Nanofabric PLA architecture with Flexible Nanowire-redundancy

Mandar V. Joshi and Waleed K. Al-Assadi
Department of Electrical and Computer Engineering, University of Missouri-Rolla
Email: {mvjvx8, waleed} @umr.edu

ABSTRACT

It has been shown that fundamental electronic structures such as Diodes, and FET's can be constructed using selectively doped semiconducting Carbon Nanotubes or Silicon Nanowires (CNT's, SiNW's) at nanometer scale. Memory and Logic cores using these technologies have been proposed, that use the configurable junctions in two-dimensional crossbars of CNT's. These Memories and Logic Arrays at this scale exhibit significant amount of defects that account for poor yield. Configuration of these devices in presence of defects demands for an overhead in terms of area and programming time. In this work, we introduce a PLA configuration that makes use of design-specific redundancy in terms of number of nanowires, in order to simplify the process of programming the PLA, increase the yield and reduce the time complexity and in turn, the cost of the system.

1. INTRODUCTION

The advances in Photolithographic techniques of today have made the miniaturization of electronic circuits possible. According to Moore's Law, the number of transistors per unit area would continue to double, approximately every two years. However, the applicability of Moore's law will cease to continue as the pitch sizes approach molecular dimensions. It therefore becomes necessary to explore devices and technologies that can match these trends of increase in transistors per area. We propose a technique to tolerate defects at Nanometer scale using Carbon Nanotubes and Si Nanowires.

Semiconducting Carbon-Nanotubes and SiNW's exhibit electronic properties similar to those of conventional lithographic-scale CMOS devices, in terms of electron and hole mobilities. It has been shown that chemical passivation of SiO_x shell surrounding single crystal SiNW cores can significantly enhance conductance-gate voltage behavior making these wires highly suitable to be used as Field Effect Transistors [1], and in turn building blocks for digital circuits. The electronic applications of NWs are based on diode and FET-like properties of NW junctions or *"crosspoints"* in two-dimensional arrays, called as *Nanofabrics* or *Crossbars.*

crosspoints can be grouped together to form a memory or logic device. Cha et al [2] have shown electro-mechanical switching devices using suspended nanotubes. The crosspoints at the junctions are programmed using this "Bistable" property that they exhibit. Their ON-state behavior is similar to that of a diode. When the two wires forming a junction are in close contact, the junction resistance is very small, and when the wires are far away, their resistance increases by a great extent (~33MΩ in ON state and ~10kΩ in OFF state)[2]. A crosspoint can be programmed ON or OFF by applying a voltage Differential of ~3.6V.

Synthesis of Boolean expressions can be made possible on PLA's based on Crossbars. A row in a Crossbar can be made to act as a Boolean product/sum term by programming ON only the junctions or crosspoints that correspond to the variables that take part in the term, as shown in Figure 1. Inputs A, C and D are called the ON inputs, as they take part in the evaluation of the product term. Rests of the inputs are called the OFF inputs. The programmability of a crosspoint is statistical in nature [7], and therefore such a configuration of PLAs gives a poor number of successfully configured crosspoints even for a small number of junctions to be programmed on a NW. We propose a redundancy scheme to tolerate the occurrence of crosspoint defects to obtain an acceptable yield for PLA configuration.

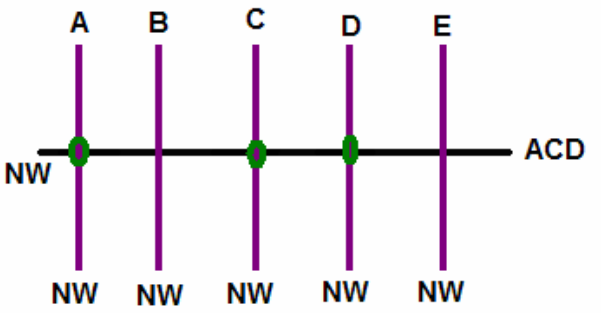


Fig.1: Generation of a product term on a NW grid

The paper is organized as follows. Sections 2 and 3 describe the defect model and introduce the problem. Section 4 deals with the previous work in terms of proposed PLA architectures and configuration algorithm. Sections 5, 6 and 7 introduce the Variable Redundancy approach and illustrate the results in terms of yield and area overhead.

2. CROSSPOINT DEFECT MODEL

The bistable property of crosspoints can be used to implement *logic blocks* in a PLA [3], or *memory cells* [4]. This is useful for implementing *NAND and NOR functions* in NanoPLA. The crosspoints can lose their

NSTI-Nanotech 2007, www.nsti.org, ISBN 1420061828 Vol. 1, 2007

programmability because of the mechanisms discussed below. We assume a random distribution of such defects throughout the crossbar, for our simulations.

Breaks in Nanowires: It has been observed that the probability of having breaks in a Nanowire increases with increase in its length. There may be some breaks during the fabrication of nanowires, on account of limitation of the fabrication techniques, and axial stress. Therefore, their lengths should, nominally, not exceed a few 10s of microns. It is reasonable to assume that as high as 5% of the Nanowires exhibit breaks, and therefore are unusable [5].

Non-Programmable crosspoints: These defects are characterized by inability of a crosspoint to be programmed "closed" or programmed "open". The latter is observed to be extremely unlikely, and therefore not considered in the present discussion. The occurrence of defective crosspoints is a function of fabrication technique, size of the array, and the random distribution of molecules at the junction area. With reasonable assumptions of operating conditions, it can be proved that the occurrence of a "permanently open" defect is largely due to absence of sufficient electrons at the junction area [7].

3.PLA CONFIGURATION REQUIREMENT

Evaluation of combinational logic outputs is performed by using AND-OR, OR-AND, NAND-NAND or NOR-NOR methods. In AND-OR implementation, the output is the logical sum of several product terms of input variables. The required AND and OR arrays can be implemented using Crossbars in Nano PLAs. A single vertical Nanowire can be dedicated for a single sum, or a single product. The presence of even a single defect at a crosspoint to be programmed on such a wire would make the entire sum/product faulty, and therefore it would result in an error at the output. Although occurrence of defects is small, the overall probability of error becomes very high, with increase in the number of crosspoints to be configured.

We define two matrices, F and G. Matrix F has 1's in place of ON inputs and 0's in place of OFF inputs. Matrix G is the defect map, and has 0's in place of defective crosspoints, and 1's in place of programmable crosspoints. The occurrence of 0's in Matrix G depends on the given defect rate. There is a successful match, only if the following condition holds:
$F(i, j) \geq G(i, j)$. This is illustrated in Figure 2.

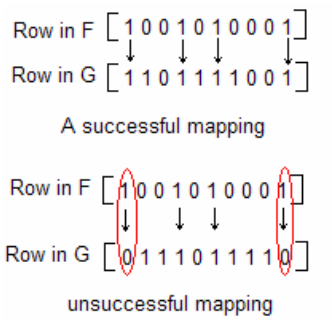


Fig. 2: Mapping Analogy

The challenge is therefore, to devise a scheme that enables the successful configuration of the PLA in the presence of given probability of defective crosspoints.

4. PREVIOUS WORK

A defect-tolerant methodology that is proposed in [7] uses a Greedy Heuristic Algorithm to find a solution to the mapping problem discussed above. The algorithm sorts the function rows in matrix F in descending order of number of ON inputs contained in them, which enhances the probability of a successful match. This algorithm is intended to be used with NanoPLA architecture proposed by DeHon et. al. in [3].

The limitation of this algorithm is that the time complexity of sorting elements in F. It is seen that the Time Complexity for sorting is an exponential function of defect rate and number of crosspoints to be programmed. It therefore becomes infeasible to use this algorithm for NanoPLA with defect rate higher than 20% and number of ON inputs than 10%.

Our redundancy technique can be used in conjunction with NanoPLA in [3] or *Nanofabric Molecular Logic Array* (MLA) Proposed by Goldstein et. al. in [6].

5. OUR APPROACH: NANOWIRE REDUNDANCY

We target a NanoPLA with higher defect rates than 20%. It can be noted that the Greedy Algorithm in [7] consumes a very high time complexity for higher defect rates. To minimize this time complexity, we introduce redundancy in terms of number of nanowires and observe the yield rates and area overhead. We dedicate 'n' number of NWs per input variable, where the value of 'n' is governed by the number of times the variable is used in the function set. It follows that we have a set of "n" crosspoints, any of which being programmable, makes the PLA work. The number of crosspoints allocated for a particular sum or product term depends on a number of factors. They include the defect rate, size of the PLA in use, number of variables taking part in the evaluation of product term.

6. ADAPTIVE VARIABLE REDUNDANCY

We propose to allocate different "Levels" of redundancy for variables with different values of ON inputs. The number of optimum number of redundant wires for a function is governed by the number of ON inputs it contains, i.e. more number of ON inputs would result in increased number of redundant NWs for the function, as shown in Figure 3. Since the allocation is made on ad-hog basis, it is called the Adaptive Variable Redundancy or AVR. As seen in Figure 3, variable A is used 8 times, and therefore maximum numbers of NWs (four as an example) are dedicated for it. Variable B is used only three times out of 16, so only two NW's are allocated for it.

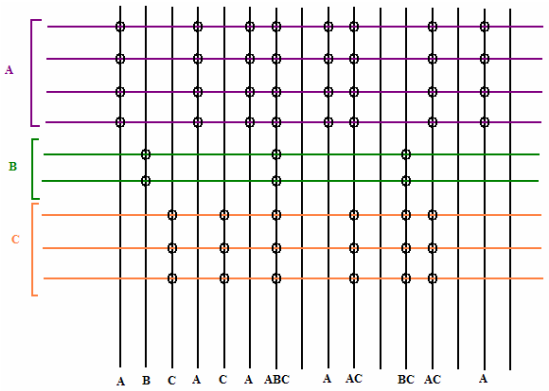


Fig. 3: Illustration of AVR

AVR Allocation Algorithm

1. Initialize the size of the Function Matrix F

2. Initialize the defect Density d=P_{def}
//Simulates matrix G.

3. Initialize Probability of ON input occurrence, P (ON)

4. For i= 1 to size of F

flex[i]=1;
//stores the redundant NWs for the given variable needed for a given number of programmable crosspoints

Loop: d=d^flex[i];

Find the probability of a defect for all possible defect orientations for a row, provided 'i' crosspoints need to be programmed. Store it in a[i]

Find the probability that the number of crosspoints to be programmed on a row is 'i' using Binomial Distribution; and store it in b[i]

Find the product c[i]=a[i]*b[i]. This gives the defect Probability for a row with 'i'
crosspoints to be programmed

if c[i]> threshold

flex[i]++;
loop;
end if
end for.

7. SIMULATION RESULTS

Adaptive Variable Redundancy Allocation Algorithm has been developed in such a way that the redundancy allocation for a given defect rate and ON input density, depends on *how many crosspoints* in a row need to be configured. In other words, the algorithm first calculates the probability of having a defective configuration for a given number of crosspoints to be programmed in a row. It then compares this value with a certain pre-defined threshold to iterate the amount of redundancy required. It follows that more likely a certain combination is, more is the redundancy allocated to it, as shown in Figure 4. More redundant resources make sure an acceptably high yield.

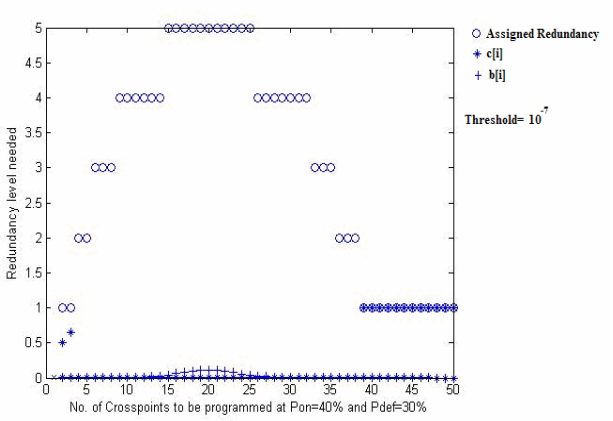
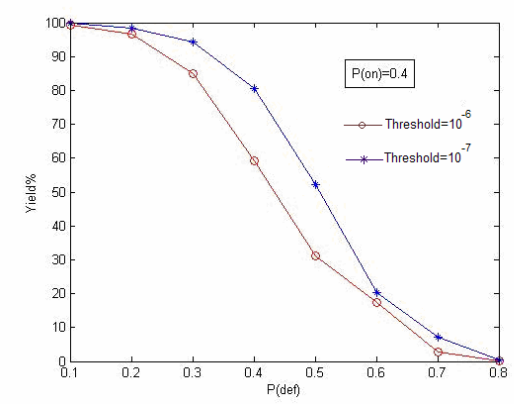
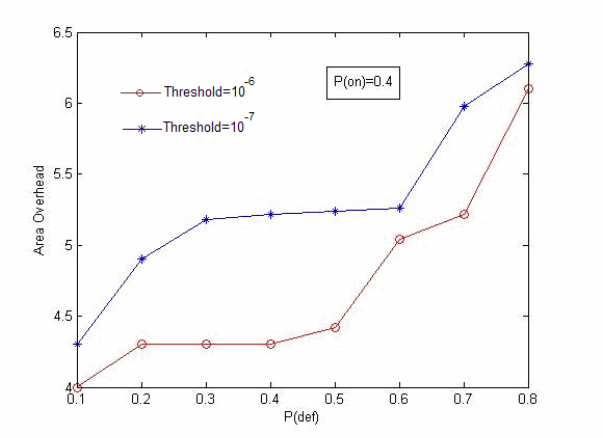


Fig 4: Redundancy levels vs. number of ON inputs



(a)

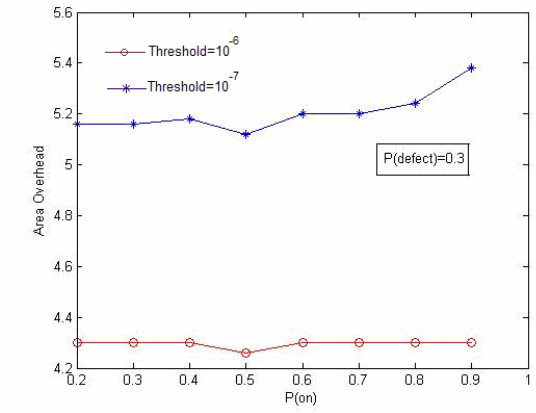
NSTI-Nanotech 2007, www.nsti.org, ISBN 1420061828 Vol. 1, 2007



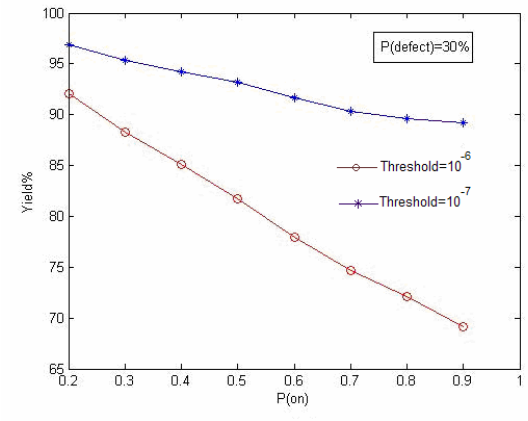
(b)

Fig. 5: Yield and Area overhead vs. Defect rate at constant P (on)

For a typical case, where P(on)= 0.4 and 20% defect density, we have a yield higher than 95% at a cost of an area overhead approximately equal to 4.8



(a)



(b)

Fig. 6: Yield and Area overhead vs. P (on) at constant Defect rate

The Adaptive Variable Redundancy Allocation Algorithm tries to fix the redundancy level as a function of input parameters such as Defect density, array size, P (on) etc. Since it always sets this value on a comparison with defect probability with the threshold value, the threshold proves to be a vital parameter in deciding the amount of redundancy. In turn, it also sets the area overhead and yield values. It can be observed in Figure 5 and Figure 6 that a lesser threshold value gives the system a greater yield at an expense of area overhead. It can also be expected that a greater threshold value would give rise to a lesser redundancy, and therefore decreasing the area overhead.

8. CONCLUSIONS

Variable Redundancy greatly increases the yield in a NanoPLA configuration even with defect rates well above 20%. Adaptive Variable Redundancy allocates redundancy based on the factors that affect the yield directly, and therefore shows better yield results than Greedy Algorithm. Since AVR needs sequential configuration, no sorting algorithm is needed before configuration. The need to obtain the defect map is completely eliminated using Variable Redundancy. The time complexity for configuration for AVR is significantly lower than that with Greedy Heuristic Algorithm. All these advantages lead to an increased area overhead.

REFERENCES

[1] Y Cui, Z Zhong, D Wang, W Wang, C Lieber, *"High Performance Silicon Nanowire Field Effect Transistors"*, Nano Letters Vol. 3, No. 2, 149-15, 2003

[2] S Cha, J Jang, Y Choi, G Amaratunga, D Kang, D Hasko, J Jung, N Kim, *"Fabrication of a nanoelectromechanical switch using a suspended carbon nanotube,* Applied Physics Letters, 083105, 2005

[3] A. DeHon and M. J. Wilson, "*Nanowire-Based Sublithographic Programmable Logic Arrays*", in FPGA, pp. 123–132, 2004.

[4] A. DeHon, S. Goldstein, P. Kuekes, "*Nonphotolithographic NanoscaleMemory Density Prospects"*, IEEE TRANSACTIONS ON NANOTECHNOLOGY, VOL. 4, NO. 2, MARCH 2005

[5] T. Hogg, G. Snider, *"Defect-tolerant logic with Nanoscale crossbar circuits"*, HP Labs, May 2004. URL:http://www.hpl.hp.com/research/idl/papers/molecularAdder/circuits.pdf

[6] S. C. Goldstein and M. Budiu, "NanoFabrics: Spatial Computing Using Molecular Electronics," in ISCA, June 2001, pp. 178–189

[7] H. Naeimi,, *"A Greedy Algorithm for Tolerating Defective Cross points in Nano PLA Design"*, MS Dissertation, California Inst. Of Technology, 2004