Nanomagnetic Logic Microprocessor: Hierarchical Power Model

Article *in* IEEE Transactions on Very Large Scale Integration (VLSI) Systems · August 2013

DOI: 10.1109/TVLSI.2012.2211903

3 authors:

Marco Vacca
Politecnico di Torino
63 PUBLICATIONS **588** CITATIONS
SEE PROFILE

Mariagrazia Graziano
Politecnico di Torino
162 PUBLICATIONS **1,076** CITATIONS
SEE PROFILE

Maurizio Zamboni
Politecnico di Torino
170 PUBLICATIONS **1,192** CITATIONS
SEE PROFILE

NanoMagnetic Logic Microprocessor Hierarchical Power Model

Marco Vacca, Mariagrazia Graziano *Member IEEE*, Maurizio Zamboni

***Abstract*—The interest on emerging nanotechnologies has been recently focused on NanoMagnetic Logic (NML), which has unique appealing features. NML circuits have a very low power consumption and, due to their magnetic nature, they maintain the information safely stored even without power supply. The nature of these circuits is highly different from the CMOS one. As a consequence, to better understand NML logic, complex circuits and not only simple gates must be designed. This constraint calls for a new design and simulation methodology. It should efficiently encompass manifold properties: 1) being based on commonly used hardware description language (HDL) in order to easily manage complexity and hierarchy; 2) maintaining a clear link with physical characteristics 3) modeling performance aspects like speed and power, together with logic behavior.**

In this contribution we present a VHDL behavioral model for NML circuits, which allows to evaluate not only logic behavior but also power dissipation. It is based on a technological solution called "snake-clock". We demonstrate this model on a case study which offers the right variety of internal substructures to test the method: a four bit microprocessor designed using asynchronous logic. The model enables a hierarchical bottom-up evaluation of the processor logic behavior, area and power dissipation, which we evaluated using as benchmark a division algorithm. Results highlight the flexibility and the efficiency of this model, and the remarkable improvements that it brings to the analysis of NML circuits.

***Index Terms*—QCA, NML, power dissipation, power model, VHDL, microprocessor, NCL.**

I. Introduction

Quantum dot Cellular Automata [1] are a recent implementation of the original cellular automata principle [2]. The computation is due to the interaction of neighbor identical cells, instead of relying on conduction. The expected main advantage is to overcome the predicted criticalities of forthcoming CMOS circuits [3]. Aside from the original proposal [4][5], two are currently the appealing implementations: molecular QCA [6][7], built using complex molecules with many oxide-reduction centers, and magnetic QCA [8][9], based on single domain nanomagnets, with only two stable magnetization states. Molecular QCA are interesting mainly for the high speed that in theory they can reach [10], but they are currently far from the technology reality [7][11]. However the magnetic QCA or NanoMagnetic Logic (NML) allows the fabrication of a fully magnetic circuit. This implies very low power consumption and the ability to mix computation and storage functions [12].

Many QCA circuits [1] have been proposed: A good summary is in [13]. Most of them are combinational circuits like arithmetic blocks and multiplexers [14][15]. More complex circuits like sequential ones [16] and memories [13][17] have also been presented, and an example of a very simple microprocessor was discussed in [18]. These works have the merit of exploring the adaptability of the QCA idea to the computation principles normally used with CMOS technology. However they lack of a clear link with the technological fabrication process, and we think that this link is mandatory to understand whether the QCA technology can be a reliable CMOS substitute. We thus focus on the magnetic implementation, because it is feasible using current technology, it has been experimentally demonstrated [19], and it is then possible to take into account its physical characteristics (backgrounds on NML behavior is in section II).

Authors are with the VLSI LAb, Dipartimento di Elettronica e Telecomunicazioni, Politecnico di Torino, Corso Duca degli Abruzzi, 24 Torino, I10129 Italy

A peculiar aspect of NML is that in order to propagate a signal without errors [20] an external field is applied which drives the cell in an intermediate unstable state lowering the potential barrier between the two stable magnetization values. When the field is removed, magnets reorder themselves in an antiferromagnetic or ferromagnetic arrangement depending on the input cell, and the magnets relative placement. This magnetic field is called "clock". If too many magnets are cascaded, there might be propagation errors when the magnetic field is removed. As a consequence circuits must be organized into small areas, called clock zones, composed by a limited number of nanomagnets, each related to a different clock signal. In order to drive the information flow in every direction, a complex layout of the clock zones is required. In the same time, the clock layout has to take into account technology limitations. A proposal relying on a complex structure has been made in [19], while in the solution we identified [21][22], a feasible and more simple clock wire topology is presented.

The necessity of a clock signal and the related creation of clocking zones generate a critical issue named "layout=timing" [19]. The propagation delay of a NML wire depends on the number of clock zones through which it is routed, i.e. it could be thought as a pipelined interconnect in standard CMOS technology. In other words, as a wire is routed through N clock zones, then $N/3$ clock cycles will be necessary to move the information from the beginning to the end of that wire. This means that a logic gate with more than one input may have signals reaching their values in incoherent time moments, unless they are routed through the same number of clock zones. This constraint could be unbearable in case of circuits with many gates and elaborated connection. A possible solution is the Null Convention Logic^{TM} (NCL) [23], which can solve this major problem due to its asynchronous nature [24][25]. A NCL gate computes a new value only when every signal is asserted at its inputs. Therefore, even in presence of substantially different propagation delays among inputs, the logic behavior is correct. A general QCA implementation is shown in [26], while we proposed a magnetic QCA implementation of NCL in [21] [22]. The results in this paper use as a starting point the achievements in [22], which we summarize in section II.

From the methodology point of view, QCADesigner [27] was exploited to analyze the QCA circuit behavior. This tool has been specifically implemented to design and simulate this type of circuits, but it is unable to handle magnetic circuits and it is problematic when complex circuits are tackled. For NML circuits, physical finite element simulators like Comsol Multiphysics [28], OOMMF [29] or NMAG [30] could be adopted, but again they should be considered for the validation and verification phase, because they are not able to handle complex architectures. Only high level description languages like VHDL give the flexibility and the abstraction capabilities necessary when more than simple gates are under design and analysis. This was already and successfully proposed in [31] and in [32] for general QCA. We relied on this idea when adapting our magnetic implementation on our technology proposal for the clock delivering in [22]. However, our version includes physical constraints related to the magnetic case, and allows us to verify with a great accuracy the behavior of a generic and complex NML circuit. In this work, as an absolute novelty in the literature, we have integrated in the previously mentioned VHDL description a power model which estimates different power contributions: the power dissipated by the nanomagnets and the power dissipated by the clock wires. This model, which is the focus of the work, and is discussed in section IV, is hierarchical and highly customizable with many parameters, so that a realistic evaluation can be reached. To test this model we have used as benchmark a simple four bit NCL microprocessor briefly described in section III. This microprocessor, that we have deeply investigated in [33] and [34], is used here only as a case study to demonstrate the VHDL model adaptability to a variety of logic structures as those included in a processor (combinational, sequential, memories, feedbacks,...). The model generality allows an easy implementation in case of other architectures (for example [35]) that might be specifically proposed for the NML logic (e.g. systolic arrays or PLA based structures [36]). Results shown in section V, highlight the flexibility of this methodology, which allows to extract useful information for both architectural and technological design of NML circuits.

II. Understanding NML

A. Background

In NML the basic cell is a single domain nanomagnet with an high aspect ratio which leads to an high shape anisotropy. This characteristic lets nanomagnets have only two stable magnetizations, which represent the two logic values 0 and 1 (Figure 1.A). The magnetization vector is parallel to the long side, the so-called easy axis. These two stable states are separated by an high potential barrier.

As a consequence, the cell switching from one state to the other requires the use of an external field, to lower the potential barrier. Typically this is an intense magnetic field applied along the short side of the nanomagnets, the so-called hard axis. When this field is applied, nanomagnets are forced into an unstable state, with the magnetization directed along the hard axis. As soon as the magnetic field is removed, nanomagnets reorganize themselves in an antiferromagnetic or ferromagnetic order (Figure 2.A). The alternate behavior of this field (applied/removed) is the reason why it is called "clock" and the signal generating such a field can be represented by a periodic waveform like the one in the top of the Figure 2.C. The magnetic field is generated by a current flowing through a wire placed under the magnets plane, a ferrite yoke is used to confine the magnetic flux lines [37] (Figure 2.B). When the magnetic field is removed a reordering phase takes place; however errors could occur if too many cells are cascaded. The maximum number of nanomagnets that can be aligned without errors, experimentally verified [19], is around 10-20 cells. Simulations show that considering the thermal noise, this number could be reduced to 5 [38]. Therefore, a complex circuit should be divided into small areas, called clock zones, composed by a limited number of cells.

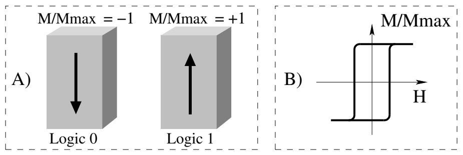


Fig. 1. NanoMagnetic Logic. A) Rectangular shaped nanomagnets are used to represent the digital information. B) Hysteresis loop of a single domain rectangular shaped magnet.

Recently, new NML topologies and technological implementations have been introduced. They use multilayer structured material to reach higher speed and lower the power dissipation due to the clock generation network. In [39] a multiferroic magnet, composed by a layer of piezoelectric material and a layer of a magnetic material with high magnetostriction was used to obtain ultra low power consumption and an electrical clock system. In [40] a Magneto-Tunnel-Junction was used as a NML cell. With this particular structure the NML dot can be reset using spin-torque coupling, allowing low power operation and an easier clock wire routing.

B. Snake-clock

According to our proposal in [21][22] we have chosen to apply first of all a 3-phases clock, which was the first solution proposed that allows the implementation of feedback signals. A 2-phases clock implementation is also considered according to [41] and discussed herein. In the case of the 3-phases clock, to every clock zone the correspondent clock signal, shown in Figure 2.C, is applied. Each clock signal has the same waveform, just a phase difference of 120 degrees differentiates the signals. Figure 2.D shows the information propagation process. During the first time step (TIME STEP 1) the third clock zone is in the SWITCH phase, which means that the magnetic field has been previously applied and now removed. Nanomagnets are therefore in an unstable situation. They see on their left a nanomagnet, the last element of the second clock

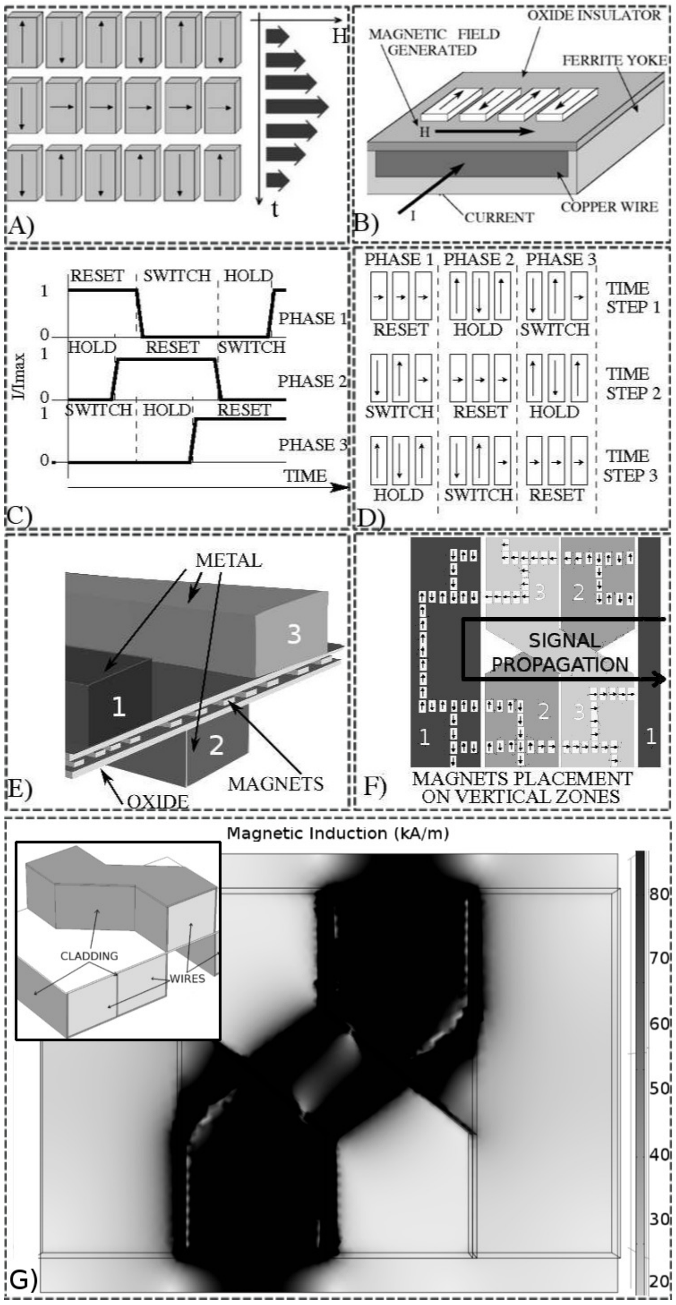


Fig. 2. Clock system. A) Magnets are forced in the Reset state when the magnetic field is high. B) Magnetic field is generated through a current flowing in a wire placed under the nanomagnets plane. A ferrite yoke is used for a better confinement of the magnetic field. C) Clock signals concerning the three phases; an overlap between two phases is requested for a more reliable behavior. D) Signal propagation: logic organization of nanomagnets in time and space following the clock signal sequence (RESET, SWITCH and HOLD status). E) 3D view of a section of three clock wires, one for each phase: nanomagnets are sandwiched between two oxide layers, two clock wires are above the magnets plane and one is below. F) Top 2D view of a portion of clock wires and nanomagnets organized so that they can propagate information according to the phase sequence 1-2-3-1-2-3... . G) Comsol Multiphysics FEM simulation of the twisted clock wires for the Snake-clock scheme.

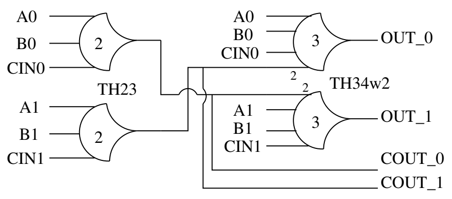
zone, that is in a stable (HOLD) state. In the same time their right nanomagnet, the first element of the first clock zone, is in the unstable state (RESET). In this situation they start to reorder from left to right. This happens because the magnets in the second clock zone act like an input for the third one, while the first clock zone is in a state which has no influence on the neighbors. During the second time step (TIME STEP 2), the above mentioned behavior is repeated with a forward shift of one clock zone, as the switching zone is now the first one. Finally, during the third time step (TIME STEP 3) another data forwarding from left to right occurs, as the switching zone is now the second one. To sum up, the signal moves through the circuit from left to right.

A further important detail should be added. When a magnetic field is applied to the magnets, a finite time (few hundreds of picoseconds) is required for their switching. If magnets are forced in the RESET state simultaneously when magnets on their left start to switch, an unwanted signal back-propagation may happen. As a consequence clock signals must be overlapped [22]: Before removing the field from a zone, say A, it must be applied to the next zone, say B, as the waveforms in Figure 2.C. In this way, when nanomagnets in zone A start to reorder, they are not influenced by magnets in zone B which are already in an unstable state. This assures a correct information propagation. A side effect of this 3-phases overlapped clock is the immunity to errors due to bad field confinement in neighbor clock zones. When the magnetic field is applied at a clock zone, the zone on their left is in the SWITCH state. If some of the switching magnets are influenced by the neighbor magnetic field they will not switch until the RESET field is removed. The signal propagation will start N magnets before the beginning of the clock zone, where N is the number of magnets influenced by the neighbor magnetic field, but it will propagate correctly. Considering now the clock zone on the right of the zone where the magnetic field is applied, due to bad confinement some of its magnets can be forced in the RESET state. This is not a problem because clock signals are overlapped so there will be a moment where both clock zones will be anyhow in the RESET state.

We have proposed in [22] a structure named “snake-clock” which is summarized in Figure 2.E,F. Wires can be placed above or under the plane of the nanomagnets, using oxide layers as separation planes (Figure 2.E). According to the snake-clock scheme one wire is straight and placed above the plane and delivers the clock for phase 1; one of the other two is placed on the same plane (phase 3), and the remaining runs below it (phase 2), but these two are “virtually” twisted, i.e they are routed using a zig-zag style (Figure 2.F). Magnets are placed where wires are vertical in the 2D sketch plane (not in the crossing zones) as in the example in Figure 2.F. This clock structure achieves two important results: it allows information propagation in every direction (Figure 2.F, from left to right in the bottom row and from right to left in the upper row) and we believe that it can be fabricated with current technology. Figure 2.G shows an accurate FEM simulation of the magnetic field generated by one of the clock wires. The confinement of the magnetic field is clearly quite good. Magnets placed near the border of neighbor clock zones can sense the influence of the magnetic field, but, as stated before, with this clock system this does not represent a problem, as we demonstrated in [22] with FEM simulations. The only remaining constraint is that no magnets must be placed where the wires are twisted.

C. NCL

The use of a multi-phase clock leads to the previously mentioned layout=timing problem. As we proposed in [42], [22] Every clock zone is equivalent to a D-latch, where at every clock cycle the output copies the input value. Therefore, the propagation delay of a NML wire depends on the number of clock zones the wire passes through. If different inputs signals of a logic gate arrive after an unequal number of clock cycles, then the circuit will not work correctly, as incoherent data will be evaluated.



TH23 = F = A + (BC + BF + CF)+ F(A+B+C+D)
TH34w2 = F=ABC+ABD+ACD+BCD + F(A+B+C+D)

Fig. 3. NCL circuit example: full adder. Every signal is coded using two bits. Logic gates are TH23 (symbol 2) and TH34w2 (symbol 3)

An alternative possible solution is using the Null Convention Logic^{TM} (NCL, [23]), an asynchronous delay-insensitive logic. Signals are coded using two bits, and they can assume two different states: NULL state when they are at the same time 0, and DATA state which represents the logic value (01 means logic 0 and 10 means logic 1). The circuit passes from NULL to DATA only when all the inputs change from NULL to DATA and it maintains its status until at least one input is in the DATA state. Before a new data can be accepted from a logic gate, every input must reach the NULL state. Only at this point a new cycle can start. This assures the circuit operations also in presence of a considerable difference in the propagation delay among the inputs.

Figure 3 shows a full adder implemented with two different NCL gates (called TH23 and TH34w2) and with encoded signals. The circuit is split into two specular parts, each of them calculating one of two encoded output bits. This solution has been adapted to general QCA in [26]. Further explanations on this logic are not reported here for the sake of brevity while details can be found in [22][33]. In the following, brief explanation on how we model this logic applied to NML logic case is given to understand the power model and its use in the microprocessor benchmark we designed.

D. VHDL behavioral model

Following the example in [31][32], VHDL can be used to describe all the NCL gates [23] once we have adapted them to the magnetic implementation and to our snake-clock organization. An example is shown in Figure 4.A, which represents the model of a NCL THxor0 gate. In Figure 4.B the VHDL simulation results are depicted. Since the use of a clock system gives to NML circuits a pipelined behavior, a NML gate can be modeled using a register for each clock zone. Each register is driven by the correspondent clock phase signal, shown in the Figure 4.A bottom-right detail. The Majority Voter (MV, the basic logic block in NML) is modeled with an ideal logic gate without delay. The timing behavior of this structure, in terms of clock cycles, is the one expected by the snake-clock structure. The exact duration of the pulse in each phase is not related to the logic function, but to the technology. It depends, in fact, on the number of magnets present in each phase zone and on the switching time during the magnets reordering. Both these quantities depend on technology choices.

The simulation results of this circuit are shown in Figure 4.B. The output changes from 0 to 1 only when the input A and the input B change from 0 to 1, according to the gate equation. The output changes with the delay of one clock cycle, because the number of clock zones the input signals pass through is equal to three $T_{ck} = TH_{ck1} + TH_{ck2} + TH_{ck3}$. The output remains stable until at least one of the inputs remains to 1, but it goes to the logic value 0 when all signals go to 0. The cycle subsequently can restart.

E. Physical implementation example

Figure 4.C shows the implementation of the THxor0 using the snake-clock, where the crosses at the bottom show the prohibited zones where clock wires are twisted (details discussed in figure 2 F and G. This was the first clock structure proposed which allows feedback signal propagation and that at the same time is expected to be feasible with current technology (wires are not particularly small, may require a non-manhattan shaping that is available in a normal advanced CMOS tehcnology and may require a magnetic material cladding, that recalls the damascene process in current VLSI technology). Recently a new clock structure was proposed [41]. It uses only 2 phases based on parallel wires. The propagation direction is determined by the first element, which is a magnet with a different shape. If this magnet is placed on the left of the clock zone, signals propagate from left to right. If it is placed on the right signals propagate from right to left. Figure 4.D shows the implementation of the THxor0 using the 2-phase clock. The 2-phase clock is easier to fabricate and circuits are more compact, because there is no wasted space due to wire twisting. However, from the VHDL modeling point of view, both systems can be easily simulated changing few configuration parameters (see section IV). Both the clock structures need relatively long interconnections (10-15 magnets) especially vertical interconnections. If one considers all the theoretical constraints [38] this might lead to not working circuits. However, accurate micromagnetic simulations show that these constraints are too strict. Figure 4.E shows a vertical wire made by 10 magnets, where helper blocks (lateral magnets) are used to avoid propagation errors [43]. OOMMF [29] simulations are shown in Figure 4.F (initial state) and in Figure 4.G (final state) where a correct circuit behavior is evident. This means that circuits can work also considering lines longer than expected, as recent experiments have demonstrated [44]. Both the 2- and 3-phases clock styles are then feasible, and have both been considered here for the power model demonstration.

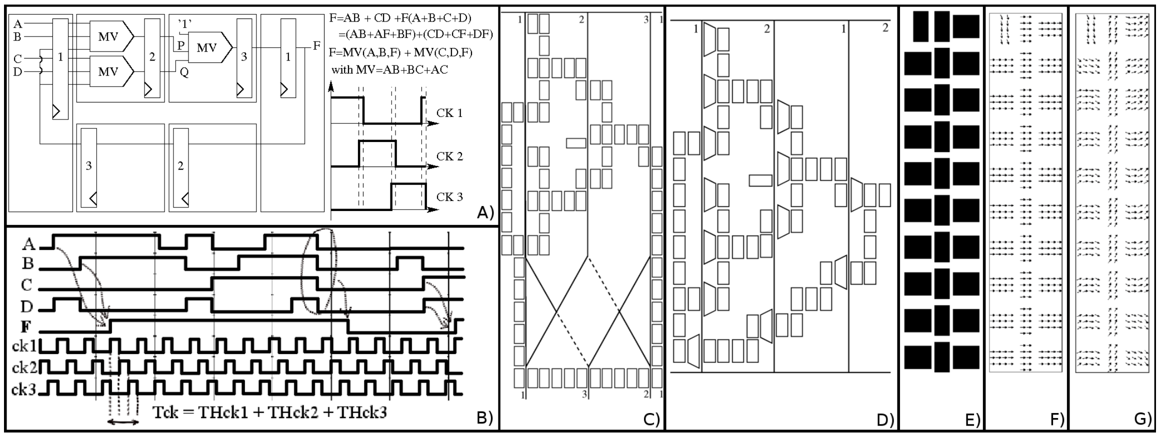


Fig. 4. A) THxor0 VHDL behavioral model. The upper-right detail shows the logic functions of the gate and the majority voter (MV), while the bottom-right detail shows the clock signals applied to each register. B) THxor0 simulation results. It is possible to observe the transition of the gate from F=0 to F=1 when the logic equation is satisfied. C) NCL THxor0 implemented using a 3-phase clock. D) NCL THxor0 implemented using a 2-phase clock. E) Vertical wire geometry: helper cell are used to improve the signal propagation length. F) Vertical wire OOMMF simulation: RESET state. G) Vertical wire OOMMF simulation: HOLD state.

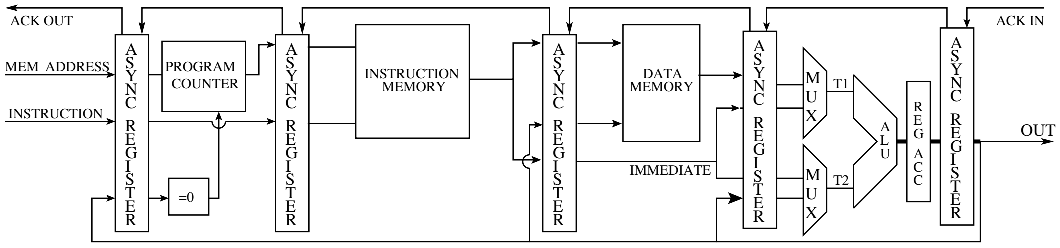


Fig. 5. Microprocessor architecture organized in four main stages separated by asynchronous registers.

III. NCL Magnetic Microprocessor

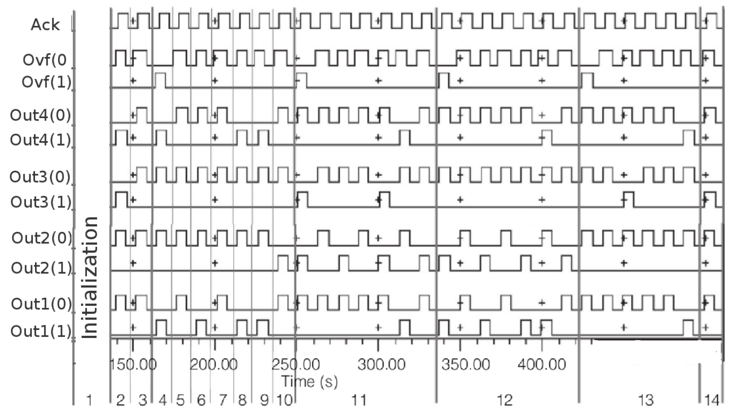
To evaluate the effectiveness of the power model a realistic architecture was implemented: A complete 4 bit microprocessor. The processor architecture is simple but it can handle many types of operations: arithmetic/logic, memory read and write, different kinds of jump. The microprocessor was chosen because it involves all types of logic circuits, e.g. combinational, sequential and of storage type, and therefore it represents the ideal test-bench for the NML technology performance evaluation. Even though it has been demonstrated that if implemented using NML an architecture like a processor suffers problems due to long feedbacks [33], for the purpose of this work it represents the perfect benchmark as its variety of structures is an opportunity to show the flexibility of the power model. Though not the focus of this work, it is worth giving a brief description for understanding the structure and the complexity of the system. A complete and detailed discussion on the internal components and architectural choices is in [33][34].

The overall architecture of the microprocessor is shown in Figure 5. It is based on four main blocks: A program counter which generates the address for the instruction memory and which manages jump instructions, a parallel memory for program memorization, a parallel memory for data storing and finally a data-path block for the arithmetic/logic operations.

Since the microprocessor is based on Null Convention Logic, which is asynchronous, it requires a communication protocol between two blocks. The protocol is implemented using asynchronous registers, as shown in Figure 5. These registers, differently from synchronous ones, do not have memory properties, as they only generate the acknowledgement ACK signals necessary as time reference. A register accepts a new data only when it receives the ACK signal from the successive one. At the same time it generates the ACK signal for the previous register, only when all the inputs are active. In this way the delay-insensitivity is assured.

The microprocessor is organized in four asynchronous pipe stages. In the first stage the NCL program counter generates the address for the instruction memory. Beyond the normal sequential address generation, in case of a jump instruction, when the jump enable signal is high, then an external value is loaded. This value, due to the 4 bit multiplexer, can be chosen from the value stored in the accumulator register or from an external source. The second and third pipe stages contain the two memories. The instruction memory is a parallel memory composed by 16 words of 14 bits, while the data memory is a serial memory of 4 words of 4 bits. The last pipe stage is the microprocessor data-path. It is composed by an arithmetic/logic unit which handles four operations: addition, subtraction, logic AND and logic OR. Two multiplexers are placed at the ALU inputs for selecting the source operands.

A comparator block makes the conditional jumps possible upon zero condition, i.e. when the previous operation result of the ALU is zero. The comparator output is connected back to the jump enable of the program counter. The final data-path block is the accumulator register, which stores results of the



Algorithm

1. Load instruction and initialize
2. Set Counter to 0 and store it
3. Store 1100 (12)
4. Subtract 0011 (3)
5. Store result
6. Add 1 to Counter
7. Store new Counter value
8. Load subtraction result
9. If result is 0 stop, else goto10
10. Subtract 0011 (3)
11. Execute same steps as 5 to 10
12. Execute same steps as 5 to 10
13. Execute same steps as 5 to 10
14. Subtract result is 0, stop
    final output result is 0100 (4)

Fig. 6. Division algorithm simulation (Modelsim). In the right detail the algorithm is reported.

ALU operation.

The instruction set of the microprocessor is composed of 4 arithmetic/logic operations which can be performed between different sources: between the data memory and an immediate value, between the data memory and the accumulator or between an immediate value and the accumulator. Data memory read and write instructions are also possible. Finally two types of jump instructions are feasible, unconditional jump or a conditional, if zero, jump. Both of them can be performed directly, to an address represented by the immediate value, or indirectly to a value stored in the data memory. A total of 28 instructions are then at disposal for executing a program. An example is detailed in the following.

The simulation results for a division algorithm, obtained using Modelsim [45], are shown in the Figure 6. The numbers in the bottom line of the simulation diagram show the different phases of the division algorithm execution. The phases are detailed in the figure right box. No details are discussed here for sake of brevity: The result after phase 14 is 0100 (01-10-01-01 considering the NCL convention) i.e. the number 4, the correct result of the 12/3 division. Further details on the microprocessor logic behavior can be found in [33] and [34].

IV. Power model

Power losses in NML circuits depend on two main components: The power dissipated by nanomagnets during their switching phase and the power dissipated by clock wires during the reset field generation. In the following, we are going to describe for both contributions the dissipation cause, the design parameters, the model proposed and its VHDL description. To generate the current which flow through the clock wires a simple circuit composed by few transistors and logic gates can be used [46]. We do not consider this power dissipation in our model because it is expected to be much smaller than the other power contributions.

A. Causes and parameters

During its switching a *nanomagnet* follows an hysteresis cycle (Figure 1.B). The area of this hysteresis cycle is proportional to the energy spent for the switching. This energy must be supplied to the nanomagnet by the source which generates the magnetic field, and this is normally dissipated in form of heat. In this work we have chosen a value of $30K_BT$ for the energy dissipated by the nanomagnet during the switch according to the discussion presented in [47]. This value is the average power consumption due to magnet switching. The energy dissipated by the nanomagnets depends on their number. The *number of nanomagnets* related to each clock phase is different, as it depends on the layout and the circuit complexity. As a consequence during each clock phase a different power dissipation occurs. However, due to the NML circuits structure, during an entire clock cycle, composed of 3 clock phases, every nanomagnet switches. As a consequence, the total magnets power consumption during one clock cycle is equal to the total *number of nanomagnets* multiplied for the value of $Energy\_mag = 30K_BT$.

The second contribution is the power dissipated by *clock wires*. This contribute can be separated in two components, the power dissipated due to the Joule effect and the power stored in the wire inductance. The power dissipated by Joule effect represents the main contribution, mainly because a high value of current is necessary to generate a magnetic field strong enough to force a reset. Furthermore, as a pessimistic approximation, no mechanism to recover the energy stored in the related inductance at every clock cycle is considered. Therefore it represents another source of energy loss. The clock frequency in this circuit is relatively low (100 MHz), so the contribution of the inductance is not relevant. However, it is considered because this model can be used also for recently proposed versions of NML technologies [39][40], which work at higher frequencies, or for molecular QCA which work at very high frequencies (1 THz). The energy dissipated by the clock depends on the *length of the wire*, which is a function of the circuit area, affected by the circuit complexity and layout.

B. Model

Both power contributions depend on the number of nanomagnets, on how they are organized in each phase and on the circuit layout. At present moment layout algorithms for NML circuits are not available in the literature. Just a simple algorithm has been implemented for general QCA [48], but it is not suited for NML logic. We are currently working on a

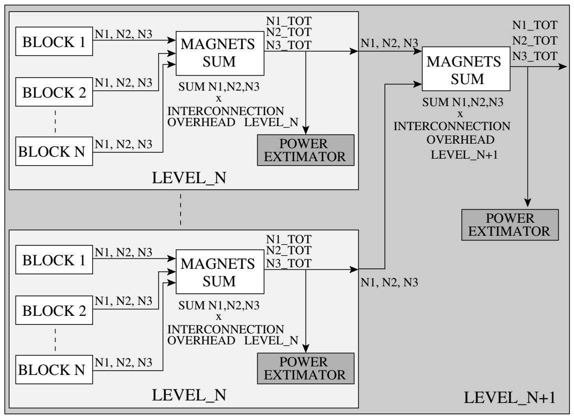

BLOCK 1
BLOCK 2
BLOCK N
N1, N2, N3
MAGNETS SUM
N1_TOT
N2_TOT
N3_TOT
SUM N1,N2,N3 x INTERCONNECTION OVERHEAD LEVEL_N
POWER EXTIMATOR
LEVEL_N
SUM N1,N2,N3 x INTERCONNECTION OVERHEAD LEVEL_N+1
LEVEL_N+1

Fig. 7. Hierarchical model for estimating the number of nanomagnets in a NML circuit and its power dissipation due to nanomagnets and to clock wires. N1, N2, N3 are the number of magnets (for each clock zone) of each lower level logic block. N1_TOT, N2_TOT, N3_TOT are the total number of magnets of the logic level considered.

tool [49], [**?**][50] for designing and exploring circuits based on emerging technologies. However, as the idea in this work was to estimate the expected power dissipation, we propose here a method that does not require the complete physical design. So, at the time of writing, we propose a method to estimate the number of magnets taking into account the circuit complexity, and, as a consequence, to evaluate the power consumption.

The model is based on five keypoints (Figure 7):

- It is **embedded** in the architecture description: Each block includes not only the logic sub-blocks, but also the functions for evaluating the two power contributions.
- It is **hierarchical**: A block of level *N* uses data on the number of magnets from sub-blocks of level *N-1*, and generates information to be propagated to the higher *N+1* hierarchical level.
- Given a block *i* in the architecture, a **power estimator** evaluates power consumption for current block *i* as a function of the number of nanomagnets in block *i*.
- In the architecture of level *N*, a **nanomagnet sum** is enabled using the number of nanomagnets of all the included blocks of level *N-1* as input. This sum is extended to each clock zone.
- An **overhead** factor is used to take into account the routing complexity; if sub-blocks have a total sum of magnets equal to *M*, the connection among them could require an additional number of magnets; this *overhead* is estimated and is multiplied by *M*. The value of this factor is different for each hierarchy level.

The model is written in VHDL and it uses many parameters to assure the maximum flexibility. In the following the model will be detailed by including portions of VHDL code of the Full Adder (Figure 3). Considering a generic logic level *N+1* (see Figure 7), every internal block of level *N* has three output signals, which are the total number of nanomagnets of that block separated for each phase. The number of magnets is of integer type, however to avoid type conversion, they are represented using real numbers. For example, if the block of level *N+1* is the Full Adder, one of the sub-blocks is the *TH23* NCL gate, which *entity* declaration is in the following. The three output port of type "real" can be observed.

```
entity th23 is
  port (a, b, c : in  std_logic;
        y : out std_logic;
        ck1, ck2, ck3 : in  std_logic;
        N1, N2, N3 : out real := 0.0);
end th23;
```

The logic function (Figure 3) requires three inputs *a, b, c* and one output *y*. Three clock signals are used internally for the registers which model the phase dependency, as mentioned in section II-D
(Figure 4.A). *N1* represents the total number of nanomagnets clocked by phase 1, while *N2* and *N3* are the number of nanomagnets clocked by phases 2 and 3, respectively. The Full Adder has two instances of both TH23 and TH34w2 gates, which have a similar entity declaration. Delay blocks are used to simulate the extra area required for signals routing.

The output values holding the number of nanomagnets for each gate instances become the inputs of a block, called *MAGNETS SUM*. This calculates the sum of all the nanomagnets of the sub-blocks, phase by phase. These values are then multiplied for a factor *interc_overhead* which represents the interconnections overhead. This factor, defined as a parameter in a package file, is different for each hierarchical level and represents the area overhead due to interconnections. The new values are finally used as output of this *SUM* block, which VHDL code is in the following.

```
entity magnets_sum is
  generic(interc_overhead: real := 1.0);
  port(ck: in std_logic;
      f1, f2, f3: in  real_vector;
      N1, N2, N3: out real := 0.0);
end magnets_sum;
architecture behavioural of magnets_sum is
begin
 compute: process (f1, f2, f3)
  variable f1_int: real_vector(f1'Length-1 downto 0)
                                           := (others => 0.0);
  ... STATEMENT REPEATED FOR PHASES 2 AND 3
  variable sum_f1, sum_f2, sum_f3: real := 0.0;
  variable sum_tot_f1, sum_tot_f2, sum_tot_f3: real := 0.0;
 begin
  f1_int := f1;
  f2_int := f2;
  f3_int := f3;
  sum_f1 := 0.0;
   for i in 0 to f1'Length-1 loop
    sum_f1 := sum_f1 + f1_int(i);
   end loop;
  sum_tot_f1 := sum_f1 * INTERC_OVERHEAD;
  N1 <= sum_tot_f1;
  ... STATEMENT REPEATED FOR PHASES 2 AND 3
 end process;
end behavioural;
```

The *SUM* output signals carries the estimated total number of magnets of the current block, in this case the Full Adder. These output signals are exported to the hierarchically higher level but they are also used as inputs for a further element, the *Power Estimator*, which locally finds the power dissipation of this block of level *N+1*. This block is inside every component, so that during the simulation it can calculate not only the total power of the entire circuit, but also the contribution of every component. The explanation of this element is reported after its VHDL code, which is here partially simplified for the sake of brevity. Every calculation is repeated for each phase and here reported only for the first one. Several parameters are here used: they are defined in a VHDL package and reported in table I and explained in the following description of the model.

```
entity power_extimator is
port(N1, N2, N3: in real;
     ck1, ck2, ck3: in std_logic);
end power;

architecture behavioural of power is
  ... signals definition, initialization....(skipped)
begin
  -- effective circuit area in terms of number of magnets
  Area_eff_1<= N1 * Wasted_Space;
  -- clock wire lenght in terms of number of magnets
  Lwire_1 <= Area_eff_1 / Width_zone_mag;
  -- effective clock wires lenght in meters
  Lwire_eff_1 <= Lwire_1 * (h + vert_space)
                  * h_zone_sep * Wire_curves;
  -- effective wire section
  Swire       <= (Width_zone - Wire_sep) * Wire_thick;
  -- wire resistance
  Rwire_1     <= Resistivity * (Lwire_eff_1/Swire);
  Log_1        <= (4.0*Lwire_eff_1) / Width_zone;
  -- wire inductance
  Ind_Wire_1  <= Lwire_eff_1 * 2.0e-7*(LOG(Log_1)-1.0);

  Process_power_1: process (ck1) -- PHASE 1
  begin
   if ck1 = '1' then
    -- clock power losses due to joule effect
    P_joule_1 <= Rwire_1 * I_max * I_max;
    -- clock power losses due to inductance charging
    P_ind_1 <= ((Ind_Wire_1*I_max*I_max)/2.0)/(T_clock/3.0);
    -- power losses due to nanomagnets switching
    P_mag_1   <= Mag_power * N1;
   elsif ck1 = '0' then
    P_ck_RI_1 <= 0.0;
    P_ck_LI_1 <= 0.0;
    P_mag_1   <= 0.0;
   end if;
  end process;

  ... STATEMENTS REPEATED FOR PHASE 2 AND PHASE 3

  -- FINAL SUM OF ALL THE CONTRIBUTIONS
  P_joule_tot <= P_joule_1 + P_joule_2 + P_joule_3;
  P_ind_tot   <= P_ind_1 + P_ind_2 + P_ind_3;
  P_mag_tot   <= P_mag_1 + P_mag_2 + P_mag_3;

end behavioural;
```

The first power contribution, the average power dissipated by the nanomagnets, is calculated as in equation 1

$$Mag\_power = \sum_{i=1,2,3} N_i \cdot \frac{Energy\_mag}{T\_clock} \tag{1}$$

multiplying the number of nanomagnets (e.g. N1 for phase 1) for the energy value of $Energy\_mag$, defined in the VHDL package (see table I), and dividing it by the clock period. This value is separately calculated for each phase, but also the average total value related to the entire clock period is evaluated.

The calculation of the power dissipated by clock wires is due to contributions of resistance and inductance. The first effect is calculated by the equation 2,

$$P\_joule\_tot = \sum_{i=1,2,3} Rwire\_i \cdot I\_max^2 \tag{2}$$

where the current I_max is chosen equal to 1mA [51]. The resistance of the wire *Rwire_i* for each phase is given by the well known equation 3, as function of the total wire length *Lwire_eff_i*, the wire section *Swire*, and the metal resistivity.

$$Rwire\_1 = Resistivity \cdot \frac{Lwire\_eff\_1}{Swire} \tag{3}$$

The wire section is calculated

$$Swire = (Width\_zone - Wire\_sep) \cdot Wire\_thick \tag{4}$$

as the product between the effective wire width (*Width_zone - Wire_sep*) and its thickness (*Wire_thick*).

The power dissipated by the inductance is calculated as in equation 5,

$$P\_ind\_tot = \sum_{i=1,2,3} \frac{\frac{1}{2} \cdot Ind_{Wire_i} \cdot I\_max^2}{\frac{1}{3} T\_clock} \tag{5}$$

where T_clock is the total clock period defined in the package. The wire inductance *Ind_Wire_i* is only approximated as if the wire were straight and alone. No mutual inductance among neighbor wires is considered. For the purpose of this model this approximation is sufficient, because it gives the order of magnitude of the inductance. Moreover this approximation is partially compensated from our first assumption, where all the energy used to charge the inductance is lost. It is calculated according to equation 6.

$$\begin{aligned} Ind\_Wire\_i = Lwire\_eff\_i \cdot 2e{-}7 \cdot \\ \cdot\, ln\left(\frac{4 * Lwire\_eff\_i}{(Width\_zone - Wire\_sep)} - 1\right) \end{aligned} \tag{6}$$

Due to the flexibility of this model, the inductance calculation can be easily improved substituting this equation with a more precise one.

Both resistance and inductance require the estimation of the wire length *Lwire_eff_i*, which is the central core of the power estimator block. This is done starting from the number of nanomagnets (N1 for phase 1, N2 for phase 2,...), assuming, that the number of nanomagnets and the area of the circuit are related. The wire width depends on the width of the clock zone, defined as a parameter in the package (*Width_zone* in table I). In this case 700nm was chosen which is equivalent to the width of 10 nanomagnets (*Width_zone_mag*). The estimation of the wire length parameter is done as shown in the Figure 8, using different parameters to takes into account the wasted area. The clock structure, in the snake clock case, is based on the three phases pattern (one straight and the other two twisted) repeated M times [22], depending on the circuit area and shape. Wires of the same phase (with the same color

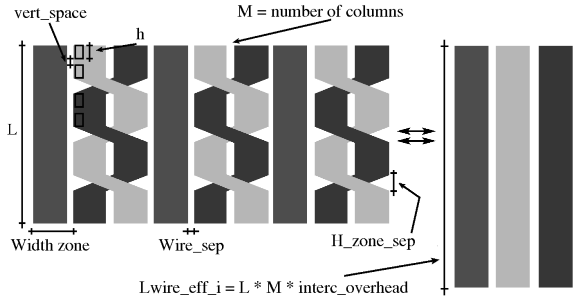


Fig. 8. Wire length calculation for the three phases. Numeric constant used for the calculation are indicated. Wires of same color are connected serially, so they can be approximated as one straight wire. A factor *Wire_curves* is used to takes into account wire angles overhead.

in Figure 8) are connected serially outside the perimeter of

TABLE I
PARAMETERS AND CONSTANT DEFINED IN THE VHDL PACKAGE USED IN THE NML POWER MODEL.

Parameter name	Default value	Explanation
h_zone	6.0e-7	Height of clock zone (meters)
h_zone_sep	2.0	Vertical separation between zones, where no magnets are allowed.
Width_zone	7.0e-7	Width of a clock zone (meters)
Width_zone_mag	10.0	Width of a clock zone in terms of magnets
Wire_thick	6.0e-7	Clock wire thickness (meters)
Wire_sep	2.0e-8	Space between clock wires (meters)
Wasted_Space	2.0	Relative area used by components (magnets + separation spaces)
h	1.0e-7	Height of a nanomagnet (meters)
W	5.0e-8	Width of a nanomagnet (meters)
vert_space	2.0e-8	Vertical separation space between magnets (meters)
oriz_space	2.0e-8	Horizontal separation space between magnets (meters)
Wire_curves	1.1	Overhead due to path wires for connecting different wires pieces
OV1_logic_gate_level	1.1	Interconnect overhead inside a logic gate
OV2_intermediate_level	2.0	Interc. overhead in small clusters of logic gates (full_adder, mux, registers..)
OV3_logic_block_level	1.5	Interconn. overhead inside functional element (alu, counter,....)
OV4_top_level	1.2	Interconnect overhead due to logic blocks interconnection
I_max	1.0e-3	Maximum current flowing in clock wires (Ampere)
clock_overlap	11.0	Clock overlap percentage
Resistivity	1.78e-8	Clock wire resistivity
T_clock	9.0e-9	Clock period (seconds)
Energy_mag	30.0*KT	Energy associated to a single magnet switch (with T=300 and K=Boltzmann constant)
N_mag_mv	5.0	Number of magnets in a Majority Voter
N_mag_inv	7.0	Number of magnets in an Inverter

the circuit, where no magnets are present. As a consequence they are equivalent to a single clock wire with an equivalent length *Lwire_eff_i*, which is approximately M times bigger than the original length *L* given by one side of the circuit rectangular perimeter. The wire length evaluation estimates the total number of magnets, which can be also considered as the area of the circuit expressed in terms of number of magnets. This value is multiplied for a constant (*Wasted_space*) that considers the separation area among magnets that are part of different gates (see eq. 7 for a single phase). This area is equal to that of one magnet and it is necessary to avoid crosstalk.

$$Area\_eff\_i = Ni \cdot Wasted\_Space \tag{7}$$

This area is expressed in terms of number of nanomagnets. Dividing this area for the width of the clock zone, we obtain the length of the clock wire (equation 8), in terms of number of magnets.

$$Lwire\_i = \frac{Area\_eff\_i}{Width\_zone\_mag} \tag{8}$$

To obtain the length of the wire expressed in meters, *Lwire_eff_i* must be multiplied for the sum of the physical height of the magnet *h* and the vertical separation between magnets *vert_space*. Two other constants are introduced to obtain a better evaluation of the wires length: the height of the vertical separation between clock zones *h_zone_sep* and a factor used to take into account the wires curves *Wire_curves*.

$$\begin{aligned} Lwire\_eff_i = Lwire\_i \cdot (h + vert\_space) \cdot \\ \cdot\, h\_zone\_sep \cdot Wire\_curves \end{aligned} \tag{9}$$

If *h_zone_sep* is set to 1 no vertical separation between clock zones is considered. In this way it is possible to estimate the wire length considering a 2-phase clock. To conclude the description it is worth noticing that this method for calculating the number of magnets and the power dissipation is repeated at every logic level, where the output signals of the lower level blocks become the input signals of the higher level SUM block. In this way the number of nanomagnets is hierarchically propagated starting from the lowest possible level, the logic gates, to the highest possible level. For the elementary logic gates the number of nanomagnets is defined as a constant in the package. In our case we defined this number for the Majority Voter and the inverter. The effectiveness of this model relies on the value chosen for the constant used to take into account overheads (interconnections overhead, wire curves, ...). We have extrapolated this constants starting from NML theory, taking into account all the physical and layout constraints actually known.

V. RESULTS

The model was applied to the whole microprocessor analyzing the main power contributions due to its internal sub-blocks and considering two possible clock structures. A preliminary comment is worth doing before commenting the results: A validation of these data is at the moment almost impracticable. From the experimental point of view a complete clocked NML system has not been implemented yet, with the exception of small circuits like in [44], were a full adder was fabricated and tested. Simulators able to take into account both nanomagnets and wire power dissipation are not available, especially if complex systems should be handled. The only available simulators which calculates energy consumption during nanomagnets switching are OOMMF or NMAG, which indeed was used to refine our model; still it is not enough to validate a complex system like the one under test. However, we believe that this model gives reasonable values, as it is based on practical assumptions, and it is flexible enough to span among a set of realistic results which variability depend on design criteria and technology choices. Moreover, we believe that this model is particularly useful to compare different designs choices

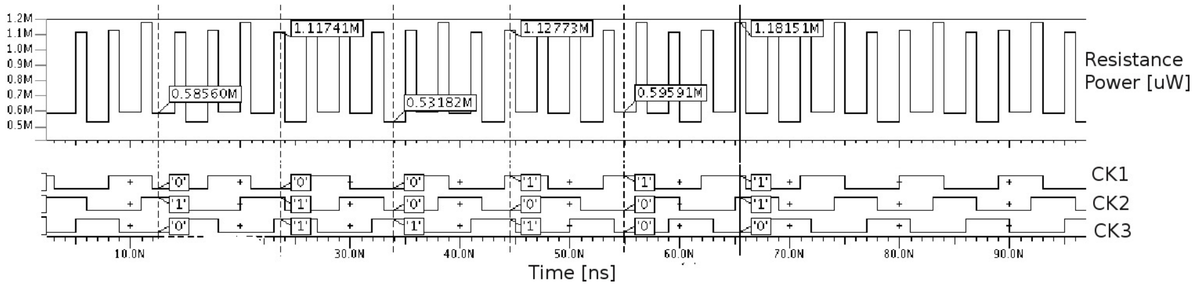


Fig. 9. Power dissipation simulation results for clock wires resistance. Peaks correspond to clock signal overlap, minimum correspond to the presence of one single active clock phase.

[33][34]. Due to the flexibility of this model it is quite easy to compare also different clock structures simply changing the configuration parameters. Table II shows the performance of the circuit with different clock structures and wire widths. The total number of nanomagnets of the entire microprocessor correspond to an approximative (safely estimated) area of $4.5 \cdot 10^3 \mu m^2$, which is remarkably smaller than the area of an equivalent processor in CMOS technology. Changing the width of the clock wire, and therefore the width of the clock zone, does not change the number of nanomagnets, because the number of logic gates in the circuits is the same. Choosing a 2-phase clock slightly reduces the number of nanomagnets because the length of the feedback path is shorter.

TABLE II
NANOMAGNETS COUNT AND AVERAGE POWER DISSIPATED DUE TO NANOMAGNETS AND TO CLOCK WIRES (RESISTANCE AND INDUCTANCE CONTRIBUTION). RESULTS ARE FOR THE WHOLE PROCESSOR CONSIDERING DIFFERENT CLOCK ZONE LAYOUTS AND SCHEMES.

	Total Number Magnets	Power Magnets [μW]	Power Resistance [μW]	Power Inductance [μW]
3-phase snake clock Wire width = 700nm	743000	3.875	662.5	5.125
3-phase snake clock Wire width = 350nm	743000	3.875	2727	11.55
2-phase clock Wire width = 700nm	701722	3.660	330	2.38
2-phase clock Wire width = 350nm	701722	3.660	1363	5.45

The same table shows the average power dissipation due to clock wires. The power lost due to the inductance is quite low since the circuit works at relatively low frequency (100 MHz). However with more complex NML technologies like [40] and [39], the frequencies obtainable are higher so in that case the inductance power losses will have a greater impact. The power lost due to the Joule effect is indeed very high, and can potentially overcome all the advantages of this technology. This can be improved using advanced clock solutions like [51] or recently introduced NML technologies [40][39]. However the most important information that can be obtained from Table II is the huge influence on the power dissipation of choices made at the logic level. For example, reducing the wire width from 10 magnets to 5 magnets to improve the reliability of the signal propagation increases the resistance of the wire. As a consequence, the power lost due to Joule effect is greatly increased. At the same time, switching to a 2-phase clock reduces by a half the power dissipation, because there is not a wasted area among clock zones due to the wire twisting. The total length of the wire is reduce by a half. This comes at the cost of a less regular fabrication step due to the different magnets shapes.

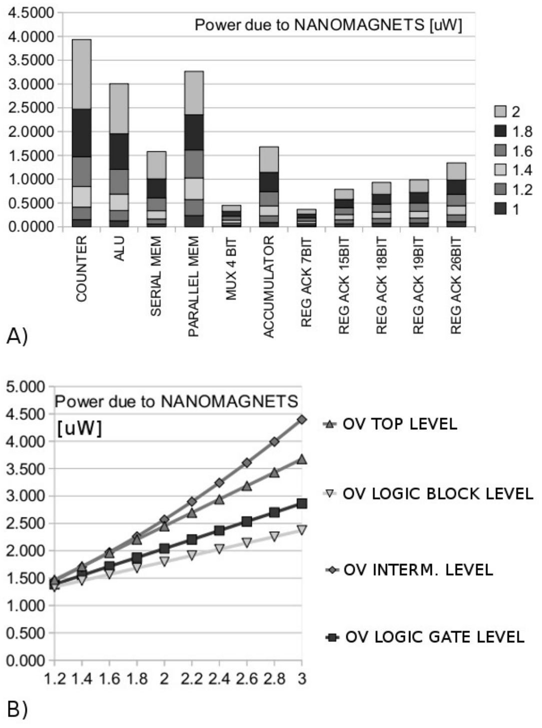


Fig. 10. Microprocessor power dissipation [μW] due to nanomagnets. A) Values are obtained by linearly and uniformly changing all overhead values due to interconnects from 1 (no influence from interconnects) to 2 (interconnect double the area of the block). obtained by varying one overhead parameter at a time from 1.2 to 3, while the others are kept constant to 1. B) Power is obtained by varying one overhead parameter at a time from 1.2 to 3, while the others are kept constant to 1.

Values shown in table II, represent the average power dissipation only, however the model allows to detail the clock power variations in time, as shown in Figure 9 for the snake clock. Only the power lost for the Joule effect is reported. For nanomagnets and inductance the waveform is the same but the absolute values change. As expected the power has a periodic behavior. Three are the visible minimum values which correspond to the time sub-periods where only one clock phase is active. Furthermore three are the peak values related to the clock phases overlap, when two clock wires are simultaneously active. As previously explained, the model evaluates the length of the 3 clock wires independently, starting from the number of nanomagnets of each single phase, which are in general different. Therefore the length of the clock wire of each phase is slightly different, and this explains why the three minimum and the three peaks differ.

Further interesting results, that outline the flexibility of this model, are in Figure 10, where the overhead parameters have been varied with respect to the default values in the snake clock case. Only the power consumption due to magnets is shown. The other two contributions have the same behavior but different absolute values. This depends on the model: Every power components is evaluated starting from the circuit area estimation. In Figure 10.A details for the main processor components are shown as a function of a linear and uniform variation of the overhead parameters: from 1 (no overhead) to 2 (wires double the core area of each sub-block). As expected a linear increment of the three power contributions occurs. It is interesting to note that the counter has an higher power consumption than the other components. This is due to the fact that it is based on arithmetic blocks and has an internal feedback (it is basically a finite state machine). As a consequence, the impact of wires of nanomagnets is much more evident, and must be taken into account when designing complex systems (e.g. avoid feedback at high hierarchical levels [34]). Correct architectural choices are then suggested by this result: This again underlines the importance of this type of analysis, which main goal is to link architecture with technology, so that bidirectional improvements, and not only constraints, can be derived.

Further results are in the Figure 10.B, where the three power dissipation components are shown for the whole microprocessor. In this case several simulations where performed each varying only one overhead parameter at a time from 1.2 to 3, keeping other values constant to 1. What is interesting to note is the impact of each parameter. As expected OV1_LOGIC_GATE_LEVEL has the smallest impact, as at the logic gate level the structures are very simple and, consequently, small is the wire overhead. Slightly worse is the OV3_LOGIC_BLOCK_LEVEL contribution. The surprise comes from the OV2_INTERMEDIATE_LEVEL term, which causes the biggest power increment and its impact is more than linear. It is even bigger than the top level contribution OV4_TOP_LEVEL. In the intermediate level the NCL gates are interconnected, and the layout is not very regular, therefore the interconnection overhead is high. Again an interesting outcome useful to refine the architecture design and guidelines to optimize the intermediate level layout is derived from this analysis.

VI. Conclusions

A unprecedented model was developed for estimating area and power of a complex Nano Magnetic Logic architecture. The model is flexible and parametric and was integrated in a hierarchical VHDL description of the circuit. It takes into account the logic itself, the possible clocking structure, the overhead due to magnetic interconnects. It allows to reckon not only the average power value but also its transient behavior.

In order to challenge the model, a four bit NML Microprocessor was designed based on three important features which are important outcomes of previous works: a feasible clock structure, Null Convention Logic adapted to NML, and a behavioral model which describes NML circuits using VHDL. The benchmark complexity allowed to explore the solution space of parameters configuration in order to understand the effects on power dissipation of architectural blocks, of interconnections overhead and of clock structures.

Results are encouraging. The microprocessor power dissipation is lower than expected, confirming the most promising feature of this technology. Our analysis, as it takes into account a realistic scenario and practical parameters, points out quantitatively and not only qualitatively that the impact of clock wires risks to nullify the outstanding performance of this technology in terms of power. The methodology helps on the one hand the designers to understand the critical points to be addressed when designing a NML architecture, and, on the other hand, the technologist to analyze the impact of technological choices on circuits performance.

References

[1] C.S. Lent, P.D. Tougaw, W. Porod, and G.H. Bernstein. Quantum cellular automata. *Nanotechnology*, 4:49–57, 1993.

[2] J.L. Schiff. *Cellular Automata: A Discrete View of the World*. Wiley & Sons, 2007.

[3] A. Pulimeno, M. Graziano, and G. Piccinini. Udsm trends comparison: From technology roadmap to ultrasparc niagara2. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 20(7):1341 –1346, july 2012.

[4] R.K. Kummamuru, A.O. Orlov, R. Ramasubramaniam, C.S. Lent, G.H. Bernstein, and G.L. Snider. Operation of a quantum-dot cellular automata (qca) shift register and analysis of errors. *IEEE Trans. On Electron Devices*, 50:1906, 2003.

[5] A.I. Csurgay, W. Porod, and C.S. Lent. Signal processing with near-neighborcoupled time-varying quantum-dot arrays. *IEEE Transaction On Circuits and Systems*, 47(8):1212–1223, 2000.

[6] U. Lu and C.S. Lent. Theoretical study of molecular quantum-dot cellular automata. *Journal of Computational Electronics - Springer*, 4:115–118, 2005.

[7] A. Pulimeno, M. Graziano, D. Demarchi, and G. Piccinini. Towards a molecular qca wire: Simulation of write-in and read-out systems. *Solid-State Electronics, Elsevier*, 1:7, 2012.

[8] Alexandra Imre. *Experimental study of nanomagnets for Quantum-dot cellular automata(MQCA)logic applications*. PhD thesis, University of Notre Dame, Notre Dame, Indiana, December 2005.

[9] G. Csaba and W. Porod. Simulation of filed coupled computing architectures based on magnetic dot arrays. *Journal of Computational Electronics, Kluwer,*, 1:87–91, 2002.

[10] M. Liu C. Lent Y. Lu. Molecular electronics - from structure to circuit dynamics. In *Sixth IEEE Conference on Nanotechnology*, pages 62–65, Cincinnati-Ohio, USA, 2006. IEEE.

[11] P. Motto, A. Dimonte, I. Rattalino, D. Demarchi, G. Piccinini, and P. Civera. Nanogap structures for molecular nanoelectronics. *Nanoscale Research Letters*, 7(113):7, 2012.

[12] G. Csaba, P. Lugli, and W. Porod. Power dissipation in nanomagnetic logic devices. In *International Conference on Nanotechnology*, pages 346–348, Munic, Germany, 2004. IEEE.

[13] J. Huang and F. Lombardi. *Design and Test of Digital Circuits by Quantum-Dot Cellular Automata*. Artech H. Pub., Boston, 2007.

[14] H. Cho and E.E. Swartzlander. Adder and multiplier design in quantum-dot cellular automata. *IEEE Transaction On Computers*, 58(6), 2009.

[15] S. Hashemi, M.R. Azghadi, and A. Zakerolhosseini. A novel qca multiplexer design. *Intl. Symp. on Telecommunications*, 2008.

[16] X. Yang, L. Cai, and X. Zhao. Low power dual-edge triggered flip-flop structure in quantum dot cellular automata. *IET Electronics Letters*, 46(12), 2010.

[17] B. Taskin and B. Hong. Improving line-based qca memory cell design through dual phase clocking. *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, 16(12), 2008.

[18] K. Walus, M. Mazur, G. Schulhof, and G. A. Jullien. Simple 4-bit processor based on quantum-dot cellular automata (qca). In *International Conference on Application-Specific Systems, Architecture and Processors, ASAP*, pages 288–293, Samos, Greece, 2005. IEEE.

[19] A. Imre, G. Csabaa, G.H. Bernstein, W. Porod, and V. Metlushkob. Investigation of shape-dependent switching of coupled nanomagnets. *Superlattices and Microstructures*, 34:513–518, 2003.

[20] M.T. Niemier, X.S. Hu, M. Alam, G. Bernstein, M. Putney W. Porod, and J. DeAngelis. Clocking structures and power analysis for nanomagnet-based logic devices. In *International Symposium on Low Power Electronics and Design*, pages 26–31, Portland, USA, 2007. IEEE.

[21] M. Graziano, A. Chiolerio, and M. Zamboni. A technology aware magnetic qca ncl-hdl architecture. In *International Conference on Nanotechnology*, pages 763–766, Genova, Italy, 2009. IEEE.

[22] M. Graziano, M. Vacca, A. Chiolerio, and M.Zamboni. An ncl-hdl snake-clock-based magnetic qca architecture. *Nanotechnology, IEEE Transactions on*, 10(5):1141 –1149, sept. 2011.

[23] K.M. Fant and S.A. Brandt. Null convention logic^{TM}, a complete and consistent logic for asynchronous digital circuit synthesis. In *International Conference on Application Specific Systems*, pages 261–273, Chicago-Illinois, USA, 1996. IEEE.

[24] M. Choi, Z. Patitz, and N. Park. Efficient and robust delay-insensitive qca (quantum dot cellular automata) design. In *International Symposium on Defect and Fault-Tolerance in VLSI Systems*, pages 80–88, Arlington-Washington DC, USA, 2006. IEEE.

[25] M. Choi, Z. Patitz, B. Jin, F. Tao, and N. Park. Designing layout-timing independent quantum-dot cellular automata (qca) circuits by global asynchrony. *J. of Syst. Architecture, Elsevier*, 53:551–567, 2007.

[26] E. Tabrizizadeh, H.R. Mohaqeq, and A. Vafaei. Designing qca delay-insensitive serial adder. *Proc. IEEE Int. Conf. on emerging trends in Engineering and Technology*, 2008.

[27] K. Walus, T.J. Dysart, G.A. Jullien, and R.A. Budiman. Qcadesigner: A rapid design and simulation tool for quantum-dot cellular automata. *IEEE Transaction on Nanotechnology*, 3(1), March 2004.

[28] Comsol multiphysics. http://www.comsol.com/.

[29] M.J. Donahue and D.G. Porter. Oommf user's guide, version 1.0. Technical Report Interagency Report NISTIR 6376, National Institute of Standards and Technology, Gaithersburg, September 1999.

[30] T. Fischbacher, M. Franchin, G. Bordignon, , and H. Fangohr. A systematic approach to multiphysics extensions of finite-element-based micromagnetic simulations: Nmag. *IEEE Transactions on Magnetics*, 43(6):Available on–line, 2007.

[31] M. Ottavi, L. Schiano, F. Lombardi, and D. Tourgaw. Hdlq: A hdl environment for qca design. *ACM Journal on Emerging Technologies in Computing Systems*, 2(4):243–261, 2006.

[32] E.W.Johnson J.R.Janulis S. Henderson and P.D. Tourgaw. Incorporating standard cmos design process methodologies into the qca logic design process. *IEEE Transaction on Nanotechnology*, 3(1):2–9, 2004.

[33] 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.

[34] M. Vacca, M. Graziano, and M. Zamboni. Asynchronous solutions for nano-magnetic logic circuits. *ACM Journal on Emerging Technologies in Computing Systems*, 7(4), December 2011.

[35] M. Martina and G. Masera. Turbo noc: A framework for the design of network-on-chip-based turbo decoder architectures. *IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS. I, REGULAR PAPERS*, 57(10):1776–1789, 2010.

[36] M. Crocker, M. Niemier, and X.S. Hu. A reconfigurable pla architecture for nanomagnet logic. *ACM Journal on Emerging Technologies in Computing Systems*, 8(1), February 2012.

[37] M.T. Alam, M.J. Siddiq, G.H. Bernstein, M.T. Niemier, W. Porod, and X.S. Hu. On-chip clocking for nanomagnet logic devices. *IEEE Transaction on Nanotechnology*, 2009.

[38] 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.

[39] M. S. Fashami, J. Atulasimha, and S. Bandyopadhyay. Magnetization dynamics, throughput and energy dissipation in a universal multiferroic nanomagnetic logic gate with fan-in and fan-out. *Nanotechnology*, 23(10), February 2012.

[40] J. Das, S.M. Alam, and S. Bhanja. Low power magnetic quantum cellular automata realization using magnetic multi-layer structures. *J. on Emerging and Sel. Top. in Circ. and Syst.*, 1(3), September 267-276.

[41] M.T. Niemier, E. Varga, G.H. Bernstein, W. Porod, M.T. Alam, A. Dingler, A. Orlov, and X.S. Hu. Shape engineering for controlled switching with nanomagnet logic. *IEEE Transactions on Nanotechnology*, 11(2):220–230, March 2012.

[42] Marco Vacca. Nanoarchitectures based on magnetic qca. Master's thesis, Politecnico di Torino, 2008.

[43] D.B. Carlton, N.C. Emley, E. Tuchfeld, and J. Bokor. Simulation studies of nanomagnet-based logic architecture. *Nanoletters*, 8(12):4173–4178, November 2008.

[44] E. Varga, G. Csaba, G.H. Bernstein, and W. Porod. Implementation of a nanomagnetic full adder circuit. *2011 11th IEEE International Conference on Nanotechnology*, August 2011.

[45] Mentor graphics. http://www.modelsim.com.

[46] M.T. Niemier, G.H. Bernstein, G. Csaba, A. Dingler, X.S. Hu, S. Kurtz, S. Liu, J. Nahas, W. Porod, M. Siddiq, and E. Varga. Nanomagnet logic: progress toward system-level integration. *J. Phys.: Condens. Matter*, 23:34, November 2011.

[47] W. Porod. Magnetic logic devices based on field-coupled nanomagnets. *Nano & Giga*, 2007.

[48] T. Teodosio and L. Sousa. Qca-lg: A tool for the automatic layout generation of qca combinational circuits. *Norchip conference*, 2007.

[49] S. Frache, D. Chiabrando, M. Graziano, F. Riente, G. Turvani G., and M. Zamboni. Topolinano: Nanoarchitectures design made real. In *2012 IEEE/ACM Int. Symp. on Nanoscale Architectures (NANOARCH*, pages 160–167, Amsterdam, The Netherlands, 2012. IEEE.

[50] S. Frache, M. Graziano, and M. Zamboni. A flexible simulation methodology and tool for nanoarray-based architectures. *IEEE International Conference on Computer Design*, pages 60–67, October 2010.

[51] C. Augustine, X. Fong, B. Behin-Aein, and K. Roy. Ultra-low power nano-magnet based computing: A system-level perspective. *IEEE T. on Nanotechnology*, 10(4):778–788, 2011.