A large multi-agent system with noise both in position and control

Automatic Place&Route of Nano-Magnetic Logic Circuits

M. Vacca, S. Frache, M. Graziano, L. Di Crescenzo, F. Cairo, M. Zamboni
Dipartimento di Elettronica e Telecomunicazioni, Politecnico di Torino, Italy

***Abstract*—The analysis of effective expectations on emerging nanotechnologies, like Nano-magnetic Logic, is currently a difficult task. The lack of tools that enable the design at logic and physical level of nano-circuits does not allow to inspect properties that can be derived only considering circuits of reasonable complexity.**

We present results of an unprecedented Place & Route engine for Nano-magnetic logic, integrated in our tool for nanotechnologies design exploration. We developed and compared several algorithms to tackle Nano-magnetic Logic constraints and limitations, derived by real-life technological implementations, on complex combinational circuits (ISCAS85 benchmarks) and show to what extent Nano-Magnetic Logic can advance CMOS.

I. Introduction

Emerging technologies are studied either with a detailed device physics point of view or following an approximated architectural approach. None of the two methods allows to obtain a reliable evaluation of the competitiveness of a new technology. Two are the requirements to make this purpose a real one: 1) circuits of reasonable complexity should be tackled in order to pinpoint credible advantages, constraints, drawbacks, aftereffects at physical and functional level [1][2]; 2) circuits and architectures should be carefully described and organized on the basis of detailed technological constraints and physical properties of devices in order to capture realistic features [3][4]. In both cases automatic tools are essential [5][6][7][8][9]. Existing CMOS tools cannot be used for this purpose because the circuits layout of emerging technologies is different and different constraints must be considered with respect to CMOS technology. On the contrary, in several cases, the computational paradigm and technological constraints of emerging technologies are so different that only a fresh approach assures a successful result. Nonetheless, this rethinking might find advantage from existing ideas, algorithms and methods.

We are working in this direction as follows. A) We are exploring a few different technologies (e.g. Quantum Dot Cellular automata, NanoFabrics, Nanomagnetic logic) [10][7] to generalize the approach. In this work we focused on Nano-Magnetic Logic (NML - see figure 1 and section II) [3] which has been experimentally demonstrated, is considerably different from CMOS, is extremely promising for power dissipation and enables a “Logic-in-memory” approach. We do not consider here different implementations of the QCA principle, because they do not have a solid experimental validation as NML, the algorithm here proposed can be adopted also to generic QCA circuits. However in that case the layout constraints are different so the layout should be not as optimized as in the NML case. B) We are working both on the device physics side, with experiments and detailed simulations [4], and on the architectural behavior [2][11], integrating the two views to obtain interesting results on potential advantages and issues of this technology. C) We are facing the design-automation side developing a tool for nanoarchitctures design (see section III) that from a VHDL description of a nano-circuit enables its physical design and detailed simulation [12]. In this work we focused on NML physical-design, developed an unprecedented automatic engine for placing and routing combinational NML circuits (see sections IV and V), considering constraints derived by experiments. Previous approaches [6][8][9] were based on general QCA, did not consider experimental limitations and implemented relatively simple approaches toward the solution for elementary circuits. We selected several algorithms from traditional technology and adapted and renewed them to solve NML technology issues. We tackled classic benchmarks (ISCAS85) and were able to compare them to CMOS up-to-date technologies in terms of area, power and routing congestion.

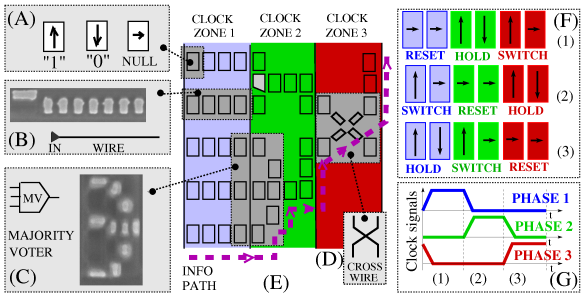


Fig. 1. A) Nanomagnets magnetization encoding logic values B) Nano-Magnetic wire we fabricated C) Majority Voter we fabricated D) Cross-wire for signals intersection E) A circuit layout organized in three clock zones, where an example of information path is underlined by dashed line F) Clock zones functions evolving in time G) Clock phases in time.

While the general concepts behind the algorithms used are derived by the literature, manifold variants, specific adaptations and new versions have been done in our implementation. The novelty of this work resides both in the variants and optimization of the single algorithms, and in how these algorithms are combined together and evolved to generate a layout compliant with the technological constraints of this technology. This approach leads to two unprecedented and most important outcomes: 1) a NML combinational circuit

of acceptable complexity can now be automatically designed and optimized down to the physical level; 2) competitiveness of NML toward CMOS technology can now be quantified.

II. NML BACKGROUND

NML is based on single domain nanomagnets of rectangular shape sized around 60X90nm (see [3] for an exhaustive description) . The magnetization of each magnetic pill is vertically oriented (on the plane) due to the aspect ratio. The two possible vertical directions are encoded with binary information (figure 1.A). Nanomagnets placed in sequence and regularly spaced (figure 1.B) represent a wire: information is propagated thanks to anti-ferromagnetic ordering of neighbors pills magnetization. Logic functions are granted by elementary gates like, for example, Majority Voters (see in figure 1.C one MV we fabricated and its symbol). Currently this technology does not allow more than one layer: two wires can cross on the plane without interference if a specific structure is used, the cross-wire (CW, see figure 1.D). NML circuits are organized as in figure 1.E: the area is divided in zones (for example zones 1, 2 and 3 in figure), each associated to an external signal called clock. Clock signals (zones) are three, in this case, repeated in sequence along the whole circuit area. Each external signal is a magnetic field generated using different possible techniques [3][10] not discussed here for the sake of brevity. This field is necessary to drive cells into an intermediate unstable state (horizontal magnetization, see figure 1.A "null" case). When the external stimulus is removed, cells realign themselves with a vertical magnetization following the input cell. For example in the case in figure 1.F and G, three periodic clock signals with a phase difference of 120 degrees are applied to the circuit. During every time step elements where field is applied are in the RESET phase, elements where field is being removed (on the left zone) are in the SWITCH phase and are realigning in antiferromagnetic sequence, and elements where field is absent (one further zone left) are in the HOLD phase and act as inputs for the switching magnets. This organization assures a correct information propagation, provided that a maximum number of magnets (defined by the technology) is included in a single clock zone, otherwise errors might occur [3][13]. This organization has a few implication at functional and physical design level. Magnets in each zone can be thought as a latch associated to a logic function (due to the present logic gates). This means that each signal propagates as in a pipelined circuit [10]. Even for combinational functions, then, the behavior is executed in sequence of clock cycles and constraints on the correct number of clock zones must be taken into account when organizing the layout (e.g. different inputs of a gate in a clock zone should traverse the same number of clock zones to assure coherence before arriving to that gate.) Another important implication is the layout flow: in figure 1.E an example of "stepped" information propagation path is underlined by the dashed line. This has important consequences at the physical design stage (see sections IV and V). More information on circuits layout and the physical constraints which is based can be found in [14].

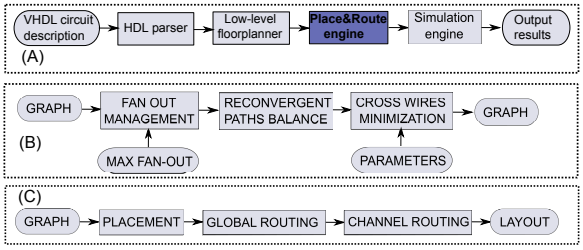


Fig. 2. A) Simplified view of tool design flow. B) Graph elaboration flow diagram. C) Physical Mapping flow chart.

III. A DESIGN TOOL FOR NANOCIRCUITS

The design tool in which the Place&Route engine is integrated is a design and simulation CAD that supports a variety of nanotechnologies, from NML [3] to nanoarrays, either based on silicon nanowires (SiNWs) or carbon nanotubes (CNTs). The tool integrates all the steps involved in the design and simulation of circuits based on emerging nanotechnologies in just one cross-platform tool. The main application organization and flow is sketched in a simple version in figure 2.A, where the part which is object of this paper is in bold. A focus on the important aspects involved in present work is pointed-out herein. It is worth noticing that wherever possible we use a taxonomy similar to the one used for standard technologies, even if in most of the cases an almost complete variation and optimization has been executed.

All the circuits can be thought-of in terms of elemental blocks at some point in the flow of the tool. For NML circuits, the elemental blocks are the magnets and all the other components (gates, Crosswires, etc.) are described in terms of small set of elemental blocks, with a specific relative position. By composing these bricks, one can get all the richness of expression needed to describe an arbitrarily complex circuit. Provided you can describe a circuit in terms of elemental components, whatever your set (library) of elemental components is, you can aggregate them and reuse the aggregates in turn as components. The user inputs a VHDL description of the circuit, by recalling the components through the familiar component statement of the VHDL language. The tool features a HDL parser, presently implementing essential parts of the VHDL specifications. An intermediate internal description of the circuit is then created, in the form of a graph data structure, capturing all the elemental block and their relations in the circuit.

For the sake of NML-Place&Route algorithms discussed in sections IV and V, a number of facts about elemental components - or aggregates of components, of course - should be known, especially about their position. This requires a coordinate system: in the case of NML, it represents an abstraction of a two-dimensional space. There are a logical and a physical coordinate system, whose mapping is handled by means of a specific class's transformation matrix, viewport and graphic window. All the components defined out of basic building bricks can have two set of coordinates. One is the relative set related to an internal reference point, and the

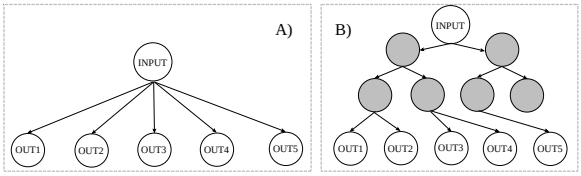


Fig. 3. A) Graph before Fan-out Limitation is applied. B) Graph after Fan-out Limitation.

second set is an absolute set of coordinate, referred to the topmost left corner of the actual circuit being described. Each component is able to map its local (relative) set of coordinates to the absolute one exploiting a lazy update technique. The advantage of this local-plus-global approach to coordinate is self-evident: virtually no calculations about the global position of the elements in a large design need to be performed, up to the point they are mandatory.

The NML-Place&Route algorithm can be divided in two parts (following the classification used for standard technology): NML GRAPH ELABORATION, discussed in section IV used to pre-process the graph representing the circuit, and NML PHYSICAL MAPPING, presented in section V, that finalizes the placement and routing.

IV. NML GRAPH ELABORATION

The NML GRAPH ELABORATION flow diagram is shown in Figure 2.B. The input is the graph generated by the HDL PARSER, which contains a structural description of the circuit mapped on the logic gates available (AND, OR, MAJORITY VOTER, INVERTER). The circuit structure is then modified according to the intrinsic characteristics of NML logic and the clock zones layout. Three important operations are sequentially executed on the input graph: NML Fan-Out Management, NML Reconvergent Paths Balance and NML Wire Cross Minimization. The elaborated graph is then used as input for the NML PHYSICAL MAPPING.

A. NML Fan-Out Management

Similarly to CMOS circuits also in NML technology a limitation on the fan-out of each logic gates holds. At the current stage of technology evolution, this is a critical constraint for NML, which makes the NML Fan-Out Management an extremely important stage. The fan-out limitation in NML is mainly related to the clock zones layout and to the physical space occupied by the wires. Particularly, with a layout of clock zones organized in parallel stripes, there is a limitation in the length of vertical magnetic wires to avoid propagation errors [13]. Moreover, every wire is made by magnets so there must be enough room to allow their physical placement. For this reasons the graph is iteratively scanned and, if nodes with more than N fan-out connections are found, additional levels and nodes are added until every node as a fan-out smaller or equal to N. The fan-out limit (N) is a parameter that must be given as input to the algorithm. While the fan-out is a parameter that can be set freely by the user, its the user responsibility to choose the correct value according to the

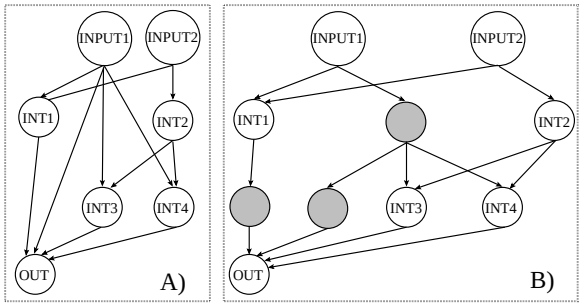


Fig. 4. Reconvergent Paths Balance. A) Graph before leveling. B) Graph after wire block insertion and wire block sharing.

circuits constraints. This additional nodes will be physically represented by NML wires. This process can sometime generate subtrees without leafs, i.e. subtrees composed only by additional blocks created by the fan-out limitation routine that do not connect any logic gate. A specific subroutine eliminates all this dead subtrees. Figure 3.A shows a generic graph before the application of the fan-out limitation routine. Figure 3.B shows the results of the algorithm if the fan-out limit is set to 2. Clearly, the stricter is the limitation on the fan-out, the bigger is the number of additional levels.

B. NML Reconvergent Paths Balance

In a graph two paths are called *reconvergent* if they diverge from and reconverge to the same blocks. For example in Figure 4.A all paths start from inputs I1 and I2 and converge to the output O. This is a common situation in an electronic circuit but in NML it presents further challenges due to the intrinsic pipelined behavior. As mentioned in section II, the delay in terms of clock cycles of a wire depends on its layout (problem known as “layout=timing” [3][10]). To be synchronized, signals at the input of a logic gate must have the same length and therefore the same delay. As a consequence, reconvergent paths must be balanced, i.e. they must have the same number of nodes. To obtain this result, starting from an unbalanced graph (Figure 4.A), intermediate nodes that physically represent NML wires are inserted in the graph to balance all the paths. The RECONVERGENT PATHS BALANCE routine can sometimes add duplicated wire nodes, i.e. wires nodes that have the same father but different children. A subroutine of WIRE SHARING merges these nodes together reducing the complexity and optimizing the graph. The results of the RECONVERGENT PATHS BALANCE algorithm are shown in Figure 4.B. Every path is balanced granting perfect signal synchronization.

C. NML Wire Cross Minimization

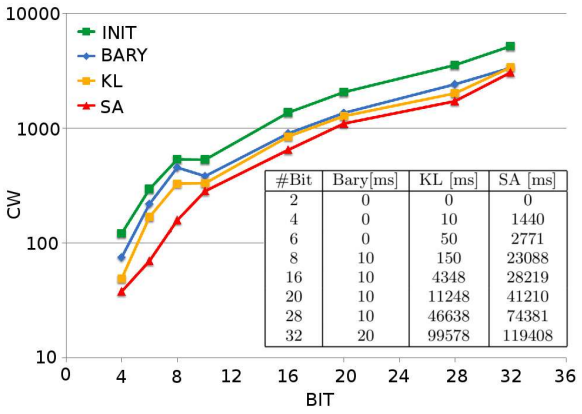
One of the most characteristic features of NML and QCA technology (from which NML is derived as a particular implementation) is that both logic gates and interconnections are on the same plane. Up to now there are no experimental evidences that it will be possible to route wires on different layers. As a consequence in this technology a particular block is available, the cross-wire (see figure 1.D), that allows the crossing of

two wires on the same plane. Even though it enables the routing of signals on only one layer, the circuit layout must be optimized to reduce the number of cross-wires required, therefore reducing the wasted area. Different techniques can be used for minimizing wire crosses. Previous works [8][9] take advantage of the clocking constraints to propose simple methods for cell placement, belonging both to the analytical and stochastic families. To the analytical set belongs the *Barycenter method* and the *Fan-out Tolerance Duplication*, while *Simulated Annealing* is a stochastic method. Not only we implemented and modified all these algorithms, but we further enriched the range of possibilities available to the user with a partitioning algorithm exploiting and improving *Kernighan-Lin* heuristic.

The *NML-Barycenter* method is a very simple technique that changes the position of each node on the graph trying to place every node directly above the nodes to which it is connected. This algorithm is in this work coupled with a fan-out duplication technique, where nodes are duplicated to reduce the fan-out of each node, improving the performance of the *NML-Barycenter* method. *NML-Barycenter* method was chosen for its simplicity, and it is normally used to obtain a first simplification of the circuit and reduce the execution time of more complex techniques. The *NML-Kernighan-Lin* algorithm iteratively divides the graph in sub parts trying to minimize the cut, i.e. the number of edges between one partition and the others. It is an heuristic techniques that requires many steps to reach an optimum solution. The *NML Simulated Annealing* is a stochastic technique where the position of each node of the graph is changed randomly until the optimum solution. Due to its stochastic nature its efficiency heavily depends on the parameters used and, generally, it requires a long time to reach the solution.

1) Comparison among crosswires minimization techniques: Figure 5 shows the comparison of crosswires minimization techniques we implemented. The test circuit is a N bit ripple carry adder, with N varying from 4 to 32 bits. Some important conclusions can be derived from Figure 5. The number of crosswires increases with the growing number of bits, however all the techniques allow a consistent reduction of the number of crosswires. In case of a 32 bit adder the number of crosswires is reduced from 5000 to about 3000. The NML BARYCENTER algorithm is the simplest but it offers the worst performance. The NML KERNIGHAN-LIN MODIFIED algorithm offers better performance but not so good as the NML SIMULATED ANNEALING. The advantage of NML SIMULATED ANNEALING is greater with a small number of bits but it is reduced for an high number of bits, where the results of all algorithm tend to saturate.

However if we look at the time requested to obtain these results shown in the figure 5 inset with tabulated data: Barycenter method (Bary), Kernigan-Lin modified method (KL), Simulated Annealing method (SA). We observe that the relative small improvement offered by KERNIGHAN LIN and SIMULATED ANNEALING algorithms has an high cost in terms of computational time. While the application of BARYCENTER



#Bit	Bary [ms]	KL [ms]	SA [ms]
2	0	0	0
4	0	10	1440
6	0	50	2771
8	10	150	23088
16	10	4348	28219
20	10	11248	41210
28	10	46638	74381
32	20	99578	119408

Fig. 5. Wire cross reduction comparison of different algorithms. A multi bit adder is used as benchmark. Inset with table: Execution time for wire cross minimization algorithms applied to a variable bit number Ripple Carry Adder.

method to a the 32 bit case takes only 32 ms, the other techniques require 1-2 minutes. It is clear from this data that KERNIGHAN LIN and SIMULATED ANNEALING algorithm must be used only on relative small circuits and only when a very high level of optimization is requested.

V. NML PHYSICAL MAPPING

In the PHYSICAL MAPPING process the graph is translated into the circuit layout. The general flow chart is shown in Figure 2.C. After every node is mapped to its correspondent logic gate, it is PLACED in the circuit. A GLOBAL ROUTING phase follows where an approximated routing is performed and the position of each gate is changed trying to obtain the minimum area solution. When the position of each logic gate is defined a DETAILED ROUTING among the blocks flows.

A. Placement

As a first step every node is mapped to its correspondent logic gate. The placement of these logic gates follows the structure of the graph. As shown in Figure 6.A, every node of the graph is placed, row by row. Each node corresponds to a logic gate with a different area. Logic gates are placed without any optimization (Figure 6.B), with the each base side aligned at the beginning of the row. After this phase it is possible to evaluate the minimum area requested for the placement of the circuit. Finally the position of each gate is shifted using a simple Barycenter approach (Figure 6.C). The final position of each gate will be decided at the end of the NML GLOBAL ROUTING phase (see later). It is important to underline that the placement of the circuit relies on the clock structure described in [3], where clock zones are organized in parallel strips. Thought other organizations are theoretically possible, this layout is chosen here because it is the only one currently experimentally demonstrated and because it is well suited for combinational circuits with a dataflow structure.

B. NML Global Routing

To obtain the definitive positions of logic gates a NML GLOBAL ROUTING phase is required. The aim of this part

of the algorithm is to find the optimal shift of the position of each logic gate. The reason is to reduce the length of the interconnection wires, obtaining therefore the global minimum area of the circuit. The flow diagram is shown in Figure 7.A. It is an iterative process where for each couple of rows i) logic gates are shifted, ii) interconnection wires are routed and iii) the interconnections area is evaluated. Every wire is based on magnets that must have a minimum separation between them, so the area occupied by the wires can be easily evaluated. The structure of the circuit can be divided in rows, corresponding to the graph nodes that represent logic gates, separated by channels dedicated to interconnections. The aim of the NML GLOBAL ROUTING phase is to obtain the minimum width for each routing channel. The results obtained at the end of this phase are two: 1) The final position of each logic gate and 2) the final position of the pin, the input and output points in the routing channel. Figure 7.B shows an example of two rows before the NML GLOBAL ROUTING phase, while Figure 7.C shows the situation after the minimum is reached. Gates position is shifted and the length of the routing channel is greatly reduced. From Figure 7.C the input and output points of the routing channel can be observed.

C. NML Channel Routing

Now that the final position of each gate is defined, wires can be routed and the final circuit layout obtained. This part of the algorithm is called NML CHANNEL ROUTING. It takes as input a channel with input and output signal positions fixed (Figure 8.A) and places interconnections wires. In CMOS technology routing is normally performed on Manhattan Grids, with interconnections made by horizontal and vertical segments perpendicular among them. This solution is not well suited for NML technology. The reason lies in the particular clock zones layout [3] and on the limited number of elements that can be cascaded avoiding errors in the signal propagation [13]. The consequences are that signals can propagate without problems in the direction perpendicular to clock zones strips, but not so in the other direction. The propagation in this second direction follows a stair-like pattern, as shown by the dashed line in figure 1.E. Signals, then, can only move in oblique (up or down). For this reason a Manhattan Grids approach cannot be used. The approach that we have chosen is called mini-swap [15], and it is shown in Figure 8.B. Interconnection wires are routed in a oblique way. When two nets cross, then they are physically mapped with a crosswire block (Figure 8.C and

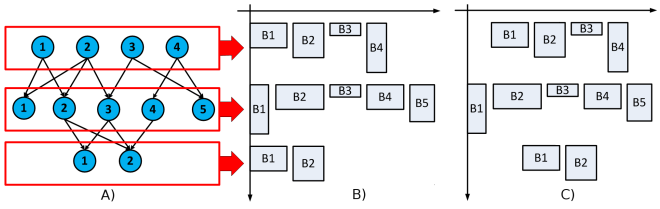


Fig. 6. A-B) Seed row placement for maximum width evaluation. C) Barycentered placement.

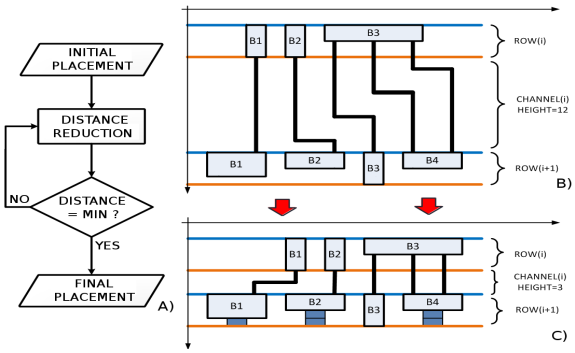


Fig. 7. A) NML Global Routing flow diagram. B) Unoptimized placement. C) Optimized placement.

1.D). Finally the oblique interconnections are mapped in the real circuit using magnets. Figure 8 shows a detail of the final resulting circuit. The "stair-like" signal propagation, which is typical of this technology, is evident. The maximum number of magnets that can be cascaded in one direction or in the other direction is a parameter that can be set by the user, and it will affect the final layout and area of the circuit.

D. Results

Figure 9 shows an example of circuit layout obtained at the end of the whole algorithm. It is a 6 bit ripple carry adder with a total area of 30x5 um^2, while the size of each magnet is 60x90 nm^2. Simulated annealing is normally used as wire cross minimization technique. However, at the increase of circuits complexity, the barycenter technique is previously applied to simplify the circuit and to reduce simulated annealing execution time.

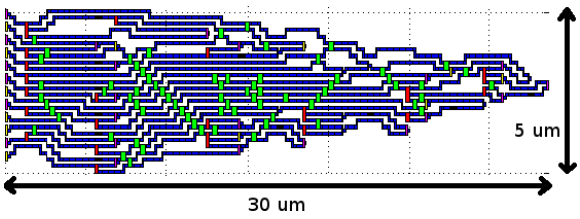


Fig. 9. Layout of a 6 bit Ripple Carry Adder.

Table I shows the results obtained on some of the ISCAS85 benchmark circuits. An intermediate step is required to use ISCAS85 with this tool, because the original circuit must be re-synthesized, using external tools, on a gate library compatible with NML technology (Majority Voter, AND, OR; inverter). The generation of the layout requires a time (PT in Table I shows the tool runtimes) between few milliseconds and few minutes depending on the circuit complexity (# cells). The circuit area (CA) is of few um^2 for the c17 which is made by 7 logic gates, while the most complex (c6288) has an area of nearly $1mm^2$, as it is composed by nearly 350000 logic gates. It is interesting to point out from the last column of Table I that there is a lot of wasted area, since the area occupied by logic gates (% OCC) is at maximum 30% of

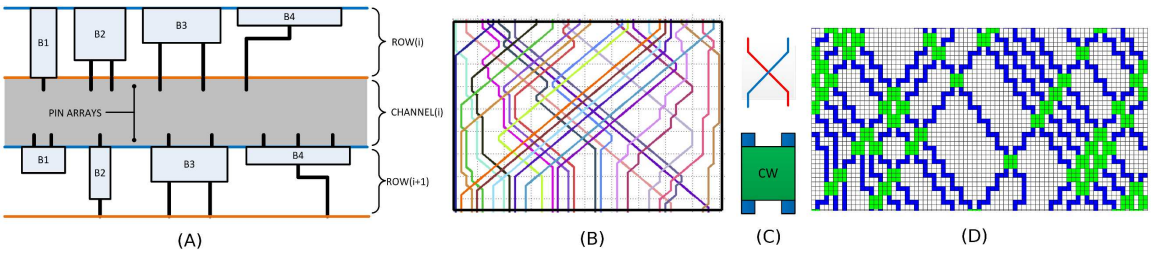


Fig. 8. A) Pins for channel definition. B) Mini Swap model for channel routing. C) Crosswire mapping. D) Physical mapping of interconnections.

the total circuit area. The impact of wasted area is more evident comparing the results with the area obtained through synthesis on a CMOS 22nm technology. The area of the CMOS implementation is lower increasing circuits complexity. This is caused by the fact that, up to now, NML circuits are constrained on only one layer, while CMOS can use up to 11 additional layers for interconnections.

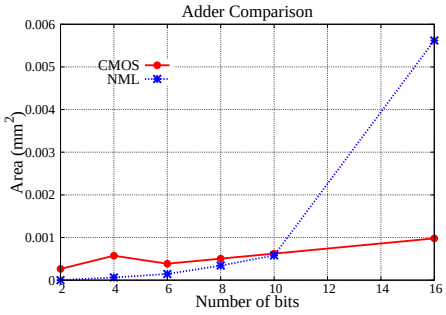


Fig. 10. Comparison for RCA between NML and CMOS 22 nm in terms of area. Data for CMOS are estimated, starting from Synopsys synthesis on a 90 nm technology node, using ITRS Roadmap values.

More interesting is here to focus on a single RCA if increasing number of bit. Figure 10 shows the RCA area for both technologies. Data for CMOS are obtained through synthesis on a liberty library file. NML data on area are a result of this work. The layout generator is very effective with small word widths: for RCA the resulting area is smaller up to 10 bit parallelism when compared to CMOS. In other words, NML area occupation converge to CMOS only for circuits of small-medium complexity (from 1000 to 5,000 cells). If NML cannot be competitive in terms of frequency (around 100-200MHz are the expected frequency ranges for NML while CMOS circuits can reach a frequency of many GHz), area is competitive up to medium size circuits due to the single layer available. This suggests it is worth inspecting the future evolutions of NML and prompts to move toward a multilayer organization of interconnects.

TABLE I
RESULTS OF ISCAS85 SAMPLES.

circuit	PT [ms]	#cells	CA [mm^2]	%OCC	CMOS22 [mm^2]
c17	88	7	0.0000035	30	0.000049
c880	5919	5753	0.00641	32	0.004018
c1908	54724	6941	0.0107	32	0.005005
c2670	125818	19350	0.0292	29	0.00684
c6288	661931	349083	0.94	25	0.02815

VI. CONCLUSIONS AND FUTURE WORK

As a total novelty in the literature scenario we have presented a Place&Route engine for combinational NML circuits which is integrated in our tool for nanotechnology design and exploration. We can automatically generate the optimized layout of NML combinational circuits of any complexity taking into account all the technological constraints currently known. Comparisons with CMOS technology show competitiveness in terms of area up to medium complexity circuits.

We are now working in two directions: i) adding a partitioning and floorplanning to handle complex and hierarchical circuits; ii) extending the algorithm to sequential circuits to allow the design of any kind of architecture.

REFERENCES

[1] G. Csaba and W. Porod. Simulation of Filed Coupled Computing Architectures based on Magnetic Dot Arrays. *J. of Comp. El., Kluwer,*, 1:87–91, 2002.

[2] M. Vacca and al. Asynchronous Solutions for Nano-Magnetic Logic Circuits. *ACM J. on Emerging Tech. in Comp. Systems*, 7(4), December 2011.

[3] M. Niemier and al. Nanomagnet logic: progress toward system-level integration. *J. Phys.: Condens. Matter*, 23:34, November 2011.

[4] M. Vacca and al. Majority Voter Full Characterization for Nanomagnet Logic Circuits. *IEEE T. on Nanotechnology*, 11(5), September 2012.

[5] Marco Ottavi and al. HDLQ: A HDL Environment for QCA Design. *ACM J. on Emerging Tech. in Comp. Systems*, 2(4):243–261, 2006.

[6] T. Teodosio and L. Sousa. A tool for the automatic layout generation of QCA combinational circuits. In *IEEE Norchip*, 2007.

[7] S. Frache and al. ToPoliNano: Nanoarchitectures Design Made Real. *IEEE NANOARCH*, 2012.

[8] R. Ravichandran and al. Partitioning and placement for buildable QCA circuits . *DAC*, 1, 2005.

[9] W.J. Chung and al. Node duplication and routing algorithms for quantum-dot cellular automata circuits . *IEE Proc. on Circ., Dev. and Sys.*, 153(5), 2006.

[10] M. Graziano, M. Vacca, A. Chiolerio, and M. Zamboni. A NCL-HDL Snake-Clock Based Magnetic QCA Architecture. *IEEE Transaction on Nanotechnology*, (10):DOI:10.1109/TNANO.2011.2118229.

[11] M. Graziano, M. Vacca, D. Blua, and M. Zamboni. Asynchrony in Quantum-Dot Cellular Automata Nanocomputation: Elixir or Poison? *IEEE Design & Test of Computers*, 2011.

[12] M. Vacca, S. Frache, M. Graziano, and M. Zamboni. ToPoliNano: A synthesis and simulation tool for NML circuits . *IEEE International Conference on Nanotechnology*, August 2012.

[13] G. Csaba and W. Porod. Behavior of Nanomagnet Logic in the Presence of Thermal Noise. In *International Workshop on Computational Electronics*, pages 1–4, Pisa, Italy, 2010. IEEE.

[14] M. Awais, M. Vacca, M. Graziano, and G. Masera. Quantum dot Cellular Automata Check Node Implementation for LDPC Decoders . *IEEE Transaction on Nanotechnology*, 2013.

[15] D. Wang. Novel Routing Schemes for IC Layout Part I: Two-Layer Channel Routing . *Design Automation Conference*, 1991.