Magnetic-Memory based Dynamically Reconfigurable Array

Abstract—This paper presents a dynamically reconfigurable array programmable by magnetic memory. Magnetic tunneling junction memory cells are fully compatible with standard CMOS circuits, and can provide non-volatility with cell areas and access speeds comparable to those of conventional static memories.

The architecture of the reconfigurable array has been organized as a one-dimensional array of up to 16 reconfigurable ALUs, whose configuration bits are stored in a double-context magnetic memory. This type of coarse-grain array is more efficient to execute data-computation algorithms and to exploit operation-level parallelism than typical fine-grained architectures.

The implementation results show that the inclusion of a second context plane for the programmable memory implies only a 20% area overhead therefore confirming that the magnetic memory technology can effectively support very efficient runtime reconfiguration. The results also show that this type of coarse-grained reconfigurable ALU can provide an area-efficiency gain of up to 50:1 over a fine-grained (FPGA) implementation for the same functionality.

Coarse- Grain Reconfigurable Array; CGRA; MRAM

I. INTRODUCTION

To date, most reconfigurable devices available use SRAM based programmable memories and consequently the configuration information is volatile. The use of magnetic tunneling junction (MTJ) cells in run-time reconfigurable systems is a promising technological alternative because MTJ technology supports realizing a nonvolatile programmable memory with area and access times comparable to conventional static memories and without the complexity inherent in the process of flash memories [1].

Therefore this has become an emergent area of research and, in recent years, a number of preliminary works have been published on MTJ-based FPGAs [1], [2], [3], [4], [5], [6]. However, all these works have addressed fine-grained FPGAs, following the mainstream trend for most existing (SRAM based) reconfigurable hardware architectures.

This work proposes, alternatively, a MTJ-based coarse-grained reconfigurable array. Coarse-grained arrays, where each reconfigurable element computes on 4-bit or larger input words, are more suitable to execute data-oriented algorithms and are more able to exploit larger amounts of operation-level parallelism than common fine-grained architectures.

Another interesting feature of the memory structures for the MTJ cells is that they can be extended to efficiently provide multi-context reconfiguration, without the need to fully replicate each memory cell. Multi-context configuration memories use multiple memory bits for each programming bit location, which act as multiple planes of configuration information that can be rapidly exchanged during runtime, therefore providing a very efficient means for run-time reconfigurability. By substantially reducing the overhead for configurability, coarse-grain architectures are also more apt (than fine-grain structures) to efficiently exploit run-time reconfiguration and therefore to take advantage of multi-context configuration memories.

A coarse-grained reconfigurable one-dimensional array based on magnetic memory technology has been designed and implemented. The current array implementation supports up to 16 reconfigurable ALUs, with 4-bits (or larger) operands. The rALU operation is defined by the state of the MTJ cells of the reconfiguration memory.

A prototype has been designed and implemented, with 4-bits reconfigurable ALUs with a double-context MTJ-based configuration memory, using the most commonly used Field Induced Magnetic Switching (FIMS) writing technology. The prototype has been fabricated using a CMOS 0.35μm standard process, with 4 metal levels. The prototype has been successfully tested and the electronic of the memory cells has been fully validated.

This presentation is organized as follows. Section II summarizes the basic operation of the magnetic tunneling junctions and reviews related work on MTJ-based FPGAs. Section III presents the main concepts behind existing coarse-grained reconfigurable architectures. Section IV describes the coarse-grained architecture implemented and details the basic building blocks that constitute the rALU. Section V describes the test infrastructure developed and presents the experimental results obtained. Finally, some conclusions and directions for future work are presented.

II. MAGNETIC JUNCTIONS

Magnetic tunneling junctions (MTJ) are fully compatible with standard CMOS processes and require only a few additional steps after the standard CMOS process. Therefore, the most common magnetic memory cells consist of MTJs vertically integrated over the silicon CMOS, as shown in figure 1.
The MTJ cell is made of two thin ferromagnetic layers separated by an ultra-thin non-magnetic oxide layer [2]. One of the ferromagnetic layers is pinned and used as reference, and the other is free and its orientation can be changed. The relative magnetic orientation of the two layers defines the two different states of the memory cell.

The memory state is read by evaluating the equivalent resistance of the junction. This requires a current generator to send a current through the junction and a circuit structure to measure the voltages at the nodes of the MTJ.

To write a new value to the memory cell, a magnetic field must be applied through the junction in order to change the magnetic orientation of the free layer. The strength and nature of the required magnetic field depends on the writing approach. Currently, there exist 3 writing approaches known as Field Induced Magnetic Switching (FIMS) [2], Thermally Assisted Switching (TAS) [4] and Spin Transfer Torque (STT) [6].

The FIMS technique is the more robust and therefore has been mostly used, namely in production Magnetic RAMs (MRAM) [18]. It has however some drawbacks [10], such as a susceptibility to soft errors due to write selectivity, a lower scalability and a high current consumption, which the TAS and STT approaches try to overcome.

A number of works has been published on the use of magnetic memories for programmable devices, but these have been so far exclusively employed as elements on fine-grained FPGA alike solutions, such as Look-Up Tables (LUT). To our knowledge, no other works have been published on MTJ-based coarse-grained reconfigurable arrays.

The published proposals for fine-grained configurable devices using magnetic configuration memories differ essentially on the writing technique supported (FIMS [1, 2, 3], TAS [4, 5] or STT [6]), on the use of the MTJ as pull up or pull down and on the support or not for multiple context planes.

The use of two MTJ magnetically oriented in opposite directions in a single Flip-Flop was initially proposed in [7] to evaluate the state of the programmable memory. This structure was further refined [8], [9] and the Unbalanced Flip-Flop (UFF) proposed in [9] has since been adopted as the foundation for magnetic configuration memory.

A TAS-MRAM based coarse-grain reconfigurable array was also developed by our team during 2009-2010 and its electronic components were successfully validated [15] [16].

For this new version of the reconfigurable system we opted to use FIMS-based memory cells because MTJs supporting the FIMS writing technique require a lesser complex manufacturing process than the one required for MTJs supporting the TAS writing technique.

Due to MTJ’s process and material parameters dispersion, it is unsafe to set a predefined value for the intensity of the currents that generate the magnetic fields responsible for re-orienting the free layer, before the memory cells are fully characterized for the given technology. Therefore, the implemented array includes a current tuning mechanism to enable the operation of the system even in the presence of large outliers.

III. COARSE GRAINED RECONFIGURABLE ARCHITECTURES

Coarse grain reconfigurable arrays (RA), with functional units that process multi-bit operands, can offer important advantages when compared with common fine-grained solutions, such as FPGAs.

They also offer a very significant reduction of the configuration memory size and thus substantially reduce the necessary configuration time. This considerable reduction of the configuration overhead for configurability, makes coarse-grain architectures more suitable to take advantage of multi-context configuration memories and exploit context-switching run-time reconfiguration.

Since the 1990s, a significant number of architectures have been proposed (see [11], [12], [13], for a review of the most relevant). These architectures vary on the interconnect structure, granularity and/or reconfiguration model. Practically all of the proposed architectures are SRAM-based and none of these is MRAM-based.

Most of the proposed architectures are organized as a 2-dimensional array or mesh, although some of the works, see for example [14], have demonstrated that 1-dimensional coarse-grained linear array architectures can provide high-performance for specific application domains.

The width of the data-path varied from 2 bits up to 128 bits. Some architectures support a single size, while others support different data-path widths and others permit bundling several processing elements to compose larger width data-path operators.

Some of the systems are dynamically reconfigurable, while others can only be statically configured. Static configuration refers to the configuration model where the configuration is loaded at the beginning of the execution and stays the same during the whole computation. To load a new configuration, the execution must stop. Dynamic reconfiguration means that a new reconfiguration can be loaded while the application is running, i.e. the circuit can be (partially) modified during runtime.

Considering that MTJ-based configuration is still an emergent technology and that 1-dimensional arrays can provide high-performance, with a not too complex architecture organization, we opted to implement this reconfigurable system.
as a linear coarse-grained array structure in order to both provide an effective performance and permit full characterization of the new run-time reconfigurable technology.

The memory circuits for the MTJ cells have been extended to support the inclusion of one additional pair of magnetic junctions per memory bit cell in order to effectively support two context planes. The sense amplifiers of the memory cells allow writing the MTJ during circuit run-time, and are therefore able to provide run-time reconfiguration directly and efficiently. The proposed architecture can therefore support dynamic reconfiguration with low overhead and an extremely fast context switch.

IV. RECONFIGURABLE ARRAY

The reconfigurable architecture implemented in this work consists of one linear array of reconfigurable ALU (rALU), which builds up and improves on the architecture we reported in [15], [16].

Each rALU (current implementation) includes one configuration memory, the ALU datapath unit and local control logic, the bidirectional current generators and the transmission gates that control the accesses to the buses.

The rALUs now support up to 16 different arithmetic and logic operations and are able to execute efficiently both binary and decimal arithmetic (BCD – Binary Coded Decimal). The operation to be executed is defined by a 4-bit configuration word stored in the associated configuration memory bits. The configuration memory includes two context configuration words that can be exchanged during run-time. Table 1 shows the set of arithmetic and logic functions currently implemented in the rALUs.

The rALU operands are 4-bit wide, but the rALU can be easily scaled to support larger bit-widths. As referred, the configuration memory of any rALU can be rewritten while the circuit is in execution, and therefore a new reconfiguration can be loaded into any rALU while the current application is running, effectively supporting run-time reconfigurability.

<table>
<thead>
<tr>
<th>Configuration Bits</th>
<th>ALU operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>A and B</td>
</tr>
<tr>
<td>0001</td>
<td>A or B</td>
</tr>
<tr>
<td>0010</td>
<td>A xor B</td>
</tr>
<tr>
<td>0011</td>
<td>A xnor B</td>
</tr>
<tr>
<td>0100</td>
<td>A</td>
</tr>
<tr>
<td>0101</td>
<td>Not A</td>
</tr>
<tr>
<td>0110</td>
<td>B</td>
</tr>
<tr>
<td>0111</td>
<td>Not B</td>
</tr>
<tr>
<td>1000</td>
<td>A + b</td>
</tr>
<tr>
<td>1001</td>
<td>B - A</td>
</tr>
<tr>
<td>1010</td>
<td>A + B + Cin</td>
</tr>
<tr>
<td>1011</td>
<td>B – A - Borrow</td>
</tr>
<tr>
<td>1100</td>
<td>A + B (BCD)</td>
</tr>
<tr>
<td>1101</td>
<td>B – A (BCD)</td>
</tr>
<tr>
<td>1110</td>
<td>A + B + Cin (BCD)</td>
</tr>
<tr>
<td>1111</td>
<td>B – A – Borrow (BCD)</td>
</tr>
</tbody>
</table>

As shown in figure 2, there are 4 global data buses, 2 for data inputs (operands A and B) and 2 for data outputs (result S and carry out). 16 selector signals, one for each rALU, ensure exclusive access to the global data buses for each rALU.

The configuration memory is made of an array of identical FIMS-MRAM cells. A global configuration bus, shown in figure 3, carries the data that will be stored on the MTJ to be used later on for configuration.

A. Configuration memory

The reconfiguration memory is based on an unbalanced flip-flop (UFF) that is used as a sense amplifier to evaluate the MTJ state. The memory cell selected for the FIMS-based implementation builds up on the cell initially proposed in [2] and [9]. This cell was extended to support two configuration planes (see figure 4).

Each memory cell has:

- One Unbalanced Flip Flop (UFF) used as a sense amplifier.
- Four MTJ (MTJ1(0), MTJ1(1), MTJ2(1) and MTJ2(0)) for the 2 context planes configuration.
- Four nMOS switches, one for each MTJ.

The flip-flop is implemented with two cross-coupled inverters (MP0, MP1, MN0 and MN1 transistors) and one nMOS switch (transistor MN2).

The memory state is defined by the pair of MTJ, which are polarized in complementary fashion. During the reading phase, the MN2 switch is shorted and the UFF is pulled to a meta-stable state, whose value depends on the equivalent resistances of the MTJ. When the Read signal is deasserted, the MN2 switch opens and the UFF settles to the correspondent stable state. Afterwards, new information can be stored into the MTJ without altering the value stored in the UFF.
The support of two context planes requires the inclusion of 4 magnetic junctions per memory bit cell, one pair for each plane. Junctions MTJ1(0)-MTJ2(0) are associated to context 0 and junctions MTJ1(1)-MTJ2(1) are associated to context 1.

The switches associated with each pair of junctions are selectively closed depending on the context information that the circuit must access.

From the moment when a pair of switches is closed and the other pair is open, reading the magnetic information stored in the magnetic junctions is done as in the case of a memory cell that supports only one context.

As new information can be stored into the MTJ without affecting the UFF state, a new configuration plane can be written to the MTJs while the rALU is executing. Therefore, run-time reconfiguration is directly supported by the UFF circuitry.

B. Bidirectional Current Generator

As aforementioned, an external magnetic field is necessary to change the magnetic orientation of the MTJ’s free layer. Since, the UFF depends for it functioning on the existence of a couple of MTJ polarized in complementary fashion, it is necessary to be able to generate a bidirectional magnetic field.

For this purpose, a bidirectional current generator capable of delivering a substantial amount of current up to the order of 25 mA was developed. Normally, one bidirectional current generator should be assigned to each pair of MTJ, and the generator should be implemented simply with a set of pull-up and pull-down transistors controlled by digital control signals (see [1], [3], [4], [5]).

However and in order to simplify the post-CMOS process, in this prototype a bidirectional current generator has been assigned to each MTJ (instead of one to each pair of MTJ). Therefore, there are 4 bidirectional current generators (two for each of the 2 context planes) assigned to each bit of the reconfiguration memory.

In order to better characterize the MTJ behavior, we opted to include a current tuning mechanism and therefore the generator has a pair of a digital controlled current source in association with a digital controlled current sink.

This circuit is shown in figure 5. The bias circuitry (Rbias1 and Rbias2) is common to all generators, only the current source and sink circuitry (MP1-2 and MN1-2) is repeated for each generator.

The switches S1 and S2 operate in complementary fashion. Their purpose is to turn on and turn off the current source-sink (MP1-MN2) as required by the logic responsible for the configuration. The switches S3 and S4 operate in complementary fashion and are responsible for turning off or on the current source-sink (MP2-MN1).

This circuit works in one of 3 states:

- When the S1 and S4 are closed and S2 and S3 are opened, an electrical current flows starting in MP1’s source terminal and ending in MN2’s source terminal (write state).
- When the S2 and S3 are closed and S1 and S4 are opened, an electrical current flows starting in MP2’s source terminal and ending in MN1’s source terminal (complementary write state).
- No electrical current flows when S1 and S3 are opened and S2 and S4 are closed.

The Rbias1 and Rbias2 provide the current tuning mechanism. These 2 resistances are external and can be employed to change the intensity of the currents that run across the current sources (MP1 and MP2) and across the current sinks (MN1 and MN2) respectively.
C. ALU and local control logic

As referred above, each rALU (current implementation) has a 4 bit arithmetic logic unit (ALU) that performs the arithmetic and logic operations shown in Table 1.

Furthermore, each rALU has also local control logic responsible for:

- Selecting the operation to be executed by the ALU.
- Opening and closing the switches (S1, S2, S3 and S4) that turn on/off the current sources/sinks that generate the magnetic fields responsible for re-orienting the free layer of each MTJ belonging to a memory cell.
- Selectively opening and closing the switches associated with each pair of MTJ (for each memory cell) in response to a request for changing the context plane.
- Drive the test infrastructure in response to a test request.

V. EXPERIMENTAL RESULTS

The test of the chip prototype was done in two steps. In step one, the memory cell electronics (including context plane switching) was tested before any back end fabrication of the MTJ cells over the CMOS wafer is done. In the second step, the whole system after back end fabrication will be tested.

For testing purposes a dedicated PCB was designed and manufactured. This PCB was attached to a FPGA based board (Digilent D2SB) that was employed as a programmable pattern generator. Finally, a digital oscilloscope/logic analyzer was attached to specific output pins of the dedicated PCB.

During the design and implementation of the chip prototype a module for testing purposes was specifically built. This module replicates a modified memory cell (see Figure 6). Each MTJ attached to the test module was emulated with a dedicated resistor.

Figures 7-9 show the test module responses to a set of stimuli with MTJ1 emulated by a resistor set to 1300 Ω and MTJ2 emulated with a resistor set to 1000Ω. The response of the test module after reading a logic ‘1’ stored in the pair of MTJ (MTJ1-MTJ2) is shown in figure 7 (the signal “Plane” is set to logic ‘1’).

Table 2. Area occupied by the memory cells (2 versus 1 context planes)

<table>
<thead>
<tr>
<th>Block</th>
<th>1 context plane (Kλ^2)</th>
<th>2 context plane (Kλ^2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>UFF</td>
<td>25.3</td>
<td>25.3</td>
</tr>
<tr>
<td>1 bit memory cell (UFF+buffers)</td>
<td>53.4</td>
<td>65.4</td>
</tr>
<tr>
<td>4-bit memory</td>
<td>227</td>
<td>276</td>
</tr>
</tbody>
</table>

Figure 8 shows the response of the test module after reading a logic ‘0’ (the signal “Plane” is set to logic ‘0’). Figure 9 shows the response of the test module to a plane context switch. The test module switches from a logic ‘0’ to a logic ‘1’ (the signal “Plane” is set to logic ‘1’) and later on it switches from a logic ‘1’ to a logic ‘0’ (the signal “Plane” is set to logic ‘0’) in response to a Read signal.

Table 2 shows the area occupied by the implemented FIMS-based memory cells, and compares the area requirements of the cells with support for 2 context planes with those for memory cells supporting only 1 context. As expected, the inclusion of a second context plane incurs only a small area overhead, around 21%.

One rALU (excluding the bidirectional current generator) requires an area of about 1.43Mλ² (the 4-bit memory occupies 0.27 Mλ² and the ALU + control logic occupies ≈1.16 Mλ²). An rALU with the same functionality has been implemented using a fine-grained FPGA. Using the area reported in [17] for a Virtex-2 CLB, it can be estimated that the same functionality in a Virtex-2 requires an area of around 74.6Mλ². This indicates an area reduction factor better than 50:1 for the rALU in comparison with an implementation using fine-grained bit-level primitives.

VI. CONCLUSIONS AND FUTURE WORK

A scaled-down prototype of one linear array of 4-bits reconfigurable ALUs, with a double-context FIMS-MTJ based configuration memory, has been designed and implemented in a CMOS 0.35µm standard process, with 4 metal levels. This prototype has been successfully tested and the electronic of the memory cells has been fully validated.

The implementation results confirm that the inclusion of a second context plane for the programmable magnetic memory implies only a small area overhead (without the need to fully replicate the memory cell) and therefore that the use of magnetic memory technology for the array configuration can provide efficient run-time reconfiguration.

The results also show that the rALU implemented may provide an area-efficiency gain of up to 50:1 when compared with the implementation of the same functionality in fine-grained array. This result further confirms that coarse-grained reconfigurable arrays can be significantly more area-efficient than fine-grained bit-level FPGAs to execute data-oriented algorithms.
In the near future, we plan to extend the current array architecture to use processing elements with larger operand words and to evaluate more sophisticated array organizations.

ACKNOWLEDGMENT

This work was supported by national funds through FCT, Fundação para a Ciência e a Tecnologia, under projects PEst-OE/EEI/LA0021/2011 and PhD grant SFRH/BD/62233/2009.

The research, development and fabrication of the MTJ elements, in the scope of this work, are being done at INESC-MN. The authors acknowledge the work and cooperation of Ricardo Ferreira, Susana Freitas and Paulo Freitas of the INESC-MN research team.

REFERENCES


