POLITECNICO DI TORINO
Repository ISTITUZIONALE

Asynchronous Solutions for Nano-Magnetic Logic Circuits

Original
Asynchronous Solutions for Nano-Magnetic Logic Circuits / Vacca, Marco; Graziano, Mariagrazia; Zamboni, Maurizio. - In: ACM JOURNAL ON EMERGING TECHNOLOGIES IN COMPUTING SYSTEMS. - ISSN 1550-4832. - STAMPA. - 7:4(2011), pp. 15:1-15:18. [10.1145/2043643.2043645]

Availability:
This version is available at: 11583/2479980 since:

Publisher:
Association for Computing Machinery (ACM)

Published
DOI:10.1145/2043643.2043645

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

24 April 2024

Asynchronous Solutions for Nano-Magnetic Logic Circuits

Marco Vacca, Mariagrazia Graziano, Maurizio Zamboni, Politecnico di Torino

In the years to come new solutions will be required to overcome the limitations of scaled CMOS technology. One approach is to adopt Nano-Magnetic Logic Circuits, highly appealing for their extremely reduced power consumption. Despite the interesting nature of this approach, many problems arise when this technology is considered for real designs. The wire is the most critical of these problems from the circuit implementation point of view. It works as a pipelined interconnection, and its delay in terms of clock cycles depends on its length. Serious complications arise at the design phase, both in terms of synthesis and of physical design.

One possible solution is the use of a delay insensitive asynchronous logic, Null Convention Logic (NCL^{TM}). Nevertheless its use has many negative consequences in terms of area occupation and speed loss with respect to a Boolean version. In this paper we analyze and compare different solutions: nanomagnetic circuits based on full NCL, mixed Boolean-NCL and fully Boolean logic. We discuss the advantages of these logics but also the issues they arise. In particular we analyze feedback signals, which, due to their intrinsic pipelined nature, cause errors that still have not found a solution in the literature. The innovative arrangement we propose solves most of the problems and thus soundly increases the knowledge of this technology. The analysis is performed using a VHDL behavioral model we developed and a microprocessor we designed, based on this model, as a sound and realistic test bench.

Nanomagnetic Logic Circuits (NLC) are mentioned in the International Technology Roadmap for Semiconductor (ITRS) [Semiconductor Industry Association 2008] as one of the most promising substitutes for CMOS technology. They rely on the Quantum dot Cellular Automata (QCA) idea [Lent et al. 1993], [Kummamuru et al. 2003], [Csurgay et al. 2000], based on bistable cells having only two stable states (Figure 1.A). These two states represent the logic values '0' and '1'. Circuits are built using many identical cells placed close together. The state of every cell is therefore defined by the electrostatic interaction between adjacent elements. This theoretical principle is normally implemented in two ways, molecular QCA and magnetic QCA, also called NML (Nano-Magnetic Logic). Molecular QCA are built using complex molecules as basic cells [Lu and Lent 2005],[Pulimeno et al. 2011],[D. et al. 2009]. On the one hand molecular QCA have great appeal due to the high speed they are expected to reach (about 1THz according to the work in reference [Y. Lu 2006]) along with the possibility of working at room temperature; on the other hand, neither wires nor gates have been experimentally demonstrated yet. At the moment, in fact, self-assembly techniques are not advanced enough to create useful circuits.

NML Devices [Imre 2005][Csaba and Porod 2002] are built using rectangular nanometer scale magnets, which can be approximated as single domain devices. Since the magnets are rectangular, the shape anisotropy allows only two stable magnetization states for these devices. In these two stable states magnetization is present in the long side of the magnet (called easy axis), and it represents the two logic values '0' and '1' (Figure 1.B). NML cannot reach the high speed of its molecular counterpart, as the expected speed is few hundred MHz, according to reference [Rizos et al. 2009]. However, small circuits have already been experimentally demonstrated [Imre et al. 2003][Pulecio and Bhanja 2010][Orlov et al. 2008]. The main advantages of this implementation are the expected low power dissipation [Csaba et al. 2004] and its intrinsic magnetic nature: NML devices maintain the information stored even without external power supply. It is interesting to note that in this technology even a simple wire is based on the elementary logic magnet (Figure 1.C), and this has consequences that will be highlighted in the following paragraphs. Other logic gates are the

Author's addresses: VLSI Laboratory, Electronics Department, Politecnico di Torino, Corso Duca degli Abruzzi 24, Torino (TO), Italy.
.

inverter (Figure 1.D) and the Majority Voter (MV). The MV can be used as an AND or an OR, depending on whether one of its inputs is fixed to '0' or to '1' (Figure 1.E).

Irrespective of the implementation, QCA circuits require the use of an external clock field to drive the information through the circuit [Niemier et al. 2007][Alam et al. 2007][Rizos et al. 2009] in order to avoid error generation. For example, with magnetic implementation, the maximum number of magnets that can correctly align in an antiferromagnetic order is between 10 and 20 [Imre et al. 2003], less if thermal noise is considered [Csaba and Porod 2010]. The external field, magnetic or electric, depending on the implementation chosen, is used to drive the cells in an intermediate unstable state, lowering the energy barrier between the stable states. When the field is removed, cells are free to reorient themselves depending on the other neighbor cells (more details are reported in Section 1). In molecular QCA this clock field may act as a reset field, or, on the contrary, as an activating field, enabling cell reorientation when it is applied. Howerer this depends on molecule characteristics. In general, the cyclic switching *on* and *off* of this external field likens it to a *clock*, even though it is conceptually more similar to a rhythmic power supply, because it *enables* the information propagation. The whole circuit is patterned by *clock regions*; distributed in the whole layout area, these regions can be reached by the external field with an appropriate timing. Typically, three or four types of region are necessary, as explained in section 1, and magnets in the regions of the same type are reached by the same clock signal; different clock signals have different phases, but the same period.

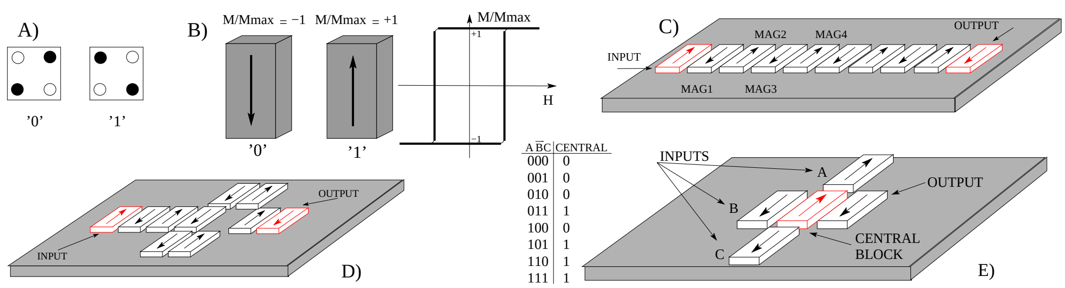

Fig. 1. A) QCA base cell. B) Nanomagnetic Logic Device (Magnetic QCA base cell) and its magnetic hysteresis; the two possible states represent the logic values '0' and '1'; C) nanomagnetic wire on a plane; D) a nanomagnetic inverter; E) a nanomagnetic Majority Voter on a plane and its truth table.

The use of this clock system leads to a problem known as "layout=timing" [Imre et al. 2003]. The delay of a QCA circuit is measured in terms of clock cycles, or, in other words, by counting the number of *clock regions* the circuit goes through. This number depends in turn on the total number of cells in the circuit, but the number of cells is related to the circuit layout (see Section 2 for a detailed explanation on the layout organization). This dependency increases the constraints on criteria and tools that can be used at the design stage, e.g. synthesis and physical design. The reason behind this is that a layout change during circuit design implies a delay variation which can cause improper behavior of the circuit. Although as yet only developed at research level, the tools [Zhang et al. 2005][Chung et al. 2005] can be helpful, but only for simple circuits. One of the proposed solutions for this problem is to use an asynchronous delay insensitive logic, like Null Convention Logic (NCL) [Fant and Brandt. 1996][Choi et al. 2006][Choi et al. 2007]. In this logic every gate switches to a new value only when all the signals arrive at its inputs. Therefore, it is possible to place gates everywhere in the circuits, without any concerns regarding signal synchronization (e.g. regarding the number of clock zones interested by that signal). The use of asynchronous logic applied to NML leads then to the realization of a Globally Asynchronous and Locally

Synchronous (GALS) architecture. The GALS technique is often used in CMOS digital designs when interconnect related issues must be solved [Casu and Macchiarulo 2007] or when different and complex synchronization blocks are to be interfaced [Martina and Masera 2010]. In the case of NML, this happens because the single NCL information propagation is synchronized with the clock signal, but the whole circuit is asynchronous and relies on a handshake communication protocol like most of the commonly used asynchronous architectures [Davis and Nowick 1998][Sparso and Furber 2011].

The use of asynchronous logic seems then to be a natural solution for NML technology, however, advantages come at a price. As in the CMOS technology, NCL leads to a bigger area occupation. From our preliminary investigations we have found that the use of NCL logic on NML circuits has even a bigger penalty than in the CMOS case, and it also causes a slow-down in circuit operations. For this reason the way in which asynchronous logic impacts magnetic logic circuits has been investigated exhaustively in this work, outlining advantages and disadvantages. Not only an NCL solution is explored, but both a mixed NCL-boolean organization and a fully boolean asynchronous solution are studied. The analysis is performed using an HDL behavioral model of NML circuits that we have developed [Vacca 2008][Graziano et al. 2009][Graziano et al.] and explained in detail in [Graziano et al. 2011]. This allowed us to design a complex architecture, a microprocessor, as a benchmark to evaluate the circuit logic behavior and to estimate its area and power consumption.

In this paper, after an explanation of the NML technology foundations, the multiphase clock system (Section 1) is analyzed. Two main issues are then discussed arising from the use of the external clock system: how to manage synchronization at layout level and how to handle feedback signals (Section 2). In Section 3 NCL logic applied to NML is examined using a microprocessor as a case study and in Section 4 the implementation of the same microprocessor based on a mix between Boolean and NCL logic is discussed. We demonstrate in the same section how this solution is a good compromise between circuits feasibility and performance. Finally Section 5 describes how the same microprocessor was implemented using Boolean logic only, but still relying on an asynchronous-like organization. This solution, although requiring some care in the physical design phase, demonstrates an important improvement in performance and discloses interesting suggestions for further research.

1. NANO-MAGNETIC LOGIC CIRCUITS BACKGROUND

Nano-Magnetic Logic circuits behavior is based on the interaction between neighbor cells. However the influence of a nanomagnet on its neighbor may be too small to effectively influence its magnetization. This happens because a high energy barrier between the two stable magnetization states exists: a desirable property, if stability is to be assured. An external mechanism is then required to lower the energy barrier and to drive cells in an intermediate unstable state. This action favors the influence between two neighbors nanomagnets. This mechanism is called *clock* and, in the case of NML, is a magnetic field applied along the short side of the magnet (hard axis). The magnetic field is generated using a current flowing through a wire buried under the plane of the nanomagnets (Figure 2.A) [Niemier et al. 2007]. Different clock signals are routed through the plane patterned with regular regions. In the example in Figure 3 the different colors represent the zones interested by different clock signals. The clock zones layout on the plane is a complex trade-off between physical feasibility of wires and NML signals propagation. We proposed a feasible and realistic layout structure to route this signal [Vacca 2008] [Graziano et al.][Graziano et al. 2011]: an example is in Figure 2.B, which planar top view of zones organization is in Figure 3. When the clock field is applied, the related nanomagnets magnetization assumes the same direction of the field. When the field is removed nanomagnets reorient themselves antiferromagnetically.

In doing so they follow input cells placed nearby that are in the meanwhile in a stable state, as in the example in Figure 2.C.

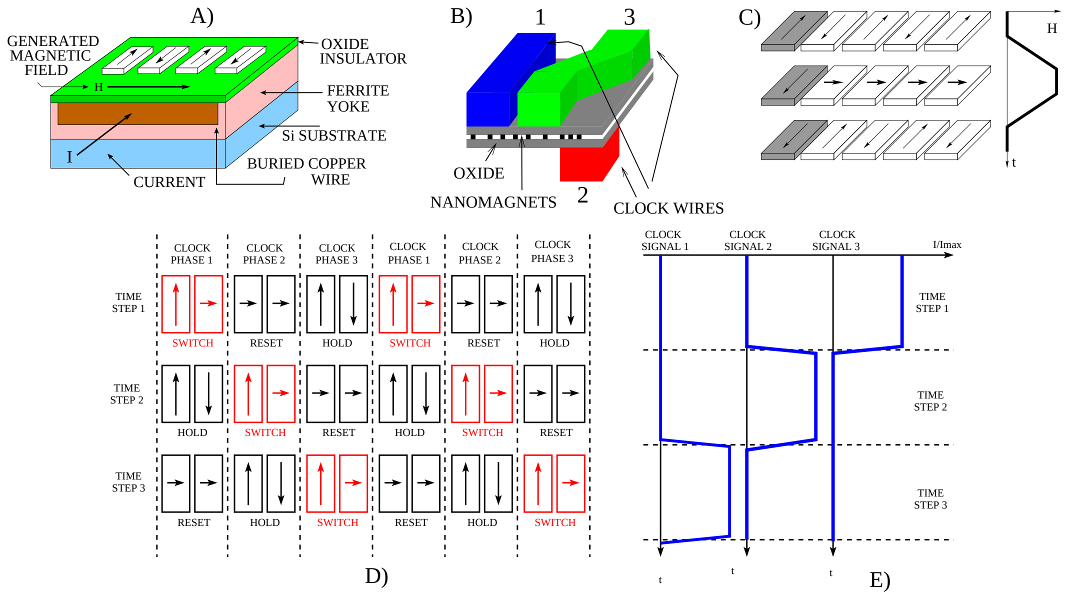

Fig. 2. A) Buried wire delivering clock current and generating the magnetic field. B) A 3D view of the snake-clock organization we proposed [Graziano et al.]. C) Effect of clock signal application on magnetization. D) Regions in which different clock phases are applied. Time and space evolution of the nanomagnets that compose the circuit. E) Clock phases.

A multhiphase clock system is necessary [Imre et al. 2003] in order to build electronic circuits. Three or four phases clock signals (Figure 2.E) are used to independently drive the different clock zones of the whole circuit. The working principle of the clock system is shown in figure (Figure 2.D), according to our three phases proposal [Graziano et al.]. At every time step a clock zone can be in on of three different phases: HOLD, RESET, SWITCH. In the HOLD phase no magnetic field is applied, nanomagnets are in one of the two stable magnetization states, and have a strong influence on the magnets of the neighbor zones. In the RESET phase, the magnetic field is applied and magnets are driven in the unstable state, therefore they have a small influence on the magnets of the neighbor zones. In the SWITCH phase, the magnetic field is first applied and then removed, therefore nanomagnets reorient themselves following the last magnets of the neighbor zone which is in the HOLD phase acting as an input. During the next time step the situation is repeated but the zone in the SWITCH phase is the next in the space sequence due to the time evolution of clock signals. The time evolution shown in Figure 2.D gives a demonstration of how the information propagates through the circuit. Figure 3 shows a possible layout of magnetic cells and a possible information flow, which follows a "snake" like propagation in zone 1-2-3-1-... (from here the name "snake-clock"). The circuit, then, is organized as a fully pipelined architecture.

It is worth underlining that this clock signal is different from the CMOS clock one, where parameters like clock frequency and number of pipeline stages are free and are designed to obtain the desired performance (provided that technology limits are respected). In this case the clock is first of all necessary to assure correct circuit operations, and its characteristics,

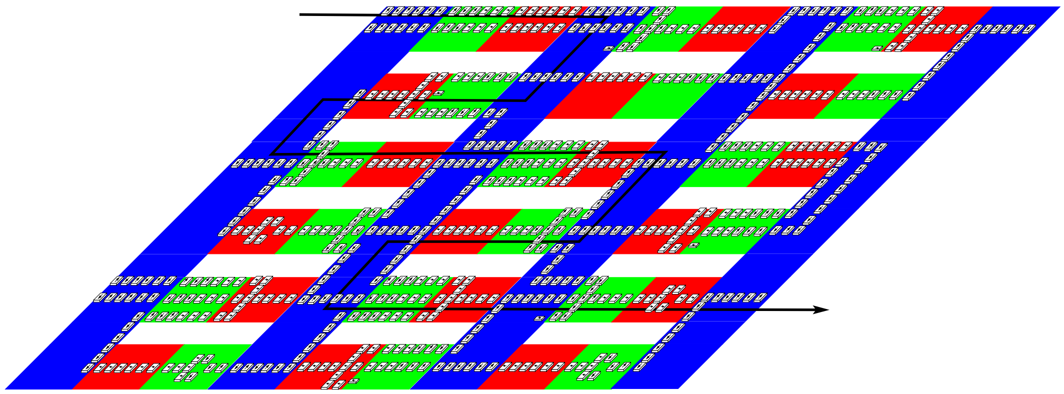

Fig. 3. An example of a plane where nanomagnets circuits and wires are placed and connected. Different colors of rectangles refer to different clock zone. In white zones no magnets are present because that is the region where two phases have a superposed effect (they virtually cross there), according to layout in figure 2.B.

as frequency or slope, are strongly related to the physical constraints of this technology and cannot be easily changed. In order to assure a correct switching, in fact the clock frequency must be strictly related to the magnets physical behavior, for example, to the switching time and the associated power consumption. In the same time, the number of pipeline stages, that is equal to the number of clock zones, depends on the physical feasibility of the clock wires (minimum metal pitch) and on the maximum number of magnets that can be placed in one zone. Experimental results show that the maximum number of magnets in a chain that switch without error generation is between 12 and 20 [Imre et al. 2003]. If we consider the thermal noise, this number is significantly reduced to a value between five and ten [Csaba and Porod 2010]. As a consequence the circuit must be divided into smaller areas with a limited number of magnets. This in turn increases the number of pipeline stages.

The clock signal frequency we have used in our simulations is obtained by accurate micromagnetic simulations performed using a finite element simulator called NMAG [Fischbacher et al. 2007]. In Figure 4.A an example of a wire starting with one fixed input and nine magnets is shown. The magnetization reorient according to an antiferromagnetic order. In the case presented the simulation is stopped during the transient in order to show an intermediate point, so that not all magnets are already correctly oriented. In Figure 4.B the magnetization transient behavior is shown for the first four magnets: magnetization goes from 0 to a negative value for magnets one and three, and to a positive value for magnets one and three. From this and other simulations we obtained that the average switching time of a permalloy nanomagnet with sizes $50nm \times 100nm \times 30nm$ is about 120ps. The duration in time of the SWITCH phase must be long enough to assure the correct reorientation of all the magnets inside a clock zone. In order to obtain the timing of the switching phase that we can apply to the whole circuit, the average number of magnets that can correctly switch (we have chosen in this example 15, as an average value between 12 and 20) is multiplied for the average switching time of a single magnet:

$$Phase = 120ps \cdot 15 \qquad\qquad f_{ck} = \frac{1}{Phase \cdot 3}$$

The clock period is evaluated as three times the duration of the SWITCH phase, and the frequency is the inverse of the clock period. From our simulations we have obtained a maximum frequency for the clock signal of 180 MHz.

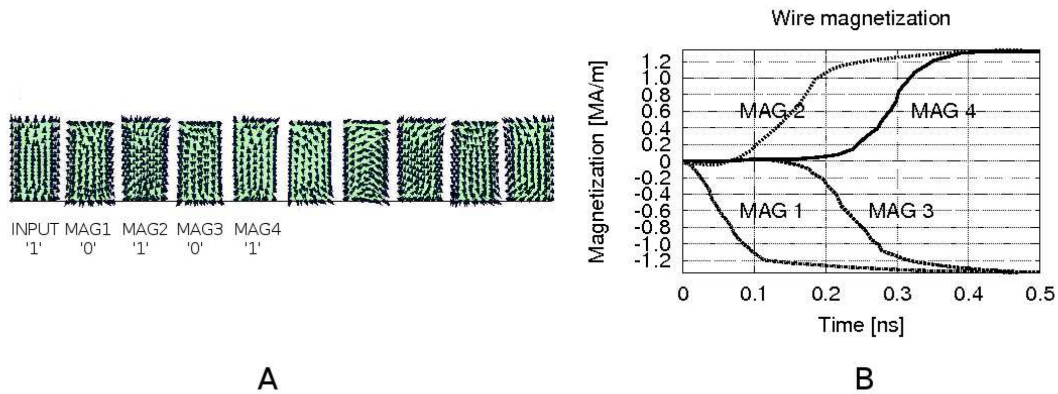

Fig. 4. A) NMAG 3D simulation of a wire of nanomagnets with a fixed input (left); the other magnets start from an initial reset state (horizontal magnetization). B) Transient behavior of the magnetization variation for the first four magnets.

2. NANO-MAGNETIC LOGIC CIRCUITS DISCUSSION: PROBLEMS AND SOLUTIONS

The clock system previously presented implies an intrinsically wave-pipelined circuit. To better understand this, a comparison can be done with CMOS circuits. Every clock zone, where just a simple magnetic wire is routed, has the same behavior of a D-ff with a clock signal similar to the waveforms in Figure 2.A. As also shown in the Figure 3 layout, every clock signal is applied to a different clock zone. In other words, the first clock signal is applied to all the zones with the first color, the second clock signal is applied to all the zones colored with the second color and the third clock signal is applied to all the zones correspondent to the third color. This means that a group of three consecutive zones has a total delay of one clock cycle. The behavior is a wave-pipelined one. This is an intrinsic characteristic of NML technology and cannot be changed.

2.1. Problems

Two problems arise in this structure. They are discussed herein.

Layout=timing. This issue is explained in Figure 5.A and 5.B, where a MV is reached by three inputs according to two different organizations. The delay of a signal, in terms of clock cycles, depends on the number of clock zone it crosses. In order to implement a correct circuit signals must arrive at the inputs of every logic gate (i.e. the MV in this case) at the same time. This synchronous arrival time is necessary because when the reset field is released a magnet switches according to its neighbors. No more sampling of neighbors magnetization can occur, if not after a new reset. Therefore the number of clock zones crossed by each input signal must be the same (Figure 5.B). If this does not happen (Figure 5.A) the operation result is not correct, because data arrive at different clock cycles. In the simple example shown here it is easy to synchronize signals by controlling the routing. However, in complex circuits only automatic tools could help, but still it may happen that constraints could not be completely satisfied.

Feedbacks. The second problem arises from feedback signals. An example is presented in Figure 5.C, where an ALU executes the addition between one input and its own output. Since in the case of NCL the structure is pipelined, the ALU input arrives at every new clock cycle, but the second input, the feedback, arrives later (in this example after 100 clock cycles) due to the length and the delay of the NML wire. Therefore at every time step the ALU performs the addition between the input and its output result obtained 99 clock cycles before. Changing the length of the input wire does not solve the problem

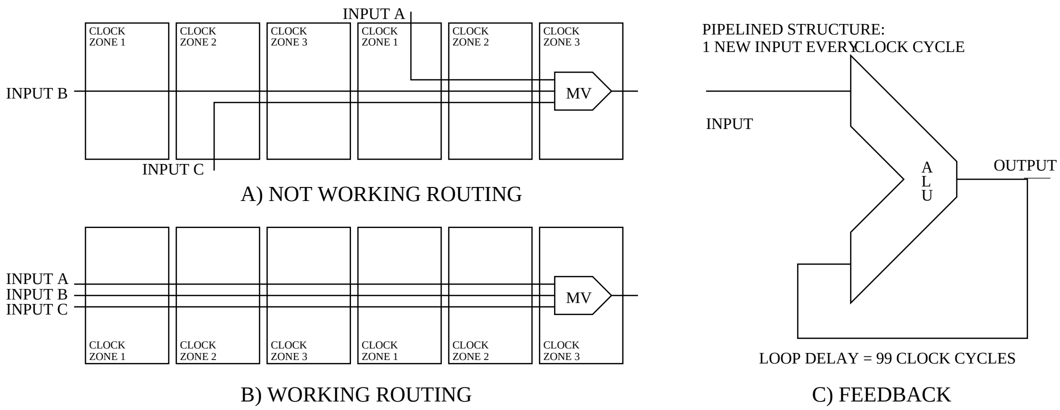

Fig. 5. QCA problems related to their intrinsic pipelined nature. A) and B) represent the problem of signal synchronization at layout level. A) Shows a case where the circuit will not work properly because the input wires pass through a different number of clock zones. B) Shows a working case with input signals correctly synchronized. C) Is a schematic representation of the “jump” problem of feedback signals.

because it simply changes the circuit latency. The circuit will not work even though the input is delayed to match the length of the loop. For example, if a new input is sent exactly every 100 clock cycles, and in the meanwhile its value is kept constant, the circuit is synchronized and works correctly. But, if the input arrives with a bigger delay (e.g. 300 clock cycles), the circuit will not work again. This happens because the feedback signal still arrives at the ALU input after 100 clock cycles (for example suppose the output is 0). The value of the first input is kept constant (for example suppose the input value is 1), therefore the ALU exacutes the addition between 0 and 1 and gives as a result ’1’. At 300 clock cycles the situation is repeated but at this time the two input values are 1 (kept constant) and 1 (the output of the previous operation). This operation gives 2 as a result. At 300 clock cycles a new input is sent, but the output of the ALU will show the wrong value 2 instead of the expected 0. So the circuit works only if the input is delayed of exactly 100 clock cycles. This is critical because a complex circuit is composed of many loops. If, for example, inputs are synchronized with the longest loop, then the shortest loop will not work. A possible solution to this problem, coming from technology, could be the use of electric interconnections only for feedback signals. The magnetization can be converted in an electric signal, using for example the “devide” developed in [Becherer et al. 2009], transferred using a copper line. Finallly it could be reconverted into a magnetic field, for example using the magnetic field generated by the flow of a current. This solution is technologically complicated, but it may solve the problem without delaying circuit operations and could be adopted al least for very long interconnects.
From the logic point of view, of interest in this paper, architectural solutions for this specific problem are introduced herein.

2.2. Solutions

A possible solution for these problems is using asynchronous circuits. In this work we explore and compare three different approaches.

For example, one possibility is using a delay insensitive asynchronous logic, i.e. Null Convention Logic (NCL). Signals are encoded using two bits: logic value ’0’ is represented with ’01’ and logic value ’1’ is represented with ’10’. These two values represent the DATA state. Value ’00’ represents the NULL state and value ’11’ is not allowed. The delay insensitivity is achieved alternating NULL and DATA states. A NCL gate, then, switches from the NULL state to one of the two DATA states; however, this happens only when all the inputs switch from NULL to DATA. A gate will remain in the two DATA state until all inputs return to the NULL state, and then the cycle restarts. Circuit switch periodically from NULL state to DATA state, therefore NULL state works as a time reference for this logic. Adopting this logic then all the synchronization problems of QCA, mentioned before, are automatically solved. Howeve, using a two bits encoding implies to double the number of wires of the circuit. This problem is partially solved, because NML technology, at the moment, allows only coplanar wire crossing. However, this technique, experimentally demonstrated in [Pulecio and Bhanja 2010], is non-trivial to make. Therefore it is better to limit the number of wire crossing in the layout. This constraint is for sure a limitation for NCL logic and can be solved finding a technological way to make multilayer structures.

Most of the works in literature focus on small circuits, but a realistic architecture should be used to reveal the potentiality and critical points of this technology. For this reason we have investigated the behavior of a complex circuit (a microprocessor) that we have designed using NML logic. A pure NCL solution is discussed in Section 3.

This version is then compared to other two possible implementations of the same circuit. We explored a mixed Boolean-NCL approach (Section 4) in order to limit the overhead due to NCL at least where the price to pay due to NCL is unbearable. Finally, still maintaining the idea of asynchronous behavior, we propose a version of the microprocessor based on Boolean logic only (Section 5). This new solution solves most of the problems and remarkably improves the performance.

2.3. Methodology

To analyze NML circuits operation and performance we have developed a VHDL behavioral model of Nano-Magnetic Logic circuits. The model is simply a circuit, described using VHDL, which behaves like the equivalent NML counterpart. We introduced the model in [Vacca 2008][Graziano et al. 2009] and explained more in details in [Graziano et al.][Graziano et al. 2011]. Even though based on the ideas in [Ottavi et al. 2006][S. Henderson and Tourgaw 2004][Huang and Lombardi 2007], that were developed for general QCA, our model is particularly dedicated to NML circuits. It accounts for the accurate physical layout of NCL gates, allowing a realistic representation of the NML circuits behavior. Moreover, it is based on the clock structure we presented in [Graziano et al. 2009] and [Graziano et al.][Graziano et al. 2011]. Starting from the logic equation of each NCL gate (an example in Figure 6.A), we have designed the custom layout of each NCL gate (Figure 6.B). Then the layout is converted in the correspondent model described using VHDL (Figure 6.C). In a real circuit, magnets within a clock zone accept a new data when the magnetic field is removed. Therefore we use a register to simulate the delay of a clock zone, using the clock signals reported in Figure 6.C. When the clock signal in the model is high, the register accepts a new data, therefore it is equivalent to the switch phase of the real circuit. In this example, in order to perform logic operations an ideal MV, without delay, is used.

This model is adopted to hierarchically build more complex circuits. It is currently based on gates layout generated ad-hoc. Clearly a more general design approach would be helpful. For this reason we are at the moment working on an automatic tool for synthesizing, placing and routing this kind of circuits in order to obtain an accurate representation for every architecture. In this work the physical layout of the gates is based on the constraint that a wire can be composed by a limited number of magnets, for example, between 12 and 20. Hovewer, if a more critical condition is considered, as in [Csaba and Porod 2010], where the presence of thermal noise is taken into account, the number of magnets is to be reduced. However, our model is fully parametric with respect to this number, and thus any case can be easily analyzed. It is worth underlining that less magnets in each clock zone means an higher clock frequency, but more pipeline stages.

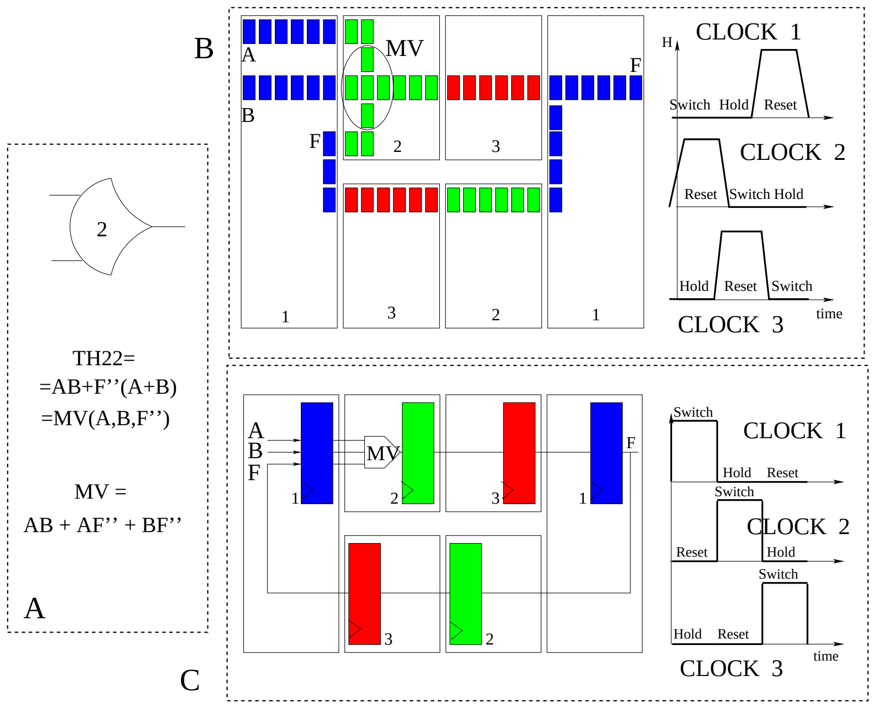

Fig. 6. VHDL behavioral model for NML logic. A) Symbol and logic equation of a NCL gate named TH22 [Graziano et al.]. B) Physical layout of the gate. Magnets of the same color are in the same clock zone. The clock signals represent the applied magnetic field of each clock zone (1,2,3 in sequence). C) Model of the gate described using VHDL. Registers are used to simulate the clock application and the consequent delay, while an ideal majority voter is used for logic operations. The clock signals represent the signal used to simulate the multi-phase behavior inside the model.

We have also improved our model allowing for a hierarchical estimation of the circuit power dissipation. Starting from the exact number of magnets of the basic logic gates (majority voter, inverter, wire), the total number of magnets of the circuit is estimated. By multiplying the total number of magnets for the average power dissipated by each of them [Augustine et al. 2011], it is possible to obtain the average power dissipated by the magnets during the switching. Moreover, knowing the dimensions of the magnets and estimating the wasted space, the circuit area can be calculated. These data allow the evaluation of the clock wires length, and then the power dissipated by the Joule effect in the clock is calculated. We based our estimation on the most efficient clock wire structure presented in [Augustine et al. 2011]. However the power model is not detailed here, as out of the focus of this work.

3. NCL LOGIC MICROPROCESSOR

Using NCL gates only we designed a simple but complete microprocessor. The processor, inpired to [Walus et al. 2005] but substantially improved, was chosen because it contains both sequential and combinational logic circuits, therefore it is a good benchmark for testing NML. The microprocessor architecture is shown in Figure 7.

It is organized in four main blocks: A program counter, an instructions memory, a data memory and an ALU. The program counter generates the addresses for the instruction memory, which is a parallel memory capable of storing 16 instructions. Another parallel memory is used for storing data. Finally the ALU is used for the computational part, and ii executes arithmetic (addition, subtraction) and logic (AND, OR) operations. The micro-

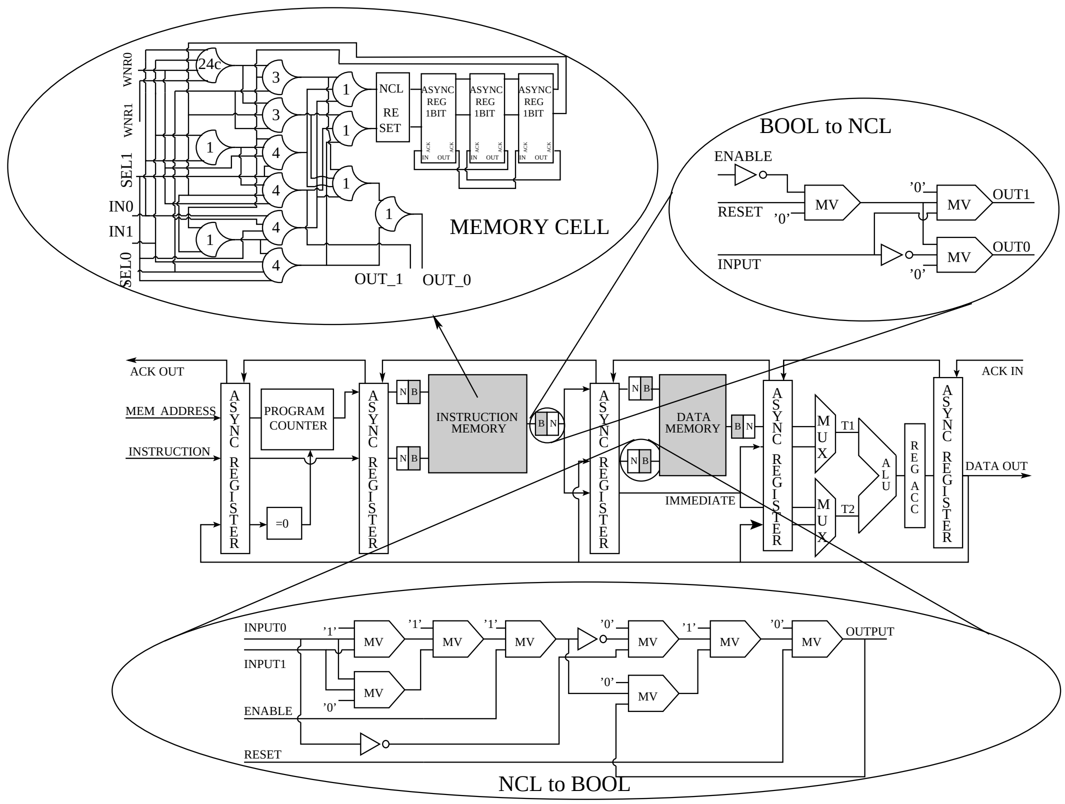

Fig. 7. Microprocessor architecture. The gray blocks represent the Boolean part of the circuit, while the B/N blocks are the interfaces between Boolean and NCL logic. In the pure NCL version of the microprocessor, memories are substituted with parallel NCL memory and the interfaces are absent. In the upper-left detail the structure of the memory cell in the NCL memory is presented. In the upper-right detail the interface between Boolean and NCL logic is shown, while in the bottom detail the NCL-Boolean interface (N/B) is represented.

processor is divided into four stages, separated by asynchronous registers. These registers are different from their CMOS counterpart, because they have no memory ability and their only purpose is to implement the asynchronous communication protocol.

As an example, the top-left inset of figure 7 shows a cell of the instruction memory. It is based on NCL gates and NCL registers. Functions and naming conventions of NCL are complex and not reported here for sake of brevity. Details are given in our work in [Graziano et al.][Graziano et al. 2011]. However, gates here labelled with number 4 and 1 are very similar (just slightly more complex) to the TH22 gate previously described (section 3.3, figure 6), chosen there as the simplest among NCL gates. By associating the layout in figure 6.B to the memory cell structure and, rising another hierarchical level up, to the whole architecture in figure 7, an idea can be conceived on how to relate the processor structure to a topology like in figure 3.

The architecture is simple, but can execute many type of instructions, like memory read/write, jump and arithmetical/logic ones. To test the microprocessor we have implemented a division algorithm, that executes, in the case reported here, 12/4. The simulation waveforms obtained using Modelsim [Modelsim] are shown in Figure 8. The two bits encoding, typical of this technology, are in Figure 8 in the two top and bottom arrays. A periodical switching from DATA state to NULL state (when al signals are '00') can also be observed in Figure 8. The time evolution of the system follows the ACK signal.

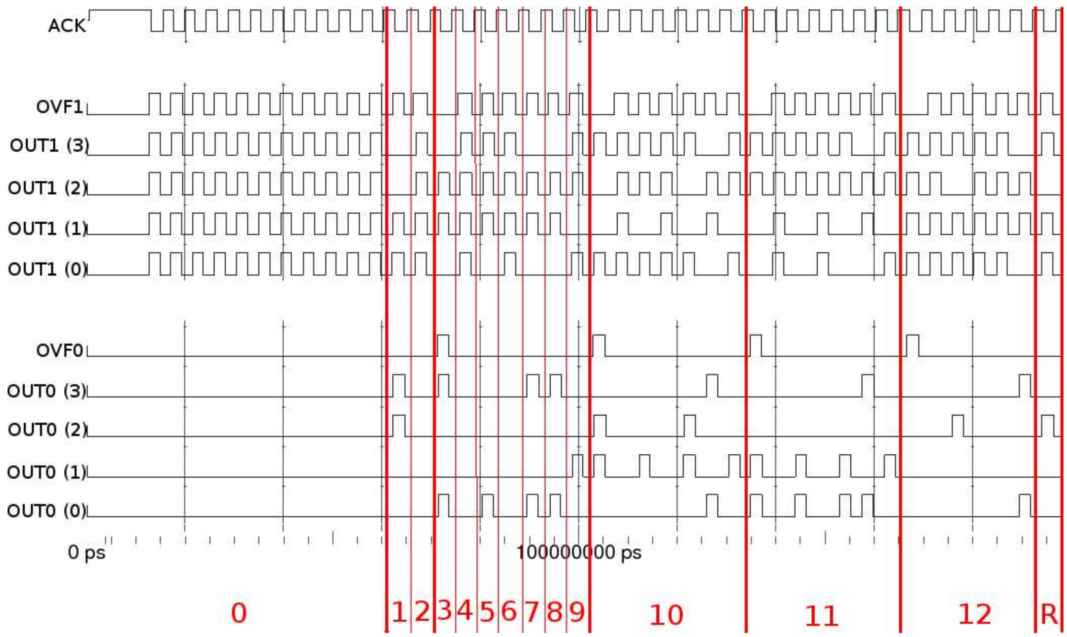

Fig. 8. Simulation results of the the division algorithm executed on the pure NCL microprocessor. In the mixed case waveforms are identical, but the time of the execution is reduced.

The algorithm is simple and follows the phases enumerated hereinafter.

0. Instructions are loaded in memory, outputs are always in DATA 0 (OUT0(x) = 0 and OUT1(x)=1)
1. First operand (1100) forced to circuit output
2. First operand stored in data memory (outputs = 0)
3. Second operand (0011) subtracted from first
4. Result is stored in data memory
5. A counter variable is incremented by 1
6. Value stored in data memory
7. Previous operation loaded
8. Check if 0
9. Jump back to step 4 and start again
10. 11.12. Repeat

R. Final result shown (0100)

The performance of the processor are shown in Table I. The time required for the execution of an instruction is about $5.35\mu s$, which is around 1000 times bigger than the clock period used (5.46ns). This can be easily explained because the delay insensitivity of the NCL logic is assured by freezing the circuit operations, while waiting for the arrival of all the signals to the circuit inputs. However, since NML have a pipelined nature, the propagation time of the wires in terms of clock cycles (latency) can be very high, therefore the operations are stopped for a very long period.

It is important to underline this point: A pure synchronous Boolean NML circuit has a

throughput of 1 data for every clock cycle, due to its pipelined nature, but only combinational data-flow circuits are allowed (no feedbacks). Therefore an hypothetical Boolean NML processor could execute one instruction at every clock cycle, i.e. every $5.46ns$. But since feedback cannot work, this kind of microprocessor cannot be really used in its pure form. The NCL solves the synchronization problems allowing the construction of any kind of circuits at the cost of dramatically decreasing the overall speed. The total power dissipation of this version of the processor (due to magnets and clock) is $63.8\mu W$, which is a very high value, compared to the results found using the other solutions discussed in Section 4 and 5. This is due to the high number of magnets that compose the microprocessor, about 4 millions of nanomagnets. This is another disadvantage of NCL logic, the area increment corresponds to an increase in power dissipation.
In the last row of Table I an estimation of the power consumption due to clock wires is shown, calculated according to the methodology described in previous section.

To summarize the results for this implementation we can state that adopting NCL completely solves the NML synchronization problems. Moreover the circuit fabrication is simpler, because gates can be placed without worrying about signal synchronization. Hovewer the drawback is very troublesome: The circuit area is significatively higher, and the area increment generates a proportional increment in power dissipation and a decrement in circuit speed. Hovewer the huge area increase depends mainly on memories, therefore if we implement them using Boolean logic, we expect to gain in performance. This lead us to the mixed logic approach discussed in the following section.

4. MIXED LOGIC MICROPROCESSOR

To improve performance we adopted a different solution. We have designed the most critical combinational parts of the processor using Boolean logic, and the sequential part using NCL logic, introducing thus a new mixed logic. This solution is based on the assertion that in NML technology combinational circuits have good performance, but also they are less complicated to implement from the synthesis point of view. From the layout point of view particular care must be used in signal synchronization (layout=timing). Therefore, if the combinational parts are implemented using Boolean logic, the performance can be substantially improved, as the number of magnets is reduced. A synchronization signal is still required but it is much more easy to be routed. It is in fact related only to some areas and not to the whole circuit, and it is also necessary only for combinational circuits. The feedback problem still remains, in this case, therefore asynchronous registers are used to better handle feedback signals.

We developed two interfaces which encode/decode signals from Boolean to NCL and from NCL to Boolean. The two interfaces are shown in Figure 7. The *Boolean-NCL* logic interface is simple, because it has only to split the Boolean signal in the two bits according to NCL encoding. The *NCL-Boolean* logic interface is more complicated. This is due to the necessity of including a memory loop inside the interfaces. NCL switches periodically from NULL to DATA, but Boolean logic is always in the DATA state. As a consequence, this interface not only has to merge the two bits encoding in one single bit, but also it has to maintain the value stored when the NCL logic is in the NULL state.

Moreover, the two interfaces must guarantee the synchronization between the two logic topologies. Therefore the interfaces use an $ENABLE$ signal which arrives from the previous stage. This signal is different from the ACK signal, which arrives from the next stage. The enable signal is generated by the logic block placed before the interface, and it is generated only when that block has updated its output.

The whole architecture is reported in Figure 7. The differences with respect to the pure NCL version consist in the Boolean blocks (gray blocks in figure) and in the interfaces (B/N and N/B) blocks in figure. We have chosen to realize in Boolean logic only the two memories and to leave the other component in NCL. This choice is due to the inefficiency of memory structures in NCL logic. The memory cell of the Boolean memory is shown in Figure 9, in the inset. It is simpler than its NCL counterpart, and thanks to its regularity it is more likely to keep under control the delays due to magnetic wires (layout=timing). This would be for sure more complicated in a sparse logic block. We have simulated the same division algorithm and measured the processor performance. Waveforms are not reported because are identical to those shown in Figure 8, but with a changed time scale.

The performance of the mixed logic processor are in Table I. The time required for an instruction execution is slightly smaller, $4.41\mu s$ instead of $5.35\mu s$. The improvement is not so high because in the previous case the NCL memory was a parallel memory so it had not so big an impact in the time balance. The big improvement is in the estimated number of nanomagnets, which is 600K instead of 4M, and the power dissipation which is 6 times smaller.

As we expected, implementing the memories using Boolean logic allows to save a lot of area, significatively increasing the performance. Hovewer, the overall performance are still not satisfactory. It is also clear from our analysis in Section 2 that the presence of at least one feedback signal slows down the operations of any QCA circuit implemented in any technology. But, on the contrary, if the whole circuit is implemented using Boolean logic, the performance can be maximized. However, the implementation of a complex NML circuit using only boolean logic has two problems: Signals synchronization become much more complicated, and an asynchronous like protocol is still needed to handle feedbacks. While trying to solve these issues, we have found a way to design NML circuits using Boolean logic only, but still implementing an asynchronous-like protocol. An analysis follows in next section.

5. BOOLEAN LOGIC MICROPROCESSOR

The innovative idea that we propose here is based on the discussion related to feedback in Section 2. If input data are sent after a certain time interval, which corresponds to the delay of the longest loop, circuit can correctly work. Hovewer in a complex circuit, many loops are present, and it is necessary to take into account the longest loop. Notwithstanding this, as mentioned in Section 2, the shorter loops will not work. The idea is to use a block, placed at the end of every feedback loop, which slows-down the faster loops to reach the speed of the slowest loop present in the circuit. This is done using an asynchronous-like register placed at the end of the loop, at the beginning of the pipe stage. It is sketched in Figure 9 where the architecture of the pure Boolean microprocessor is shown. The architecture is the same presented before, but now all the blocks are implemented using Boolean logic, and the synchronization block is placed at the end of every feedback loop.

The scheme of this block is shown in Figure 9 in the inset (bottom right). It is implemented using a multiplexer with the output connected to one of its input. The other input accepts incoming data from outside. The multiplexer normally is in the loop mode. The selected input is its output. In this situation the output will maintain always the same value. The circuit placed after this block works normally and therefore a feedback signal is generated.

However no new inputs are accepted and the circuit is then frozen in the same state: this is a latch in the memory stage. When a time correspondent to the longest loop inside the circuit passed, a new input is sent. But at the same time a short pulse (ENABLE) is sent to the selection bit of the synchronization multiplexer. This signal travels through the circuit slightly more slowly than the input signal. This slower behavor can be achieved, if necessary, forcing an appropriate layout. This condition, even if not simple, causes less bourdens at layout level with respect to a totally synchronous solution, because only the ENABLE signal routing would be critical. Therefore, when this signal reaches the multiplexer, which behaves like a latch responding to a token, all inputs are already at destination. The pulse allows the multiplexer to sample the new inputs, that are stored till the next pulse arrives.

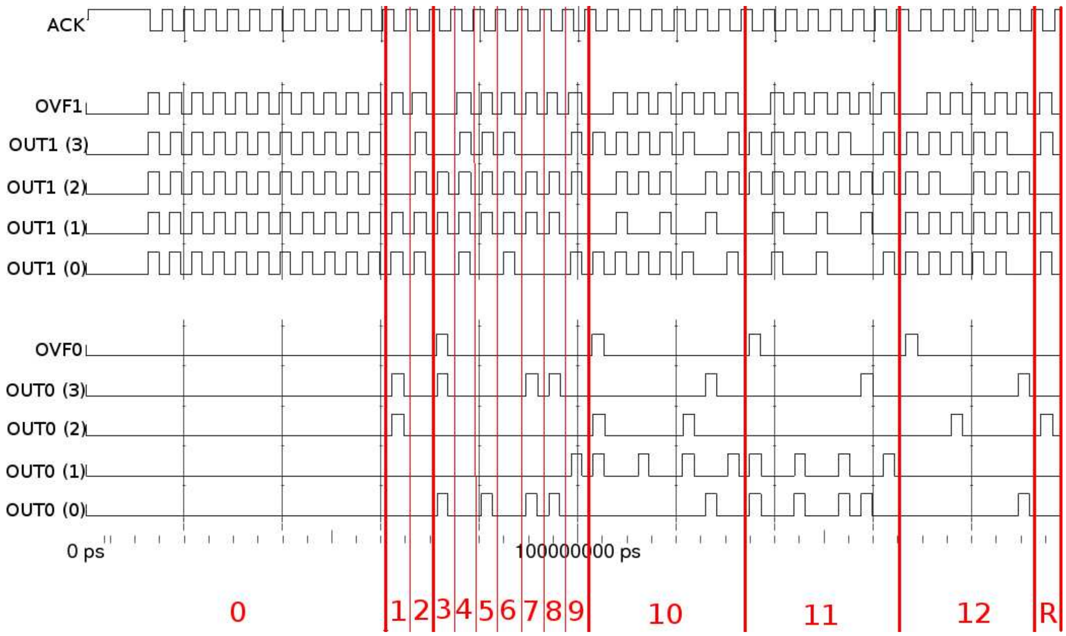

Fig. 9. Boolean microprocessor architecture. The architecture is similar to the previous cases but simpler. Asynchronous registers are substituted with synchronization blocks (bottom right inset) that realize an asynchronous-like structure, and solve the feedback problem of QCA circuits. In bottom left inset the boolean memory cell is shown, used in the mixed Boolean-NCL and in the fully Boolean versions of the microprocessor.

In this way we have implemented again an asynchronous communication protocol, but the whole circuit has a lower complexity as there is not encoding and no handshaking.

We have tested the microprocessor using the division algorithm, also in this case. The results are shown in Figure 10. The waveforms are similar to those shown in Figure 8, but in this case there is no signal encoding. The performance are boosted with respect to the mixed logic case, as the execution of the algorithm requires only $28\mu s$ instead of $194\mu s$. The synchronization pulse is the ENABLE signal shown in Figure 10.

In the final columns of Table I the performance of this last version of the microprocessor are shown. A remarkable improvement in terms of speed, area and power is evident. In particular, the time execution of one instruction is 8 times smaller than in the mixed version, while the number of nanomagnets and the power dissipation are 3 times smaller. The clock wire dissipation, in particular, is also approximately four times smaller due to the reduced area and complexity. So it is clear from these results that this is a promising direction to work on in the future.

Table I. Microprocessor types comparison.

	NCL	Boolean-NCL	Boolean
Istruction execution time [μs]	5.35	4.41	0.546
Area (number of nanomagnets)	$4 \cdot 10^6$	$0.6 \cdot 10^6$	$0.2 \cdot 10^6$
Nanomagnets power dissipation [μW]	23.9	3.51	1.09
Clock power dissipation (Joule Effect) [μW]	39.99	7.95	1.86

To summarize, this solution greatly enhances the circuit performance, and at the same time provides a simple way to build any kind of NML circuit. However signal synchro-

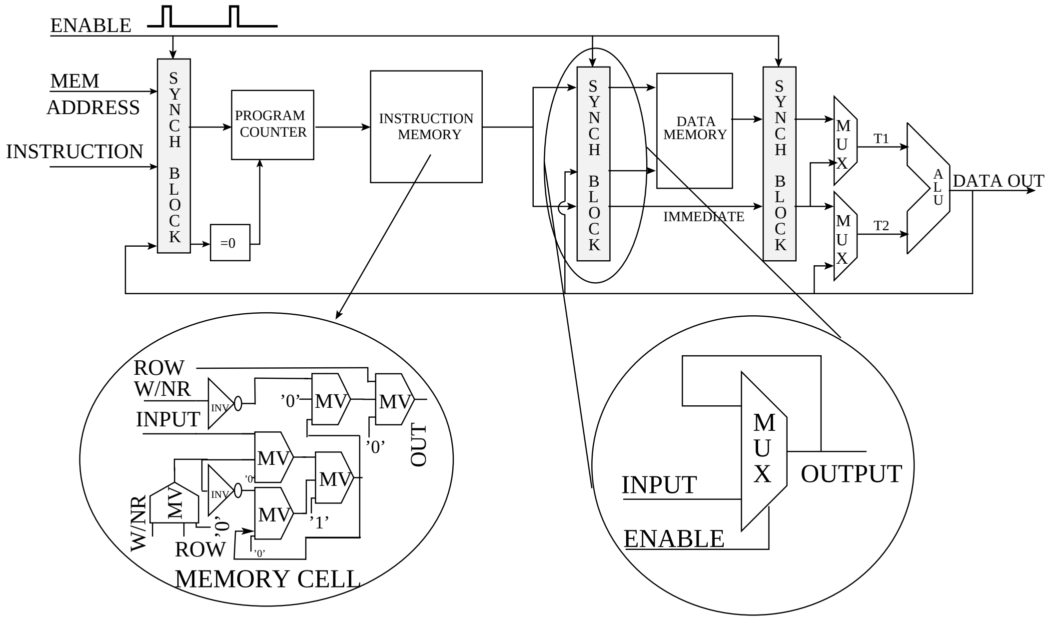

Fig. 10. Simulation results of the division algorithm executed on the pure Boolean microprocessor.

nization still remains a problem inside each boolean block. We were able to synchronize signals, because our model is an high level behavioral model, which does not take into account the real layout of complex interconnections. We are currently building an automatic circuit synthesizer, placer and router in order to obtain realistic representations of NML circuits. Considering the preliminary results we obtained, in pure boolean circuits the signals synchronization requirement can generate a big area overhead that partially cancels the advantages of this approach. This is something that we are still investigating, but for now we can state that the approach here proposed is the best solution ever proposed in literature.

6. CONCLUSIONS AND FUTURE WORKS

We have demonstrated that Nanomagnetic Logic Circuits technology is best suited for pure combinational circuits. Feedback signals need complex solutions, and slow-down circuits operations, unless a technological solution is adopted as, for example, using electric interconnections for long range data transmission. The only way to solve NML problems, from the logic point of view, is to use asynchronous logic. We completed a detailed analysis of asynchronous circuits implemented using QCA technology. We have performed the analysis comparing three different types of logic: A full NCL logic, a mixed Boolean-NCL logic and a full Boolean logic.

As expected the use of NCL logic completely solves both the synchronization problem and the feedback signals criticality. However the pay-back in terms of area and power is too high. A mixed Boolean-NCL solution is a very good compromise between performance and circuit feasibility. The performance are lower than the in the full Boolean case but notably better than in the full NCL solution. At the same time signal synchronization is required, but it can be done more easily, because it is limited to some regular combinational part of the circuit.

The full Boolean solution instead grants a huge saving in terms of speed, area and power consumption. However signal synchronization remains troublesome. Given the delay of the biggest loop inside the circuit, the inputs must be updated according to that delay. Furthermore a special block (latch) is required to slow down the operations of the fastest loops inside the circuit, for synchronization purposes. This block is similar to the asynchronous register in NCL logic which handles the communication protocol, but it this case there is not handshake. Therefore the solution that we have proposed is an "asynchronous-like" one. We believe that the results and solutions discussed in this work soundly enhance the knowledge of QCA circuits and will be useful as guidelines for the future development of this technology.

REFERENCES

Alam, M., J.DeAngelis, Putney, M., Hu, X., Porod, W., Niemier, M., and Bernstein, G. 2007. Clock scheme for nanomagnet qca. In *International Conference on Nanotechnology*. IEEE, Hong Kong, 403 – 408.

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

Becherer, M., Kiermaier, J., Csaba, G., Rezgani, J., Yilmaz, C., Osswald, P., Lugli, P., and Schmitt-Landsiedel, D. 2009. Characterizing magnetic field-coupled computing devices by the extraordinary hall-effect. In *Proceedings European Solid State Device Research Conference*. IEEE, Athens, Greece, 105 – 108.

Casu, M. R. and Macchiarulo, L. 2007. Adaptive latency-insensitive protocols. *IEEE Design & Test of Computers 24,* 5, 442 – 452.

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

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

Chung, W. J., Smith, B., and Lim, S. K. 2005. Qca physical design with crossing minimization. In *International Conference on Nanotechnology*. IEEE, Nagoya, Japan, 108 – 111.

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

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

Csaba, G. and Porod, W. 2010. Behavior of nanomagnet logic in the presence of thermal noise. In *International Workshop on Computational Electronics*. IEEE, Pisa, Italy, 1–4.

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

D., D., P., C., G., P., M., C., and D., P. 2009. Electrothermal modelling for eibj nanogap fabrication. *Electrochimica Acta 54*, 6003 – 6009.

Davis, A. and Nowick, S. 1998. *An introduction to asynchronous circuit design.* Marcel Dekker, In A. Kent and J. G. Williams, editors, The Encyclopedia of Computer Science and Technology, New York.

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

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

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

Graziano, M., Vacca, M., Chiolerio, A., and Zamboni, M. A ncl-hdl snake-clock based magnetic qca architecture. *IEEE Transaction on Nanotechnology* 10, DOI:10.1109/TNANO.2011.2118229.

Graziano, M., Vacca, M., and Zamboni, M. 2011. *Magnetic QCA Design: Modeling, Simulation and Circuits.* Cellular Automata - Innovative Modelling for Science and Engineering, In-

Tech, http://www.intechopen.com/articles/show/title/magnetic-qca-design-modeling-simulation-and-circuits.

Huang, J. and Lombardi, F. 2007. *Design and Test of Digital Circuits by Quantum-Dot Cellular Automata*. Artech House Publishers, Boston/London.

Imre, A. 2005. Experimental study of nanomagnets for quantum-dot cellular automata(mqca)logic applications. Ph.D. thesis, University of Notre Dame, Notre Dame, Indiana.

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

Kummamuru, R., Orlov, A., Ramasubramaniam, R., Lent, C., Bernstein, G., and Snider, G. 2003. Operation of a quantum-dot cellular automata (qca) shift register and analysis of errors. *IEEE Transaction On Electron Devices 50*, 1906–1913.

Lent, C. S., Tougaw, P., Porod, W., and Bernstein, G. 1993. Quantum cellular automata. *Nanotechnology 4*, 49–57.

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

Martina, M. and Masera, G. 2010. Turbo noc: A framework for the design of network-on-chip-based turbo decoder architectures. *IEEE Transaction on Circuits and Systems I 57,* 10, 2776 – 2789.

Modelsim. Mentor graphics. http://www.modelsim.com.

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

Orlov, A., Imre, A., Csaba, G., Ji, L., Porod, W., and Bernstein, G. 2008. Magnetic quantum-dot cellular automata: Recent developments and prospects. *ASP Journal of Nanoelectronics and Optoelectronics 3,* 1, 55–68.

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

Pulecio, J. and Bhanja, S. 2010. Magnetic cellular automata coplanar cross wire systems. *Journal Applied Physics 107,* 3.

Pulimeno, A., Graziano, M., Abrardi, C., Demarchi, D., and Piccinini, G. 2011. A write-in system based on electric fields for molecular qca. In *2011 IEEE International NanoElectronics Conference (INEC)*. IEEE, Tao-Yuan, Taiwan, 1–2.

Rizos, N., Omar, M., Lugli, P., Csaba, G., Becherer, M., and Schmitt-Landsiedel, D. 2009. Clocking schemes for field coupled devices from magnetic multilayers. In *International Workshop on Computational Electronics*. IEEE, Beijin, China, 1–4.

S. Henderson, E.W.Johnson, J. and Tourgaw, P. 2004. Incorporating standard cmos design process methodologies into the qca logic design process. *IEEE Transaction on Nanotechnology 3,* 1, 2–9.

Semiconductor Industry Association 2008. International technology roadmap of semiconductors, 2008 update. http://public.itrs.net.

Sparso, J. and Furber, S. 2011. *Principles of asynchronous circuit design - A systems perspective*. Kluwer Academic Publishers, Boston/Dordrecht/London.

Vacca, M. 2008. Nanoarchitectures based on magnetic qca. M.S. thesis, Politecnico di Torino.

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

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

Zhang, R., Gupta, P., and N.K., J. 2005. Synthesis of majority and minority networks and its applications to qca, tpl and set based nanotechnologies. In *International Conference on VLSI Design*. IEEE, Kolkata, India, 229 – 234.