or use the vertical (horizontal) nanowire and leave the horizontal (vertical) nanowire unused in AND (OR) plane. For the switch stuck-closed fault (sign $\bullet$) in Fig. 2(a) at the crosspoint of v_3 and h_3, we can either leave h_3 unused, or map a variable to v_3 and a product term containing the variable to h_3.
- *Nanowire broken fault:* A nanowire in PLA is broken into two or more segments. In this situation, the segment that connects to the PLA input/output can still be used, such as the crosspoint at the intersection of the broken nanowire v_7 and horizontal nanowire h_1 shown in Fig. 2(a).
- *Other faults that result in unusable nanowires:* Since a complete unusable nanowire can be removed from the PLA model, these faults are not considered in the mapping algorithm.

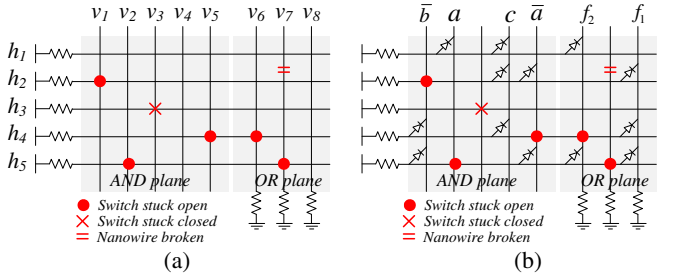


Fig. 2. Defective PLA: (a) defect model, and (b) defect-aware mapping

Fig. 2(b) illustrates a possible mapping of the same functions as seen in Fig. 1 on the defective PLA given in Fig. 2(a), where various forementioned defects are presented.

III. Defect-aware PLA Mapping

We next present the SAT-based defect-aware logic mapping in detail. We will address the method to integrate various PLA defects as constraints. A brief discussion of computation complexity is also provided.

A. Problem formulation

A nanowire-based PLA architecture (Fig. 1), with an $|H| \times |V_a|$ crossbar for the AND plane and an $|H| \times |V_o|$ crossbar for the OR plane, is a 3-tuple $C(H, V_a, V_o)$, where set H represents the set of horizontal nanowires and sets V_a and V_o represent the set of vertical nanowires in the AND and OR plane, respectively. A set of logic functions can also be expressed as a 3-tuple $F(I, P, O)$, where sets I, P, O represent the set of input variables, product terms, and outputs, respectively. Given a PLA $C(H, V_a, V_o)$ and a function set $F(I, P, O)$, the PLA mapping problem can be defined as finding a mapping between C and F such that there exist injective relation $g_1 : I \rightarrow V_a$, $g_2 : P \rightarrow H$, and $g_3 : O \rightarrow V_o$.

B. SAT-based defect-free mapping

In order to model the PLA mapping as a satisfiability problem, we encode the mapping by introducing Boolean variables and formulating the constaints into SAT clauses. For the given input variables I and vertical nanowires V_a of the AND plane, we use $|I| \times |V_a|$ Boolean variables $X_I^{V_a}$=$\{x_i^v | i \in I, v \in V_a\}$ to represent the injective mapping $g_1 : I \rightarrow V_a$. If the i^{th} input variable is mapped to the v^{th} vertical nanowire, variable x_i^v is true, otherwise false. Similarly, we introduce $|P| \times |H|$ Boolean variables Y_P^H=$\{y_p^h | p \in P, h \in H\}$ to encode product set to horizontal nanowire set mapping $g_2 : P \rightarrow H$, and $|O| \times |V_o|$ Boolean variables $Z_O^{V_o}$=$\{z_o^v | o \in O, v \in V_o\}$ for $g_3 : O \rightarrow V_o$ mapping.

To encode the constraints, let us consider the requirements for a feasible mapping solution. The requirements can be categorized as two types: covering constraints and closure constraints. The convering constraints ensure that each input, product term, or output is mapped to at least one nanowire. The closure constraints ensure that no input, product term, or output of the function and no nanowire is mapped more than once. For the injective mapping $g_1 : I \rightarrow V_a$, the constraints can be summarized as follows.

Covering constraints: Each input must be assigned to at least one vertical nanowire:

$$\bigwedge_{i \in I} \left(\bigvee_{v \in V_a} x_i^v \right)$$

Closure constraints: The first closure constraint is that each input must be assigned at most one vertical nanowire. In other words, for each pair of variables $x_i^{v_1}$ and $x_i^{v_2}$, at least one is assigned to zero (false).

$$\bigwedge_{i \in I} \left(\bigwedge_{v_1, v_2 \in V_a}^{v_1 \neq v_2} (\neg x_i^{v_1} \vee \neg x_i^{v_2}) \right)$$

Another closure constaint ensures that at most one input is assigned to each nanowire. In other words, for variable pair $x_{i_1}^v$ and $x_{i_2}^v$, at least one is assigned to zero.

$$\bigwedge_{v \in V_a} \left(\bigwedge_{i_1, i_2 \in I}^{i_1 \neq i_2} (\neg x_{i_1}^v \vee \neg x_{i_2}^v) \right)$$

Similarly, the constraints for $g_2 : P \rightarrow H$ are:

$$\bigwedge_{p \in P} \left(\bigvee_{h \in H} y_p^h \right) \wedge \bigwedge_{p \in P} \left(\bigwedge_{h_1, h_2 \in H}^{h_1 \neq h_2} (\neg y_p^{h_1} \vee \neg y_p^{h_2}) \right)$$

$$\wedge \bigwedge_{h \in H} \left(\bigwedge_{p_1, p_2 \in P}^{p_1 \neq p_2} (\neg y_{p_1}^h \vee \neg y_{p_2}^h) \right)$$

The constraints for $g_3 : O \rightarrow V_o$ are:

$$\bigwedge_{o \in O} \left(\bigvee_{v \in V_a} z_o^v \right) \wedge \bigwedge_{o \in O} \left(\bigwedge_{v_1, v_2 \in V_o}^{v_1 \neq v_2} (\neg z_o^{v_1} \vee \neg z_o^{v_2}) \right)$$

$$\wedge \bigwedge_{v \in V_o} \left(\bigwedge_{o_1, o_2 \in O}^{o_1 \neq o_2} (\neg z_{o_1}^v \vee \neg z_{o_2}^v) \right)$$

We constuct a satisfiability formula by conjucting all the above constarints. The assignment that satisfies all the clauses is a possible assignment result for mapping function set $F(I, P, O)$ on a defect-free PLA $C(H, V_a, V_o)$.

C. Defect-aware constraints

To take into account the PLA defects discussed in Section II-B, we formulate all these defects as satisfiability constraints such that the solution for these constaints in conjunction with the formula of the defect-free assignment is a feasible defect-aware mapping result.

Switch stuck-open fault: If the switch is stuck at open at the crosspoint of vertical nanowire v and horizontal nanowire h on the AND plane, we cannot map either input i to nanowire v or product term p to nanowire h, given that i is a literal contained in p.

$$\bigwedge_{p \in P} \left(\bigwedge_{i \in p} (\neg x_i^v \vee \neg y_p^h) \right)$$

If such fault presents on the OR plane, we cannot map either ouput o to nanowire v or product p to nanowire h, if h is a product term of output o.

$$\bigwedge_{o \in O} \left(\bigwedge_{p \subseteq o} (\neg y_p^h \vee \neg z_o^v) \right)$$

Switch stuck-closed fault: We use AND plane as an example. If a switch is stuck at closed at the crosspoint of vertical nanowire v and horizontal nanowire h, and if a product p is map to nanowire h, we can either map an input belonging to p or a logic 1 to nanowire v. Therefore, we introduce a number of $|V_a|$ addition variables here to represent mapping logic 1 to vertical nanowires: $x_{one}^v (v \in V_a)$. Then the corresponding constraints for the stuck-closed fault are formulated as:

$$\bigwedge_{p \in P} \left(\neg y_p^h \vee x_{one}^v \vee \bigvee_{i \in p} x_i^v \right)$$

If such defect presents on the OR plane, and if an output f is map to nanowire v, a product of output o or a logic 1 must be mapped to nanowire h. Similarly a group of new variables $y_{one}^h (h \in H)$ are introduced to formulate the constaints as:

$$\bigwedge_{o \in O} \left(\neg z_o^v \vee y_{one}^h \vee \bigvee_{p \subseteq o} y_p^h \right)$$

Due to the introduction of new variables, proper changes are made on closure constraints to guarantee that no other variables and logic 1 are mapped to the same nanowire.

Nanowire broken fault: If a horizontal nanowire is broken, we treat the whole nanowire as defective. For a broken vertical nanowire, the only segement that can be used are the ones that connect to the PLA input/output. We define the horizontal nanowires that intersect the defective verical nanowire v at the unusable crosspoints as nanowire set B_v. For example, for the broken nanowire v_7 in the OR plane in Fig. 2(a), the horizontal nanowires other than h_1 are in set B_v. Therefore, if an input/output is mapped to the broken nanowire v, its related product terms cannot be assigned on the horizontal nanowires

in set B_v. We thus derive the following constraints for a broken nanowire in the AND and OR plane, respectively.

$$\bigwedge_{i\in I}\left(\bigwedge_{h\in B_v}\left(\bigwedge_{p\ni i}(\neg x_i^v \vee \neg y_p^h)\right)\right)$$

$$\bigwedge_{o\in O}\left(\bigwedge_{h\in B_v}\left(\bigwedge_{p\subseteq o}(\neg z_o^v \vee \neg y_p^h)\right)\right)$$

D. Compuation complexity

The computation complexity of SAT solving is related to the size of the SAT instance, specifically, the number of variables and clauses. Given a PLA of $N \times N$ crossbars, the number of variables and clauses of the SAT formula for a defect-free mapping is respectively $O(kN)$ and $O(kN^2)$, where $k = |I|+|P|+|O|$. Due to our efficient problem formulation, the defect-aware constraints do not increase the order of both variable and clause numbers. Thanks to the achievements of the current SAT solvers, a feasible solution can be generated in a reasonable time for N up to several hundred.

IV. Experimental Results

We evaluate our proposed defect-aware PLA mapping methodology by using PLA benchmarks from the *LGSynth93* benchmark set [21]. We compare the effects when different PLA defect rates are present. *Berkmin561* [18] SAT solver is employed to solve the formulated CNF instances. The experiments are performed on an *Intel Xeon* $3GHz$ workstation with $2GB$ memory running *Linux* operation system.

Table I summarizes the experimental results of solving the CNF instances generated for mapping problems at different defect rates. P_o, P_c, and P_b represents the defect rate of switch stuck-open fault, switch stuck-closed fault, and broken nanowire fault, respectively. We randomly generate these three types of faults with a ratio 3:1:1, since the switch stuck-open fault is the most common. The size of the PLA structure is related to the size of the benchmark circuit. For the experiment results demonstrated in Table I, the PLA size is $1.5X$ the size of the benchmark circuit, in other words, $|V_a| = 1.5 \times |I|$, $|V_o| = 1.5\times|O|$, and $|H| = 1.5\times|P|$. The number of variables and clauses for each CNF instance is given under columns *Var* and *Cls*, respectively. Since the numbers of variables for defective-aware mapping problems are the same, the variable numbers are listed once. Column *SAT* lists whether the SAT solver can successfully find a solution within a time limit of 1200 seconds. The actual run time is shown under column *time*.

Demonstrated in Table I, the proposed defect-aware PLA mapping method can find a feasible mapping efficiently even at a high defect rate. Compared with the work in [11], our method can be used for larger circuits with more defects yet with great improvements in the solving performance. The SAT solving time of defect-aware PLA mapping is comparable to the defect-free mapping for most of the cases at a total defect rate of both 10% and 15%. As the defect rate increases, the constraints for a feasible mapping become large. Finding a mapping solution or proving that no solution exists may require longer computation time, possibly exceeding the time limit. However, the proposed method has demonstrated its capability to efficiently solve PLA mapping problems for large circuits, such as *clip* (with 167 product terms), at a total defect rate of 20%.

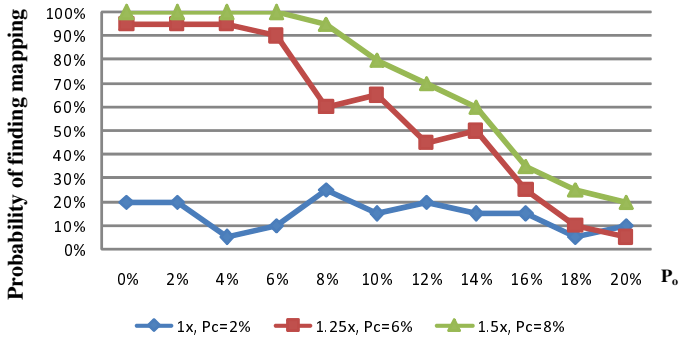


Fig. 3. Circuit *5xp1*: probability of mapping at different PLA sizes and defect rates

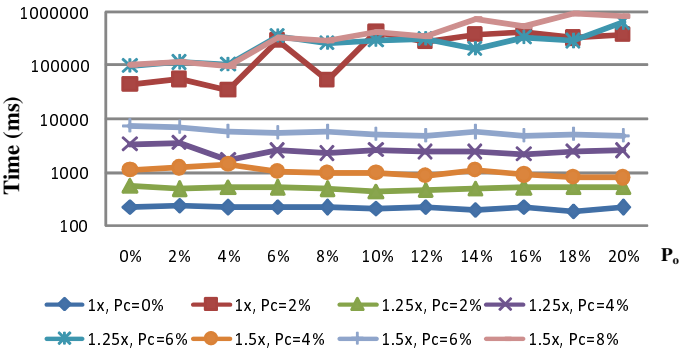


Fig. 4. Circuit *5xp1*: average SAT solving time at different PLA sizes and defect rates

We further provide experimental results to analyze the impact of different types of defects on defect-aware PLA mapping. We use benchmark circuit *5xp1* as an example and other benchmarks have shown a similar trend. Fig. 3 illustrates the probability of finding a mapping solution at different defect rates. For each defect rate, 20 defective PLA structures are randomly generated. The PLA size is also proportional to the benchmark circuit size. For example, the $1.5X$ in the legend represents $|V_a| = 1.5 \times |I|$, $|V_o| = 1.5 \times |O|$, and $|H| = 1.5\times|P|$. For those defective PLAs with P_c less than the value shown in Fig. 3, we can always find a feasible solution. Fig. 4 gives the corresponding average SAT solving time.

As expected, the PLAs with higher defect rates require longer SAT solving time and lead to a lower probability of finding a feasible solution. Especially when the defect rates are high enough to limit the feasible solutions, the solving time grows exponentially. Compared with the switch stuck-open fault, the switch stuck-closed fault contributes more in limiting performance. We can observe from Figs. 3 and 4 that the solving performance degrades more if increasing P_c by 2% than increasing P_o by 2%. The reason is because a switch stuck-closed defect may affect the use of a complete

TABLE I
SAT SOLVING COMPARISONS AT DIFFERENT DEFECT RATES

Circuit	$\|I\|/\|O\|/\|P\|$	Defect-free			P_o=6%, P_c=2%, P_b=2%				P_o=9%, P_c=3%, P_b=3%			P_o=12%, P_c=4%, P_b=4%		
		Var	Cls	Time(s)	Var	Cls	Time(s)	SAT	Cls	Time(s)	SAT	Cls	Time(s)	SAT
rd53	10/3/32	1701	61719	0.017	1764	70505	0.046	Y	73986	0.039	Y	77608	0.042	Y
inc	14/9/34	2154	78192	0.018	2226	97314	0.054	Y	105977	0.063	Y	114507	0.056	Y
misex2	50/18/29	5512	286469	0.068	5631	333131	0.146	Y	353956	0.483	Y	374828	111.2	Y
sao2	20/4/58	5670	375367	0.087	5787	452475	0.176	Y	488167	0.214	Y	524370	0.312	Y
bw	10/28/65	7696	554597	0.131	7809	614708	0.331	Y	641679	0.724	Y	668442	16.57	Y
5xp1	14/10/75	8919	794850	0.179	9053	857081	0.411	Y	883892	0.488	Y	910628	0.618	Y
9sym	18/1/87	11885	1241432	0.297	12043	1371379	0.738	Y	1430541	0.630	Y	1489626	0.506	Y
rd73	14/3/141	30201	5251100	1.463	30434	5527061	2.219	Y	5649527	2.288	Y	5772977	2.456	Y
table5	34/15/158	39525	7436517	2.120	39813	9088189	188.8	Y	9893247	>1200	NA	10700839	>1200	NA
clip	18/5/167	42443	8729595	2.380	42721	9175331	4.107	Y	9377051	6.582	Y	9579688	108.5	Y

horizontal nanowire, while a switch stuck-open defect only affects a single crosspoint.

The probability of a feasible mapping under a given defective PLA is directly related not only to the defect rate but also to the PLA size. For the same defect rate, the increase of PLA size can improve the probability of successful mapping. As demonstrated in Fig. 4, for a total defect rate of 15% on crosspoints under the current fabrication technology [9], a 50% redundancy of PLA size is sufficient.

These relations among defect rate, PLA size, and solving performance demonstrated by our experimental results provide a guidance for yield estimation and improvement for future nano-scale PLA implementations. Given a defect rate, a properly selected PLA size can help to achieve a high yield within a reasonable design time.

V. CONCLUSIONS

In this paper, we present a SAT-based framework for defect-aware logic mapping on nanowire-based PLAs. The proposed method formulates PLA logic mapping into Boolean CNF formulas, and the PLA defects as covering and closure constraints. The experimental results have demonstrated that it can efficiently solve large PLA mapping problems at high defect rates. The proposed method does not only limit itself to the AND/OR PLA structure, but is also inherently suitable for other newly-developed nano-scale PLA structures. Furthermore, the impact and analysis of different defects on PLA mapping is provided, which helps set up an initial contribution for yield estimation and utilization of defective PLAs.

REFERENCES

[1] A. DeHon and K. K. Likharev, "Hybrid CMOS/nanoelectronic digital circuits: Devices, architectures, and design automation," in *Proc. Int. Conf. Computer-Aided Design*, Nov. 2005, pp. 375–382.

[2] S. K. Shukla and R. I. Bahar, *Nano, Quantum and Molecular Computing: Implications to High Level Design and Validation*, Kluwer Academic Publishers, Boston, MA, 2004.

[3] Y. Huang, X. Duan, Q. Wei, and C. M. Lieber, "Directed assembly of one-dimensional nanostructures into functional networks," *Science*, vol. 291, no. 5504, pp. 630–633, Jan. 2001.

[4] V. V. Zhirnov and D. J. C. Herr, "New frontiers: Self-assembly and nanoelectronics," *IEEE Computer*, vol. 34, no. 1, pp. 34–43, Jan. 2001.

[5] J. R. Heath, P. J. Kuekes, G. S. Snider, and R. S. Williams, "A defect-tolerant computer architecture: Opportunities for nanotechnology," *Science*, vol. 280, no. 5370, pp. 1716–1721, June 1998.

[6] A. DeHon and M. J. Wilson, "Nanowire-based sublithographic programmable logic arrays," in *Proc. Int. Symp. Field-Programmable Gate Arrays*, Feb. 2004, pp. 123–132.

[7] S. C. Goldstein and M. Budiu, "NanoFabric: Spatial computing using molecular electronics," in *Proc. Int. Symp. Computer Architecture*, June 2001, pp. 178–189.

[8] G. S. Rose and M. R. Stan, "A programmable majority logic array using molecular scale electronics," *IEEE Trans. Circuits & Systems I*, vol. 54, no. 11, pp. 2380–2390, Nov. 2007.

[9] Y. Chen, G.-Y. Jung, D. A. A. Ohlberg, X. Li, D. R. Stewart, J. O. Jeppesen, K. A. Nielsen, J. F. Stoddart, and R. S. Williams, "Nanoscale molecular-switch crossbar circuits," *Nanotechnology*, vol. 14, no. 4, pp. 462–468, Apr. 2003.

[10] H. Naeimi and A. DeHon, "A greedy algorithm for tolerating defective crosspoints in NanoPLA design," in *Proc. Int. Conf. Field-Programmable Technology*, Dec. 2004, pp. 49–56.

[11] W. Rao, A. Orailoglu, and R. Karri, "Topology aware mapping of logic functions onto nanowire-based crossbar architectures," in *Proc. Design Automation Conf.*, July 2006, pp. 723–726.

[12] M. B. Tahoori, "A mapping algorithm for defect-tolerance of reconfigurable nano-architectures," in *Proc. Int. Conf. Computer-Aided Design*, Nov. 2005, pp. 667–671.

[13] E. I. Goldberg, M. R. Prasad, and R. K. Brayton, "Using SAT for combinational equivalence checking," in *Proc. Design Automation & Test Europe Conf.*, Mar. 2001, pp. 114–121.

[14] A. Biere, A. Cimatti, E. M. Clarke, M. Fujita, and Y. Zhu, "Symbolic model checking using SAT procedures instead of BDDs," in *Proc. Design Automation Conf.*, June 1999, pp. 317–320.

[15] R. G. Wood and R. A. Rutenbar, "FPGA routing and routability estimation via Boolean satisfiability," *IEEE Trans. Very Large Scale Integration Systems*, vol. 6, no. 2, pp. 222–231, June 1998.

[16] J. P. Marques-Silva and K. A. Sakallah, "GRASP: A search algorithm for propositional satisfiability," *IEEE Trans. Computers*, vol. 48, no. 5, pp. 593–595, May 1999.

[17] M. Moskewicz, C. Madigan, Y. Zhao, L. Zhang, and S. Malik, "Chaff: Efficient SAT solver," in *Proc. Design Automation Conf.*, June 2001, pp. 530–535.

[18] E. Goldberg and Y. Novikov, "Berkmin: A fast and robust SAT solver," in *Proc. Design Automation & Test Europe Conf.*, Mar. 2002, pp. 142–149.

[19] M. R. Stan, P. D. Franzon, S. C. Goldstein, J. C. Lach, and M. M. Ziegler, "Molecular electronics: From devices and interconnect to circuits and architecture," *Proc. IEEE*, vol. 91, no. 11, pp. 1940–1957, Nov. 2003.

[20] T. Hogg and G. S. Snider, "Defect-tolerant adder circuits with nanoscale crossbars," *IEEE Trans. Nanotechnology*, vol. 5, no. 2, pp. 97–100, Mar. 2006.

[21] *ACM/SIGDA benchmarks: 1993 LGSynth Benchmarks*, http://www.cbl.ncsu.edu/benchmarks/LGSynth93/.