BUILDING CAD PROTOTYPING TOOL FOR EMERGING NANOSCALE FABRICS

Catherine Dezan, Loic Lagadec
Université de Bretagne Occidentale, France
LESTER A&S, CNRS
{Catherine.Dezan,Loic.Lagadec}@univ-brest.fr

Michael Leuchtenburg, Teng Wang,
Pritish Narayanan, Andras Moritz
University of Massachusetts at Amherst
Electrical and Computer Engineering
{mleuchte, pnarayanan, twang, andras}@ecs.umass.edu

ABSTRACT

The design of CAD tools for nanofabrics involves new challenges not encountered with conventional CMOS technology. In this paper, we describe the design of a *prototyping* CAD tool targeting design automation of applications on a wide range of nanofabrics. Our proposal is based on a variety of models that capture as well as isolate the differences between the fabrics. This tool supports the design flow from behavioral description to final layout. It integrates fault-tolerant techniques and fabric-related density transformations with more conventional design automation techniques. After an overview of common requirements, physical models, and associated techniques, a case study in the context of NASIC fabrics is used to illustrate some of the concepts.

1. INTRODUCTION

As an alternative to CMOS based designs, novel nanofabrics are being proposed based on a combination of lithographic processes and bottom-up self-assembly based manufacturing. These fabrics include NanoPLA [2], CMOL [7], FPNI [10], and NASIC[12] - to name a few. They are based on a variety of devices such as FETs, spin-based devices, diodes, and molecular switches. Furthermore, all these architectures would include some support in CMOS: some like FPNI would move the entire logic into CMOS, others, like NASIC, would only provide the control circuitry in CMOS. Other differences include fault handling: some proposals would use reconfigurable approaches, while others would rely on built-in techniques based on redundancy, voting, error correction, and/or unique fabric structures. The architectures proposed range from general purpose processors, to programmable logic arrays similar to FPGAs, and to more specialized devices such as cellular arrays and cellular neural networks.

In order to implement an application on a nanofabric, specific tools are already proposed by the respective research groups [2][8]. It is clear that CAD tools are necessary to be able to design and evaluate the capabilities of larger-scale systems. As the underlying technologies are still evolving according to advances in devices, manufacturing, and fabric structures, CAD tools for nanofabrics should be ideally generic enough to integrate added features or to enable new paradigms.

This paper proposes a *prototyping CAD tool* that considers a nanofabric specified through a variety of models to allow optimizations on generic data structures. Through a computational model, an architectural model, a technological model and a fault model all key aspects of a particular fabric can be captured. The proposed models interact with the behavioral tools and the physical tools to produce an abstract layout of the design from high-level description. Parts that are mapped to nanoscale are separated from parts that use conventional CMOS technology.

Through this design process, a particular attention is given to fault tolerance techniques. Fault management is one of the key differences between CMOS and nanoscale design automations. To date, this tool utilizes redundancy-based techniques, error-correction circuits, and defect map information for reconfigurable fabrics to achieve reliable computation. Our objective is to enable future extensions as it would be impossible to start with all optimizations and fault management techniques in place in the first version. Much of this is still cutting edge research.

In summary, this paper makes the following key contributions: (1) it discusses the overall architecture of a generic CAD prototyping tool for nanoscale designs; (2) it introduces nanofabric related models to allow fairly generic processing in the presence of very different assumptions. Our objective is to create a tool that could be used by most research groups in this field. The paper is organized as follows. Section 2 gives an overview of the general organization of the proposed CAD tool and the following sections discuss the different contributions. Each section gives an illustration based on NASIC Fabric.

2. A PROTOTYPING TOOL BASED ON NANOFABRIC SPECIFICATION

The prototyping tool presented here is named NanoMadeo. It is based on four specific models that specify the nanofabrics. These models provide some abstractions of the nanofabrics concerning their computation paradigm (*computational model*), their structural organization (*structural model*) and technological constraints (*technological model*), including their fault-tolerance ability (*fault model*). These models interact with behavioral transformations, structural transformations and physical tools, needed to design and to implement an application onto the nanofabric support. The general flow of this tool is presented in figure 1 and the specific models are detailed in the following sections.

2.1 Computational Model

CAD tool for emerging nanofabric is intended to handle use both traditional CMOS and nanoscale technologies. The distribution of functionality between the two depends on the nanoscale capabilities, the trade-off between area and performance and the reliability of the underlying nanoscale technology.

For instance, the nanoscale parts of the system can be used solely for computation in order to gain orders of magnitude improvements in density and performance compared to CMOS technology [11]. However, this may have some drawbacks like the need for signal restoration when using nanowire diodes or the inversion problem with Single Electron Transistor (SET) technology. The solution can be to add logic in nanotechnology (FET logic for signal restoration) or some dedicated CMOS logic/cells (for inverter functionality).

Nanoscale technology could also be utilized for interconnects only to speed-up communication that can be problematic in deep submicron technology. NanoMadeo must handle both of these extremes in order to be useful as a prototyping tool for new nanotechnology designs.

The computational model specifies the role of the nano or CMOS segment in computation and interconnect. This division of labor is a new requirement with hybrid fabrics, which all early nanoscale computation devices are likely to be.

ISBN :978-2-35500-003-4

2.2 Architectural Model

The architectural model contains information about the building components and topological structure of the fabric. These components correspond to nanoscale or CMOS elements necessary to build the architecture on the fabric. These can be classified into basic devices, pre-composed blocks and wires as shown in figure 2.

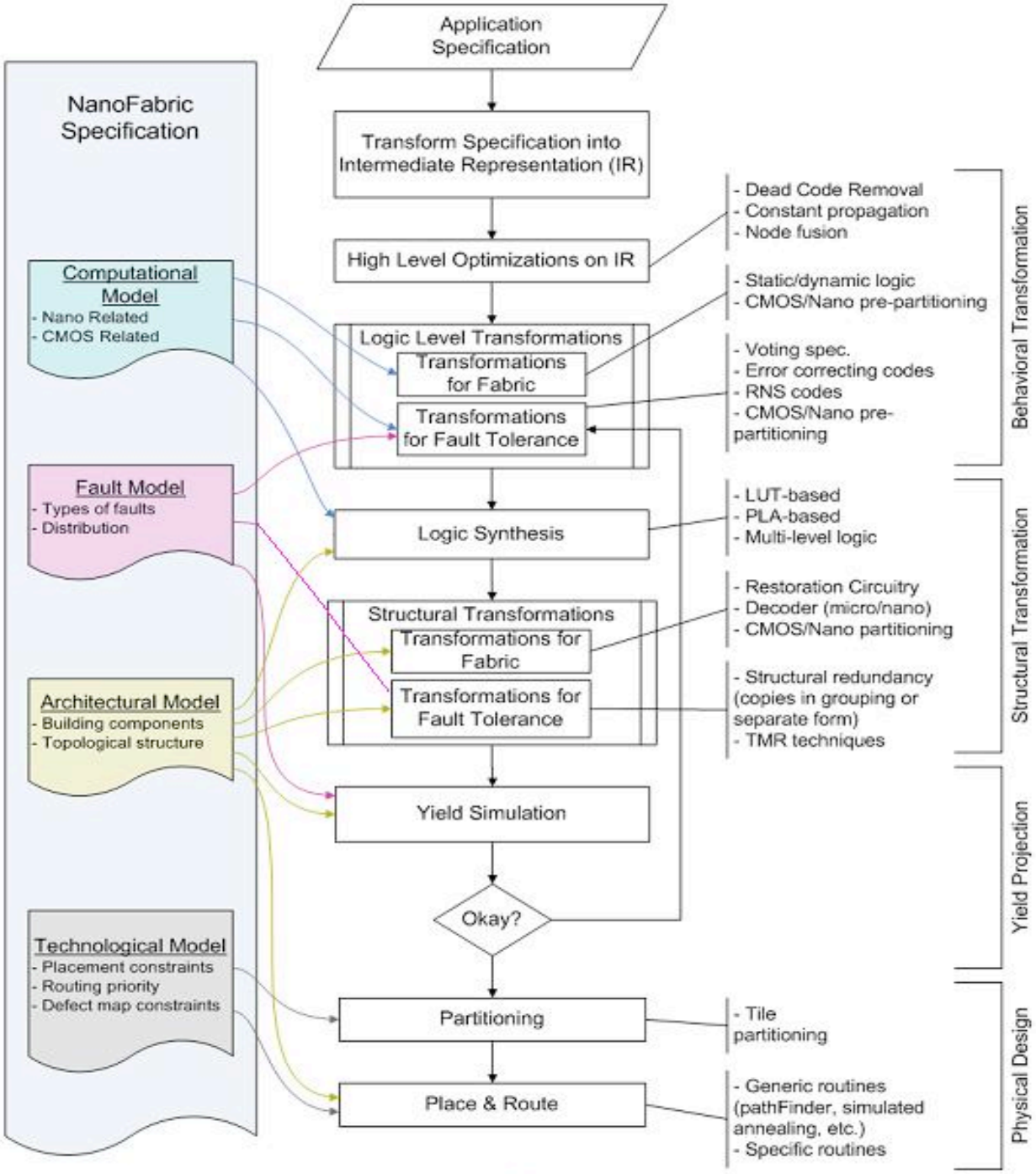

Figure 1. Flow of NanoMadeo

The architectural model also contains information about the types of tiles specific to fabric: nanoBlocks for NanoPLA [2], tiles of basic cells for CMOL [7], hypercells for FPNI [10] and tiles for NASIC[12]. The model describes the topological organization of the nano and micro components, including their hierarchical structure in tiles.

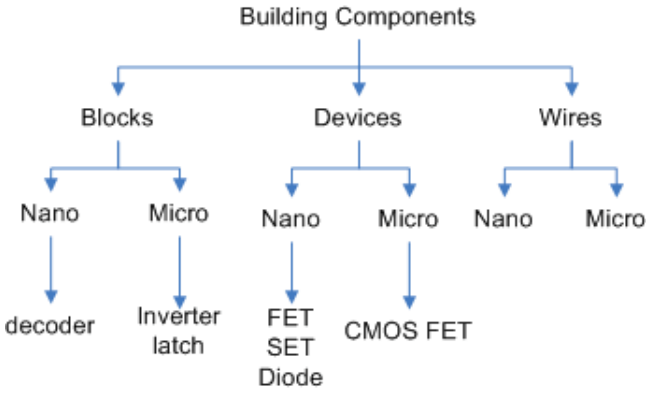

Figure 2. Building components used in architectural model

2.3 Technological Model

The technological model contains more information on the physical constraints based on the underlying technology and useful for the place-and-route routines. Nanoscale systems often do not allow arbitrary routing and placement, complicating their design as compared to CMOS designs. For example, a placement constraint related to the fabric might be the doping constraints in the NASIC fabric that limits each type of transistor to one dimension: horizontal or vertical.

Constraints on routing are particularly important in the case of reconfigurable fabrics where connections are limited to certain routes. The costs for these routes also give guidance to the routing procedures for choosing the best routes. Another constraint may be defects present in a particular chip in the case of a reconfigurable fabric. These present additional routing and placement constraints in configuring around the defects[11].

2.4 Fault Model

As nanoscale computational fabrics are commonly based on bottom-up manufacturing processes, reliability of this new kind of circuit is imperfect. Information on the expected faults is provided through the fault model, including what types of faults are expected, their distribution, and their probability.

Fault types include: permanent defects due to the manufacturing process such as stuck-on or off transistors or broken nanowires; transient faults due to internal noise, particle impacts, or electromagnetic interference; and process variation, including doping, channel length, wire thickness, and others. For each possible fault in a given technology, the rate and distribution (uniform/clustering [11]) is included in the fault model.

2.5 Example: NASIC Fabric Description

The models described above are illustrated through some emerging nanoscale fabrics in table I. Special focus is given on NASIC fabric for deeper explanations as case study.

NASIC [19][20][21] is a hybrid system based around tiles of nanowires and FETs with CMOS providing support and some control circuitry. The tiles are made up of crossed nanowires with FETs at the intersections, forming cascaded PLA-like structures. In each tile (or supertile), there is one couple(or several couples) of planes of transistors, one with the channels running horizontally and one with the channels running vertically. Thus, each tile implements two basic logic functions and can implement any logic function using two-level logic.

In the NASIC fabric, each nanotile is surrounded by microwires which provide power and control signals. The control signals implement typically various styles of a dynamic control scheme. This use of dynamic logic puts a synchronization constraint on the synthesis of applications onto NASICs, which NanoMadeo must manage. CMOS also provides support for modular redundancy schemes, encoding/decoding of inputs and outputs for the entire system (not between tiles), and control signal generation. The distribution of role between CMOS and nano layer is driven by the computational model of the NASIC fabric and explicit two main points: one is the computation organization of the nanogrid in two level logic in order to be mapped later into PLA structures (information related to synthesis), the other is related to the control aimed to be mapped at CMOS level.

TABLE I

Main Features of Some Hybrid Nanofabrics Related to Models

Fabrics Models	NanoPLA	CMOL	FPNI	NASIC
Computational Nano CMOS	Computation, interconnect limited to I/O	Computation, interconnect Specific Computation(Inv)	Interconnect only Computation	Computation, interconnect Control
Architectural Devices Structure	FET, Diode 2D-grid	Molecular switch, SET 3D-grid	CMOS 3D-grid	FET 2D-grid
Techno placement routing	Connection restricted	Connection restricted	Connection restricted	Doping constraints
Fault	Permanent	Permanent	Permanent	Transient, permanent

The architectural model of the NASIC fabric points out the building components used (here: FET, nanowire, mircowire) and explicit the structural organization in tiles based on topological rules on building components. Building components may be assembled in a predefined way (for instance for the nano and microwires, if the number of nanotiles are supposed to fixed) or in an adaptive way related to the application (placement of FETs on the PLA structure). In the latter case, these topological rules are active during the place-and-route phase.

The technological model provides the physical constraint of the NASIC fabric essentially due to the doping constraints of the two types of transistors that can be used (N-FET, P-FET). This constraints introduce some complexity in the placement routines inside one tile. The mapping onto different tiles partially replaces the doping constraints by I/O constraints between tiles.

The fault model takes into account two types of faults: permanent and transient fault. Distribution and rate of these types of faults have an impact on the reliability of the implementation. The fault tolerant transformations have different capabilities in masking errors that can be evaluated using a yield simulator [12]. This point is more detailed in section 4.

3. APPLICATION SPECIFICATION AND BEHAVIORAL TRANSFORMATIONS

The behavioral description of an application is written in an object-oriented language (Smalltalk) that is similar to the traditional description used in Madeo [17]. The compilation procedure produces Directed Acyclic Graphs (DAGs), which are the intermediate representation (IR) used by NanoMadeo. This model is largely unchanged between architectures, allowing the same behavioral description to be used in comparing designs.

3.1 Behavioral Transformations

These transformations include nano/CMOS pre-partitioning taking into account the computational model: computation and interconnect tasks will be assigned to CMOS or nanoscale parts .

Fault-tolerance techniques may be applied at this level and correspond to three kinds of transformations: a) transformation introducing voters like TMR in order to duplicate portion of codes and to vote between copies, b) transformation introducing different data encoding based on redundant codes (RNS codes, Error-correcting codes, expressed at this level like data types, c) transformation changing the physical support: the computation can be realized by the CMOS part to be more reliable.

3.2 Illustration: Wisp0 application

WISP-0[20] is a stream processor, built on NASIC, that implements a streaming processor architecture with 5-stage pipeline: fetch, decode, register file, execute, and write back. It is a multi-tile design with 5 nanotiles. A key feature is that intermediate values during execution are often stored on the nanowires directly without explicit latches using a three-phase dynamic control. Other key aspects relate to its fault-masking strategy and density optimizations.

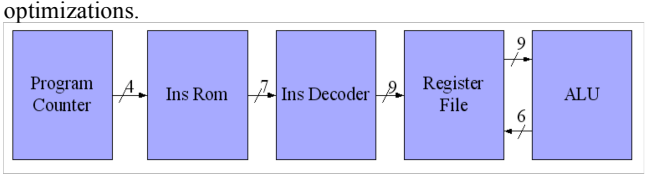

Figure 3. WISP-0 Block Diagram

The specification of WISP0 with NanoMadeo is functional, including :

- A synchronization primitive
- a reflexive operator ('yourself' operator) to define feed-back
- specific types to introduce redundancy

We give in figure 4 an example of a DAG produced by the compiler from source code defining the ALU and Register File (RF) stages of WISP-0. In this DAG, nodes correspond to function calls or operators that will be synthesized into logic PLA blocks. This description implicitly describes some data synchronization, but this information could be more explicit and could be managed through specific behavioral transformations. No explicit partitioning between CMOS/Nano is done at this point because every functionality is aimed to be implemented on nano parts in the case of NASIC fabric. Nevertheless, if some fault-tolerant parts in CMOS are needed, this information needs to be explicit (for instance, the generation of CMOS voter by transformation for fault-tolerance in TMR case). Other transformations for fault-tolerance

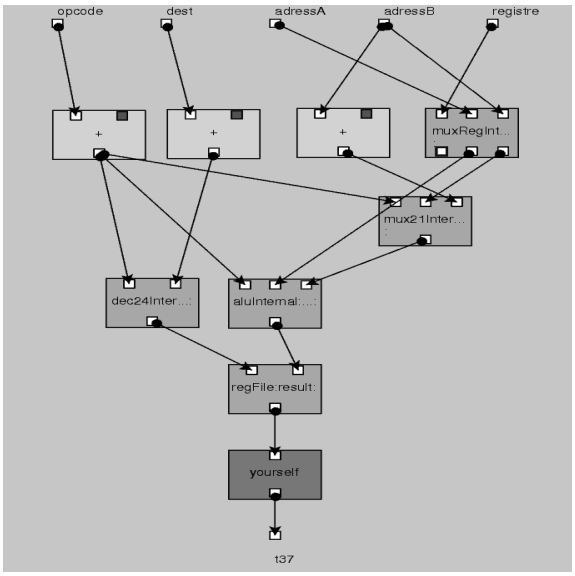

Figure 4. DAG representing ALU and RF

can be applied by injecting specific data types for the inputs. These types represent future data encodings for the input data; for instance, in the case of NASIC fabric, BCH codes (as error correcting codes ECC) are used to introduce built-in redundancy.

4. SYNTHESIS, STRUCTURAL TRANSFORMATIONS AND YIELD PROJECTION

4.1 Synthesis

The resulting logic is then synthesized in the appropriate type of logic (PLA, LUT, multi-level logic) with the appropriate building components as defined by the computational model and architectural model. Standard external tools such as SIS are used for this process. This is done on a block-wise basis, with each operation as compiled from the high-level code compiled as a single block. Different levels of operator decomposition can be applied, allowing the complexity of each block to be traded off against the number of blocks.

4.2 Structural Transformations

Once the initial structural representation of the application has been generated through synthesis, transformations at the structural level are applied. The synthesis is based only on the type of logic and

may not take into account all structural requirements. An example is the signal restoration required by NanoPLA. At this point, logic is synthesized for the combinatorial parts such as decoders and sequential parts such as registers or dynamic logic controls are defined around the synthesized segments as required to support the architecture.

Structural level fault-tolerance related transformations are also done in this step. These include techniques like N-way redundancy to provide additional copies of input/output signals and of some intermediate signals (the logic minterms in a case of PLA structures). Modular redundancy techniques are also possible at this stage – specific structures are selected and voter circuits are provided to implement TMR or similar schemes, but at this level, a more detailed architecture of this kind of circuitry is provided.

4.3 Yield Projection

The structural representation of the circuit plus the fault distribution given by the fault model can be used to make yield projections. This is performed by an external yield simulator. A yield simulator for PLA-based structures proposed in [6] could be used for different kinds of 2D nanofabrics. Yield estimation can also be done using Monte Carlo simulation [11].

If the predicted yield is not satisfactory, it is possible to iterate, applying different kinds of fault-tolerant transformations to different portions of the application. These iterations may continue until an acceptable level of yield is reached.

4.4 Structural Transformations and Yield Projection for WISP-0

In the case of NASIC fabric, NanoMadeo utilizes the external synthesis tool SIS to perform the two-level logic synthesis of PLAs associated to each node of the DAG. Assembly of synthesized portions is addressed by NanoMadeo to define the complete logical structure of the WISP-0 application. WISP-0 may use some structural fault-tolerance techniques such as TMR and N-way redundancy of signals. The result of synthesis is then transformed here to implement these techniques, when they are in use.

We have developed a yield simulator to evaluate fault tolerance techniques in NASICs. The simulator generates random defect maps for designs based on a defect model and runs logic simulations on them, testing with many different possible sets of input. By measuring what proportion of the generated defect maps result in correct output when simulated, the yield can be estimated. NanoMadeo can automatically call the yield simulator to evaluate defect and fault tolerance techniques. One example of output after several runs of the yield simulator, using different fault rates and different fault-tolerance techniques(TMR, ECC,N-way) is shown graphed in figure 5. This graph provides information on the efficiency of the fault-tolerant techniques related to the fault rate and the types of permanent defects (for instance, if the fault rate is above 6%, the yield is better with ECC techniques considering 10% Stuck-off and 90% Stuck-on).

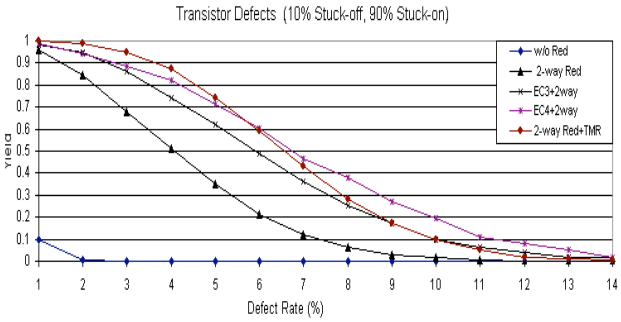

Figure 5. Graph of yield simulator output for WISP-0

5. PHYSICAL DESIGN

Nanofabrics are generally organized into tiles, hypertiles or nanoblocks that correspond to clusters of PLAs, basic cells or hypercells. The partitioning techniques used to define such blocks are based on clustering heuristics for PLA packing, as in PLAmap[4], T-VPACK [15] or Singh Algorithm [9].

The parameters for clustering are the number of elementary cells or P-terms of the PLA and the number of inputs and outputs associated with the cluster. The placement problem consists of placing each basic cell inside a cluster, once the clusters are defined. This is achieved using generic optimization heuristics like simulated annealing, using e.g., congestion costs in the case of reconfigurable fabrics.

Routing procedures for nanofabrics can use adaptive maze router algorithms like Pathfinder from VPR, or they can be more specific to the fabric using, for example, custom adaptation of shortest Steiner tree problems or other VLSI algorithms [1].
For reconfigurable fabrics, a defect map provides extra constraints for placement and routing to configure around the defects previously detected.

Table II gives an overview of the different algorithms applied in physical layout tools for the NanoPLA, CMOL and FPNI fabrics. Physical tools for nanofabrics use two kinds of algorithms or heuristics: adaptive generic algorithms or custom procedures. Adaptive generic algorithms include general purpose optimization heuristics like simulated annealing or genetic algorithms and algorithms for FPGAs like Pathfinder and PLA clustering as the ones implemented in Madeo [18] or the VPR tool.

TABLE II

Physical Tools Published For Specific Nanofabrics

	NanoPLA [2]	CMOL [8]	FPNI [10]
Partitioning	PLAMAP [4]	T-VPACK [15]	Singh's greedy algo [9] Specific cost
Placement	Simulated Annealing(VPR-like)	Simulated annealing(VPR-like) Modified congestion cost function	Simulated annealing(VPR-like)
Routing	NPR (custom tool) Pathfinder based	Custom tool RSA heuristic based [14]	maze router (PathFinder-like) with several iterations

5.1 Wisp0 layout

Based on the architectural and technological model of NASIC fabric, an abstract layout can be produced taking into account the doping constraints inside one tile. In figure 6, we present the abstract layout of WISP-0 onto a nanogrid of three tiles, partially integrating some fault-tolerant techniques. A more efficient place-and-route algorithm without constraints on the size of the tile is under study.

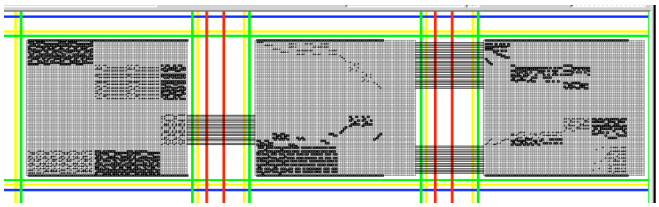

Figure 6. Abstract WISP-0 Layout from NanoMadeo

6. CONCLUSION

In order to handle next generation hybrid nano-architectures, CAD tools will have to evolve. Highly heterogeneous multi-part fabrics introduce new challenges which must be met to efficiently use those new fabrics in building real applications. In this paper, we have shown that the proposed tool, NanoMadeo, can handle many of these challenges and can be used productively for work on NASIC. Its generic design will make it easy to adapt it for work on other hybrid architectures while using much of the same functionality already implemented.

REFERENCES

[1] S. H. Gerez, *Algorithms for VLSI Design Automation* Hoboken, NJ: Wiley, 1999.

[2] A. Dehon, "Nanowire-based programmable architectures", *ACM Journal on Emerging Technologies in Computing Systems,* vol. 1, pp. 109-162, July 2005.

[3] A. Dehon, "Design of programmable interconnect for sublithographic programmable logic arrays", *Proceedings of the 2005 ACM/SIGDA 13th international Symposium on Field-Programmable Gate Arrays,* pp. 127-137, February 2005.

[4] D. Chen, J. Cong, M. Ercegovac and Z. Huang, "Performance-Driven Mapping for CPLD Architectures", *IEEE Transactions on Computed-Aided Design for Integrated Circuits and Systems,* vol. 22, pp. 1424-1431, October 2003.

[5] L. McMurchie and C. Ebeling, "PathFinder: a negotiation-based performance-driven router for FPGAs ", *Proceedings of the Third International ACM Symposium on Field-Programmable Gate Arrays, FPGA'95,* pp. 111-117, 1995.

[6] F. Angiolini, M. H. B. Jamaa, D. Atienza, L. Benini and G. D. Micheli, "Improving the fault tolerance of nanometric PLA designs", *Design, Automation & Test in Europe Conference and Exhibition DATE 2007*, pp. 570-575, April 2007.

[7] D. B. Strukov and K. K. Likharev, "CMOL FPGA: a reconfigurable architecture for hybrid digital circuits with two-terminal nanodevices", *Nanotechnology,* vol. 16, pp. 888-900, 2005.

[8] D. B. Strukov and K. K.Likharev, "A Reconfigurable architecture for hybrid CMOS/nanodevice circuits ". *Proceedings of the 2006 ACM/SIGDA 14th international symposium on Field-programmable gate arrays,* pp. 131-140, 2006.

[9] A. Singh and M. Marek-Sadowska, "Efficient circuit clustering for area and power reduction in FPGAs ", *Proceedings of the 2002 ACM/SIGDA tenth international symposium on Field-programmable gate arrays,* pp. 59-66, 2002.

[10] G. S. Snider and R. S. Williams, "Nano/CMOS architectures using a field-programmable nanowire interconnect ", *Nanotechnology,* vol. 18, 11 pp, 2007.

[11] C. He and M. F. Jacome, "Defect-Aware High-Level Synthesis Targeted at Reconfigurable Nanofabrics", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 26, pp. 817-833, May 2007.

[12] C.A. Moritz, T. Wang, P. Narayanan, M. Leuchtenburg, Y. Guo, C. Dezan and M. Bennaser, "Fault-tolerant nanoscale processors on semiconductor nanowire grids ", *IEEE Transactions on Circuits and Systems I Special Issue on Nanoelectronic Circuits and Nanoarchitectures1*, in press.

[13] J. Kong, "CAD for nanometer silicon design, challenges and success", *IEEE Transaction on very large scale integration systems,* vol.12, pp. 1132-1147, November 2004.

[14] S. K. Rao, P. Sadayappan, F. K. Hwang and P. W. Shor, "The rectilinear Steiner arborescence problem", *Algorithmica,* vol. 7, pp. 277-288, December 1992.

[15] V. Betz, J. Rose and A. Marquardt, *Architecture and CAD for Deep-Submicron FPGAs*: Kluwer Academic Publishers, 1999.

[16] E. M. Sentovich, K. J. Singh, L. Lavagno, C. Moon, R. Murgai, A. Saldanha, H. Savoj, P. R. Stephan, R. K. Brayton, A. L. Sangiovanni-Vincentelli, "SIS: A system for sequential circuit synthesis", *Technical Report, UCB/ERL M92/41, Dept. of EECS, Berkeley,* May 1992.

[17] E. Fabiani, L. Lagadec, B. Pottier, A. Poungou and S. Yazdani, "Abstract execution mechanisms in a synthesis framework", *Workshop on Non-Silicon Computations (NSC3)*, June, 2004.

[18] L. Lagadec. "Abstraction, modélisation et outils de CAO pour les circuits intégrés reconfigurables", *Ph.D. thesis, Université de Rennes1,* Rennes, France, 2000.

[19] C. A. Moritz and T. Wang, "Latching on the wire and pipelining in nanoscale designs", *Non-Silicon Computing Workshop, NSC-3*, 2004.

[20] T. Wang, M. Bennaser, Y. Guo, and C. A. Moritz, "Wire-streaming processors on 2-D nanowire fabrics", *Nanotech 2005, Nano Science and Technology Institute,* 2005.

[21] T. Wang, M. Bennaser, Y. Guo, and C. A. Moritz, "Self-healing wire-streaming processors on 2-d semiconductor nanowire fabrics", *Nanotech 2006 . Nano Science and Technology Institute,* 2006.