A Circuit and Architecture Codesign Approach for a Hybrid CMOS–STTRAM Nonvolatile FPGA

Somnath Paul, *Student Member, IEEE*, Saibal Mukhopadhyay, *Member, IEEE*,
and Swarup Bhunia, *Senior Member, IEEE*

***Abstract*—Research efforts to develop a novel memory technology that combines the desired traits of nonvolatility, high endurance, high speed, and low power have resulted in the emergence of spin–torque transfer RAM (STTRAM) as a promising next-generation universal memory. Although industrial efforts have been made to design efficient embedded memory arrays using STTRAM, the prospect of developing a nonvolatile field-programmable gate array (FPGA) framework with STTRAM exploiting its high integration density remains largely unexplored. In this paper, we propose a novel CMOS–STTRAM hybrid FPGA framework, identify the key design challenges, and propose optimization techniques at circuit, architecture, and application mapping levels. We show that intrinsic properties of STTRAM that distinguish it from conventional static RAM (SRAM), such as asymmetric readout power, where a cell storing "0" has 5× less read power than a cell storing "1", can be leveraged to skew lookup table contents for FPGA power reduction. We also argue that the proposed framework should operate on static voltage-sensing-based logic evaluation. We identify static power dissipation during logic evaluation and read noise margin as key design concerns and present an optimized resistor–divider design for voltage sensing to reduce static power and noise margin. Finally, we investigate the effectiveness of Shannon-decomposition-based supply gating to reduce static power. Simulation results show improvement of 44.39% in logic area and 22.28% in delay of a configurable logic block (CLB) and average improvement of 16.1% dynamic power over a conventional CMOS FPGA design for a set of benchmark circuits.**

***Index Terms*—Emerging memory technologies, nonvolatile field-programmable gate array (FPGA), Shannon decomposition, spin torque transfer RAM (STTRAM).**

I. Introduction

Modern reconfigurable computing platforms offer major advantages such as reduced design cost, rapid prototyping, and a generic hardware for mapping arbitrary applications [1], [10]. Since their introduction, field-programmable gate arrays (FPGAs) have become the most popular hardware reconfigurable platform. Present FPGA platforms can be broadly divided into two classes: 1) SRAM-based FPGAs (such as Xilinx Virtex), which employ SRAM cells to store the configuration and 2) flash and antifuse-based FPGAs (such as Actel and Quicklogic), which use nonvolatile memory for storing configuration [23]. While the former enjoys the advantage of standard fabrication flow and the capability for multiple reconfiguration, the latter offers the advantages of fast start-up and minimal reconfiguration power [2]. Clearly, the most desirable FPGA platform is the one that combines the benefit of both worlds. FPGA frameworks can, therefore, benefit greatly from emerging memory technologies that combine the best features of current volatile and nonvolatile memories in a fabrication technology compatible with CMOS process flows.

On the basis of the salient features of the present and emerging memory technologies reported in [3], Fig. 1 shows the advantages and disadvantages for each of them. Note that the comparison parameters such as read time, write time, retention time, write cycles, etc., vary widely from one memory technology to another [3]. For example, the read time for SRAM is of order of picoseconds, while the read times for other memory technologies such as dynamic RAM (DRAM) are of the order of nanoseconds. Thus, for relative comparison purposes, we have plotted the logarithm values of the corresponding parameters. The range 0–2.5 in Fig. 1(a) and the range 0–15 in Fig. 1(b) are determined by the logarithm of the range of read time and write cycle values for the memory technologies being compared in Fig. 1. As for the other parameters such as cell area, read voltage, and write voltage, the relative values have been normalized by a factor so that they can be represented within the range 0–2.5 and 0–15 in Fig. 1(a) and (b), respectively. Values for these parameters are, therefore, represented on a linear scale.

As may be observed from Fig. 1(a) and (b), modern flash memory (NAND or NOR) provides a nonvolatile alternative at the cost of higher read and write power dissipation. Moreover, the number of write cycles that the flash memory may reliably undergo is limited to only 10^5. Therefore, in order to find a viable nonvolatile alternative, several research efforts [4]–[6] have been undertaken in recent years. These research efforts have resulted in the development of new nonvolatile memory technologies such as: 1) one transistor and one capacitor (1T1C) ferroelectric RAM (FeRAM) [4]; 2) one transistor and one resistor (1T1R) spin torque transfer magnetic RAM (STTRAM) [5]; and 3) 1T1R phase-change RAM (PCRAM) [6], etc. However, as may be noted from Fig. 1, technologies such as PCRAM and FeRAM, though promising, have high read latencies and smaller number of write cycles compared to STTRAM. It is due to this high speed of operation along with unlimited endurance

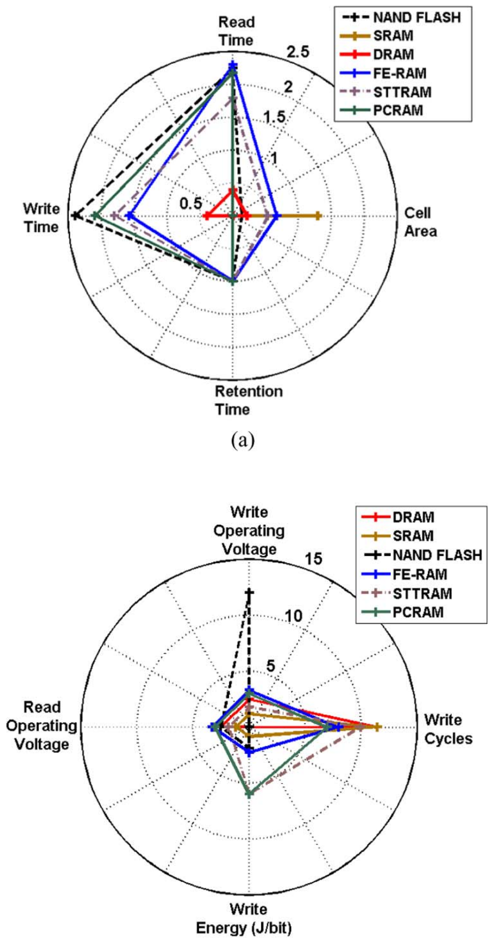
Manuscript received November 29, 2008; revised August 26, 2009 and October 22, 2009; accepted December 13, 2009. Date of publication February 8, 2010; date of current version May 11, 2011. The review of this paper was arranged by Associate Editor D. Hammerstrom.

S. Paul and S. Bhunia are with the Department of Electrical Engineering and Computer Science, Case Western Reserve University, Cleveland, OH 44106 USA (e-mail: appusom@gmail.com; skb21@case.edu).

S. Mukhopadhyay is with the Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: saibal@ece.gatech.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNANO.2010.2041555



(a)

(b)

Fig. 1. Comparison between baseline (SRAM, DRAM, and flash) and emerging memory technologies (FeRAM, STTRAM, and PCRAM). (a) Comparison on the basis of area and performance. (b) Comparison on the basis of energy and voltage.

and high integration density that STTRAM is being considered as the next-generation universal memory [7]. Although recent advances [8] in fabrication and novel memory architectures [9] have been reported for STTRAM arrays, due to the high-write-current requirement, implementation of STTRAM-based normal embedded memory has proved to be a challenging task. A high-write-current requirement is, however, of less concern when the memory is reconfigured only infrequently. This makes STTRAM particularly suitable as a storage media in FPGA frameworks, which are configured only once in a while.

Due to the volatile nature of the SRAM, configuration in FPGA is lost when the power is turned down. The configuration can be stored in an external programmable ROM (PROM) and again downloaded into the FPGA at start-up. Since the configuration memory is distributed throughout the device and the data are serially loaded as a shift register, configuration time might be high for larger designs (∼100 ms). An existing solution is to use flash memory as a nonvolatile storage to hold the configuration even when the power is turned down [11]. Actel ProASIC [12] and Xilinx Spartan-3AN [13] FPGAs employing such a solution are, therefore, ready to run at power-up. STTRAM is a fast-emerging nonvolatile alternative that outperforms existing flash technologies due to its higher endurance and low access latencies.

The nonvolatile resistance of an STTRAM device is either in high- or low-resistance state [14] and can be used to represent a binary value. The STTRAM device can, therefore, serve as a configuration bit for an FPGA [15]. The key challenge in an STTRAM-based reconfigurable framework is how to sense the resistance state for each configuration bit during normal functioning of the FPGA. Sense-amplifier-based data read-out mechanisms, commonly employed in embedded STTRAM memories, will incur large overhead when used in an FPGA framework. Previous works have, therefore, explored the possibility of a dynamic sensing approach [15], which incurs minimal design overhead for sensing the stored configuration. However, a spatial computing framework such as FPGA does not lend itself easily to the use of dynamic sensing approach at each configurable logic block (CLB) of the framework. It is, therefore, necessary to come up with an efficient circuit and architecture codesign approach that would: 1) preserve the normal functionality of the conventional FPGA framework; 2) exploit the advantages of STTRAM storage such as nonvolatility and high integration density; and 3) incur minimum design overhead. In context of the aforementioned challenges, we have explored circuit- and architecture-level optimization techniques and developed appropriate design mapping techniques for a CMOS–STTRAM hybrid FPGA. In particular, the paper makes the following contributions.

1) It investigates a CMOS–STTRAM hybrid *nonvolatile* FPGA platform that leverages on the high integration density of the STTRAM process. It identifies dynamic and static current dissipation and read noise margin as the key design challenges for the proposed CMOS–STTRAM hybrid platform. These challenges are addressed using efficient circuit and architecture codesign approaches and application mapping techniques.
2) It proposes the use of an STTRAM device in a resistor–divider configuration for static sensing of the logic value stored in the STTRAM device.
3) Since the STTRAM cell in a resistor–divider configuration has asymmetric power dissipation depending on the logic value stored, the paper proposes a preferential mapping approach in order to skew the lookup table (LUT) contents.
4) It investigates a Shannon-decomposition-based hardware/software codesign approach for static power reduction.

The rest of the paper is organized as follows. Section II explores the existing nonvolatile FPGA architectures and discusses the motivation for a novel CMOS–STTRAM hybrid FPGA framework. Section III explores the circuit and architectural aspects of the proposed STTRAM FPGA design. Section IV suggests appropriate changes to the application mapping flow in order to reduce the static and dynamic current consumption of the proposed CMOS–STTRAM hybrid FPGA. Section V discusses the results obtained from our simulation. Finally, a conclusion is presented in Section VI.

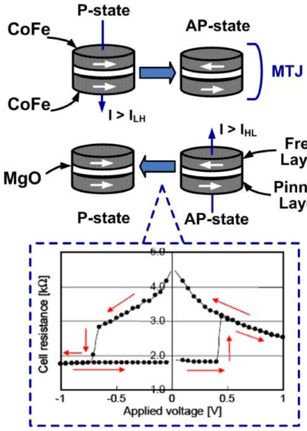


Fig. 2. Change in magnetization states for an MTJ on application of current. Callout shows a representative *R–V* curve along with the dependence of the TMR ratio on the bias voltage [7].

II. Background

A. STTRAM Device Operation

The basic building block of an STTRAM cell is the magnetic tunneling junction (MTJ) (see Fig. 2). Each MTJ consists of two ferromagnetic layers (typically, CoFe) separated by a very thin tunneling dielectric film (typically, crystallized MgO). Magnetization in one of the layers (referred to as pinned layer) is fixed in one direction by coupling to an antiferromagnetic layer (such as PtMn) [7]. The other ferromagnetic layer (referred to as a free layer) is used for information storage. The direction of magnetization of free layer can be controlled by the injection of spin-polarized electrons. Hence, the MTJ can be switched between two stable magnetic states with high (R_{AP} or R_H) or low (R_P or R_L) resistances. One of the quality metrics for an MTJ device is the tunneling magnetoresistance (TMR) ratio [14], defined as $(R_H - R_L)/R_L$. An MTJ cell with a high TMR ratio is desirable in order to easily distinguish between the two states ($\Delta R = R_H - R_L$). It may be noted that recent advances in STTRAM technology has been able to achieve a TMR ratio as high as 470% [9]. As observed from Fig. 2, the write current for an MTJ cell is required to be larger than the switching threshold current (I_{HL} or I_{LH}) in order to switch the magnetization of the *free layer* from antiparallel to parallel spin, or *vice versa*. Ongoing research efforts in companies such as Sony [7], Hitachi [8], and Samsung [17] have been successful in reducing this write current below 100 μA. Parallel research efforts in academia [18]–[20] have attempted to address the reliability challenges associated with unwanted writes during an STTRAM read operation.

B. Existing Architectures for a Nonvolatile FPGA

A typical STTRAM memory cell consists of a transistor connected in series with an MTJ. When a particular cell is selected, a high or low current flows through the cell, depending on whether it is in a low- or high-resistance state. The scheme, though suitable for large embedded arrays, incurs significant overhead for the case of small 1-D LUTs present in the CLB of FPGAs. A smarter scheme for sensing the resistance of the MTJs in case of FPGAs has been proposed in [15] and [16]. The principle idea is to use two MTJ elements for storing one-bit information, and then, employing a sense amp for reading out the stored configuration. This dynamic sensing scheme requires the latch of the sense amp to be precharged before evaluation. However, in a spatial computing framework such as an FPGA, the time for evaluation of a particular CLB is not fixed and depends on the path delay. Moreover, it is possible that a CLB is evaluated more than once in each clock cycle, thereby questioning the viability of the dynamic sensing scheme. Another interesting solution to realize the nonvolatile FPGA has been presented in [21], where each CLB is realized using macrocells employing two-level threshold logic. The principle idea is to incorporate a magnetoelectronic device [referred to as a hybrid hall element (HHE)] in each macrocell and configure each macrocell such that it produces a high voltage only when sufficient number of signals are active at its input . In order to reduce the power consumption, the authors have proposed a dynamic sensing scheme where a read signal is used to gate off the bias current required to generate the output voltage. A clock-based evaluation approach is proposed in [21] for each macrocell, which, we feel, will unduely complicate the FPGA design.

Instead of a dynamic sensing approach, a static sensing approach, as presented in [22], would be beneficial for the STTRAM-based LUT implementation. The scheme, as proposed in [22], relies on the fact that a STTRAM technology with a high TMR ratio allows information to be stored in the form of a resistor–divider configuration. Depending on whether the low or the high resistance is in contact with V_{dd}, the output logic level would be either high or low, respectively. However, even for a TMR of 200%, the maximum and minimum voltage levels attained at the output of the resistor–divider for a supply of 0.8 V are 0.525 and 0.175 V, respectively. The design as presented in [22] increases the noise margin of the resistor–divider configuration by series insertion of p and n transistors, biased with 0 and V_{dd}, respectively. Using the aforementioned resistor–divider configuration as a nonvolatile storage, one can use the STTRAM as a backup storage device for each CMOS SRAM configuration bit [see Fig. 3(b)]. An FPGA framework employing such a nonvolatile solution undergoes a "reconfiguration on power-up" during which the configuration stored in the nonvolatile STTRAM is written into the SRAM back-to-back inverter configuration by the application of suitable bias voltages to the series p and n transistors [see Fig. 3(b)]. HSPICE simulations for our hybrid CMOS–STTRAM model suggest that the scheme achieves very fast reconfiguration ($<$0.1 ns) at the expense of 47.4 μA of current per configuration bit. After the configuration bit is assigned a value, the MTJ leg is cut off by removing the bias voltages, and therefore, does not consume any static "ON." However, this scheme fails to exploit the high integration density of the STTRAM process. Due to the smaller footprint of the MTJ device ($25F^2$) compared to the SRAM ($140F^2$, with F being the minimum feature size), an architecture that reduces the contribution of the CMOS will be able to save the valuable die area.

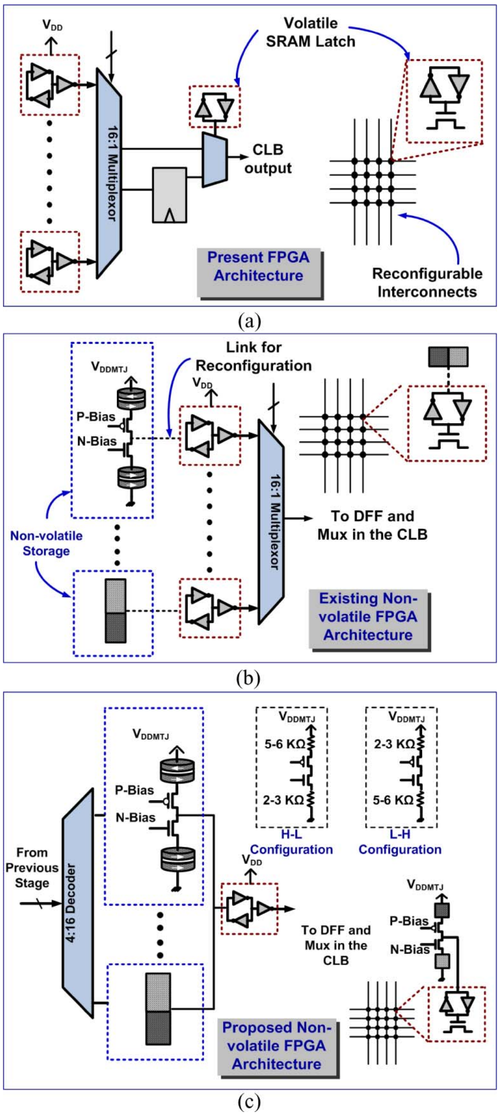


Fig. 3. Proposed scheme for integration of STTRAM in the CLB and programmable interconnects of a CMOS–STTRAM hybrid FPGA architecture.

C. Models for a CMOS–STTRAM Hybrid Framework

The viability of integrating STTRAM with the CMOS process flow has already been demonstrated for a number of technology nodes [5], [9], [14], [24]. Fig. 4(a) shows an example of such integration. We have considered the integration of STTRAM device with 130-nm-technology CMOS process. Fig. 4(b) shows a possible 3-D layout of the STTRAM resistor–divider configuration used in the proposed architecture. Following are the primary motivations behind the choice of 130 nm process node.

1) Simulations have already demonstrated the feasibility of integrating STTRAM in a 130 nm CMOS FPGA framework [16].
2) Coavailability of the TSMC models for 130 nm bulk CMOS [25] and 130 nm Stratix FPGA models [26] used for our simulations.

Although more elaborate dynamic models are presented in [19] and [27], these models account only for the change in resistance during a write to the MTJ cell. Since a simple resistance model suffices to determine the circuit behavior while reading at low bias voltages [15], [22], we have used a static two-resistance model for our analysis. For the MTJ, the resistance states (3 kΩ–1 MΩ [28]), and therefore, the TMR ratio may vary widely from 40% to 220% [29], depending on the thickness of the oxide layer (1–2 nm). However, as demonstrated in [22] and [30], the high- and low-resistance states may be assumed to have values in the range of 5–6 and 2–3 kΩ, respectively. This corresponds to an achievable TMR ratio of 67%–200%. As in [9], in our model, the read-operating voltage ($V_{dd\mathrm{MTJ}}$) for the STTRAM is limited to 0.8 V to prevent any undesired writes during read operation. The operating voltage for the CMOS devices are, however, fixed at 1 V. As demonstrated in [7] and [9] [see Fig. 4(a)], all the MTJs are processed on top of the CMOS process. Dimensions for these MTJs are typically of the order of 100 nm × 50 nm. The lower metal layers are used for routing the CMOS logic, while the upper metal layers are used for connecting to the MTJ. For estimating the performance of the CMOS–STTRAM hybrid framework, it is, therefore, necessary to consider the resistance of the intermediate vias [see Fig. 4(b)]. Representative numbers for a four-layer metal process at 130 nm node were obtained from [31]. The resistance models used hereafter include the MTJ as well as the via resistances. With this model, we now seek to find an effective architecture for a CMOS–STTRAM hybrid FPGA.

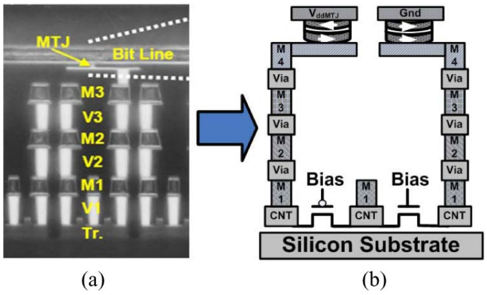


Fig. 4. (a) Cross-sectional SEM images showing integration of STTRAM in CMOS processes [7]. (b) 3-D layout for STTRAM as a resistor–divider configuration.

III. Circuit/Architectural Optimizations for a CMOS–STTRAM Hybrid FPGA

A. Circuit Optimizations for a Static Voltage-Sensing-Based Logic Evaluation Architecture

1) Read Operation: The proposed architecture is illustrated in Fig. 3(c). The multiplexor inside the CLB of a conventional design [see Fig. 3(a)] is replaced with a 4 to 16 decoder producing one-hot output. Each of the 16 outputs from the decoder drive the p-type metal–oxide–semiconductor (PMOS) and n-type metal–oxide–semiconductor (NMOS) transistors present in the 16 resistor–divider structures of the LUT. The output of the leg that is turned ON, therefore, determines the final latch output. As shown in Fig. 3(c), hereafter, we will denote the two resistor–divider configurations as H–L and L–H, depending on whether the higher or the lower MTJ resistance is connected to

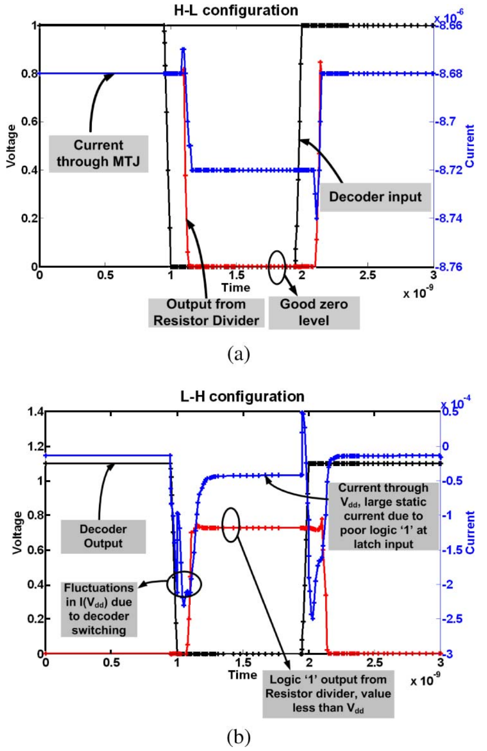


Fig. 5. Voltage and current levels for (a) H–L configuration and (b) L–H configuration during a read operation in the proposed architecture.

$V_{dd\mathrm{MTJ}}$. The H–L configuration leads to an acceptable logic zero at the output of the selected MTJ leg (0.014 V), and hence, at the input of the final latch [see Fig. 5(a)]. The configuration produces only a small static current through the MTJ leg that is required to be ON over the entire clock period. Note that this static current ($\sim$8.7 μA) is much less compared to the minimum MTJ write current ($\sim$100 μA) reported to date [9]. The resistor–divider, however, does not provide a rail to rail switching, and for an L–H configuration, the high-output logic level is at a voltage of 0.7 V. This is primarily due to the moderate TMR ratio (200%) used in our models. A higher TMR ratio (403%) similar to that proposed in [9] leads to an improved logic one level. It is due to this degraded high logic level that a static ON current flows through the CMOS latch over the entire clock period. Note that this static ON current ($\sim$43 μA) is considerably higher than the static current in the H–L configuration or the current that flows through the MTJ leg during this period ($\sim$3.5 μA). The proposed architecture, therefore, provides an "instant ON" nonvolatile FPGA framework with a higher integration density at the cost of increased power overhead. The power overhead for a single interconnect switch for an H–L or L–H configuration is same as that for a single configuration bit of a CLB. However, after application mapping, most of the interconnect switches inside an FPGA are OFF, which allows the use of the low-power H–L state being used for holding the configuration. Table I provides a comparison of the design overheads for the CLBs in the present and the proposed architecture.

For comparison purpose, we have chosen Altera Stratix FPGA at 130 nm [26] as our baseline configuration. The Stratix FPGA consists of four-input LUTs with 16 configuration bits per CLB. The decoder and the multiplexors used in our estimation of design overhead were optimized for area using a Synopsys Design Compiler for a 130-nm library. The power was estimated for a 12.5% input activity for a cycle time of 4 ns. As we note from Fig. 3(a), each LUT in a conventional FPGA framework stores 16 configuration bits using 16 back-to-back inverter configurations. The configuration bits are followed by a multiplexor tree that selects 1 of the 16 configuration bit locations. Area calculation for the proposed framework considers the area for the decoder, CMOS area for the resistor–divider, and a single latch at the final stage of the LUT. Note that our area calculation also includes the decoder logic required to generate the opposite p- and n-bias values for the MTJ legs. From Table I, we note that the proposed architecture improves the area and delay requirements for a single CLB by 48.38% and 22.28%, respectively. The savings in area primarily occurs due to the fact that instead of a latch, only a resistor–divider is present at each configuration bit location. We have considered only the CMOS area in our calculations, since the total number of STTRAM devices required in a CLB design is much less than the number of CMOS devices. Moreover, as illustrated in Fig. 4, for CMOS–STTRAM integration, STTRAM devices are typically on a process layer different from the CMOS layer. Hence, the total CMOS area provides a close estimate of the total CLB area. As we may note from Table I, the dynamic power requirement for the H–L and L–H configurations of the proposed framework are lesser and greater than the conventional design by 28.48% and 33.33%, respectively. The CMOS-based logic does not consume any static power during a normal operation. Here, by static current we refer to a steady dc current flowing from supply to ground. Therefore, the static power reported for a CMOS CLB in Table II corresponds to the leakage current at 130 nm. However, in the proposed framework, a static current flows through the MTJ leg that is turned on in the following latch. Following circuit optimization techniques were employed to improve the logic "1" at the output.

1) Upsize the PMOS transistor at the output.
2) Since the series NMOS cannot be downsized due to the constraints for delivering the write current, the output logic

TABLE I
DESIGN OVERHEAD FOR A CLB

Design Overhead	Stratix	Proposed Arch.	
		H-L	L-H
Area (μm^2)	556.23	287.07	287.07
Delay (ns)	0.193	0.151	0.149
Dynamic Power (μW)	51.93	37.14	69.24
Static (or Leakage) Power (μW)	3.21	8.71	44.68

TABLE II
SAVINGS IN STATIC CURRENT IN A CLB FOR CONVENTIONAL AND THE PROPOSED APPROACH

Design Overhead	CMOS FPGA	CMOS FPGA with High-Vt latches	Hybrid FPGA with supply gating
Static(or Leakage) Power(μW)	3.21	1	0.072
Delay	-	Config. time increases by 10.4%	Evaluation time increases by 18.6%

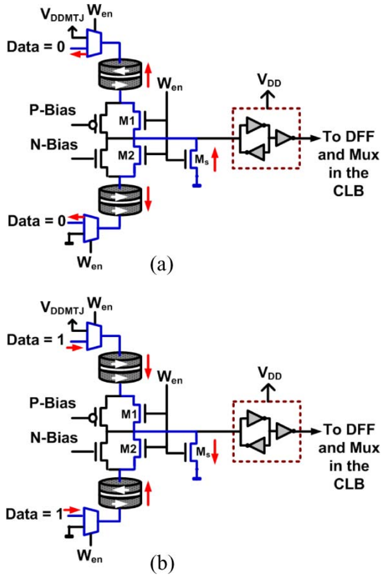


Fig. 6. Circuit scheme for writing to the MTJ cells in the resistor–divider configuration. (a) H–L configuration to L–H configuration. (b) L–H configuration to H–L configuration.

level was further skewed toward “1” using a low-V_t PMOS device and a high-V_t NMOS device.

These optimizations in the resistor–divider circuit improves the logic “1” output level from 0.68 to 0.75 V, thereby minimizing the static current in the final latch stage.

2) Write Operation: As we may note from Fig. 2, a current with value higher than the thresholds I_{HL} or I_{LH} is required for changing the MTJ cells from high- to low- and from low- to high-resistance states, respectively. Using a value of $I_{HL} = I_{LH} = 90\ \mu$A, we have attempted to derive the circuit scheme required for writing the MTJ cells in the resistor–divider configuration. Fig. 6 shows the modified circuit with additional transistors for performing the write operation. Note that transistors M1 and M2 need to be incorporated in every configuration bit location, while M_s is shared for all the configuration bits in a single CLB. Although M1 and M2 are introduced for performing the write operation, they do not incur additional area overhead compared to the configuration bits in conventional CMOS FPGA circuits. The reason is that in CMOS FPGA circuits, these additional transistors are already present and serve as access transistors for writing into the configuration bits. When write enable (W_{en}) is 0, the additional transistors are OFF and a normal read operation is performed. At this point, “$V_{DD\mathrm{MTJ}}$” and “0” are the voltages applied at the two ends of the resistor–divider circuit, as shown in Fig. 6. During the write operation, $W_{\mathrm{en}} = 1$; M1, M2, and M_s are ON; and input data are written into the MTJ cells. Depending on whether “data = 1” or ‘’data = 0,” either an H–L or an L–H state is written into the resistor–divider configuration. In our simulations, with static resistance values of 6000 and 2000 for the antiparallel and parallel MTJ states, the current noted through the top and bottom MTJ cells are 91.35 and 126.32 μA for the “data = 0” case and 98.94 and 100.22 μA for the “data = 1,” respectively. The transistors M1, M2, and M_s were sized to be 2X, 2X, and 4X, respectively, where X denotes the minimum transistor width at 130 nm technology node.

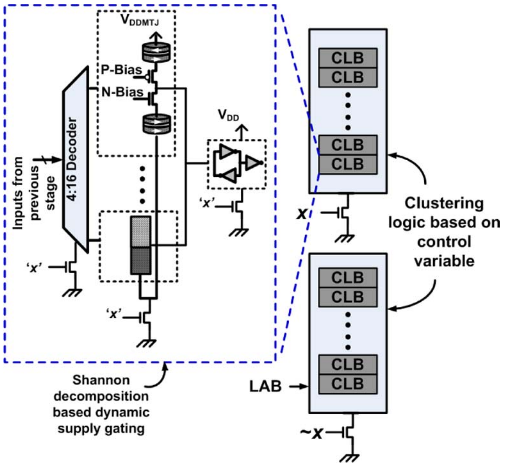

V_DDMTJ
P-Bias
N-Bias
V_DD
Inputs from previous stage
4:16 Decoder
‘x’
‘x’
‘x’
CLB
CLB
CLB
CLB
CLB
CLB
CLB
CLB
X
Clustering logic based on control variable
Shannon decomposition based dynamic supply gating
LAB
~x

Fig. 7. Hardware architecture for Shannon-decomposition-based dynamic supply gating in a CMOS–STTRAM hybrid FPGA.

B. Architecture for Shannon-Decomposition-Based Dynamic Supply Gating

Supply gating has emerged as an extremely effective technique for reduction of dynamic and active leakage power in standard cell designs [33]. The savings in dynamic and leakage power becomes more substantial when the supply gating is dynamically applied to finer granularity of logic. To achieve this objective, a hypergraph partitioning of the original circuit followed by Shannon-decomposition-based synthesis of the partitioned netlists has been proposed in [32]. The primary concept is illustrated in [32]. The Boolean expression *f* corresponding to the original netlist can be factored into two cofactors: *f*(0) and *f*(1). These correspond to the condition when the control variable *x* is 0 or 1. In general, any Boolean expression *g* can be decomposed into two cofactors and a logic *h* independent of the control variable *x*. Thus, if *x* is used as the control variable for the gating transistors, the synthesized design achieves dynamic supply gating. For a standard cell design, such an approach has proven to be extremely effective for dynamic and leakage power reduction. However, the same has not been extensively explored in the realm of designs mapped to FPGA. The primary reason is that if the supply is gated off, the configuration stored in the SRAM present in the LUTs is lost. Due to its nonvolatile nature, the CMOS–STTRAM hybrid FPGA presents an opportunity to extend the same concept for power reduction in FPGA circuits.

1) Hardware Support for Shannon Decomposition: The nonvolatile nature of the proposed hybrid CMOS–STTRAM FPGA framework allows a dynamic supply gating of the decoder, the MTJ legs, and the output latch without destroying the configuration inside the CLB. Fig. 7 shows the hardware support for the proposed supply gating scheme. Separate supply gating transistors have been used for gating the CMOS logic and the MTJ. We have used 2X and 1X minimum-sized transistors for gating the CMOS and the MTJ legs, respectively. Since the MTJ leg consumes a static current over the entire clock period, a fine-grained dynamic supply gating brings significant improvement to the power requirements for the mapped design. Table II compares the static power requirements for a

TABLE III
SAVINGS IN STATIC POWER RESULTING FROM A SHANNON-DECOMPOSITION-BASED DYNAMIC SUPPLY GATING APPROACH

Benchmark Circuits	Static Power with H-Vt latches(μW)	Static Power with supply gating(μW)	%Power savings
c1355	79.50	57.13	28.14
c1908	68.25	31.07	54.48
c2670	126.00	46.20	63.33
c3540	190.50	61.08	67.94
c5315	311.25	79.40	74.49
c6288	394.50	168.98	57.17
c7552	358.50	120.55	66.38
s1423	123.00	50.79	58.71
s5378	340.50	125.17	63.24
s9234	243.75	81.81	66.44
s13207	585.75	134.84	76.98
s15850	697.50	181.18	74.03
s35932	2160.00	844.24	60.92

single CLB in a normal design, a design with high-V_t latches and the proposed design with supply gating. As before, note that the CMOS-based FPGA framework does not consume any static current. Hence, the reported numbers stand for the leakage current at 130 nm node. For normal standard-cell-based design, placement of muxes and routing of control signal for fine-grained supply gating leads to increased complexity in the design process. However, the close organization of the CLBs in logic array blocks (LABs) for modern FPGA lends itself to the effective implementation of dynamic supply gating. Logic corresponding to the cofactors can be clustered effectively into each LAB, which may then be gated using the control variable and its complement. The scheme is illustrated in Fig. 7.

2) Improvement in Power Consumption With Dynamic Supply Gating: The steps for mapping a design to a framework with support for dynamic supply gating were adopted from [32]. The original circuit was first partitioned using a hypergraph partitioning approach described in [32]. Each partition was then decomposed into four cofactors using two-level Shannon decomposition. By two-level Shannon decomposition we mean that for two control variables x and y, the cofactors correspond to the four combinations $x' \cdot y'$, $x' \cdot y$, $x \cdot y'$, and $x \cdot y$. For a given input combination to a partition, only one of the cofactors will be active, while the other three cofactors will be supply-gated. Such a partitioning of the original circuit followed by Shannon decomposition resulted in a substantial (68.4%) power improvement for the benchmarks (see Table III). Although the Shannon decomposition resulted in significant savings of static power for all the benchmarks, it led to an increase in the total number of resources required for mapping all the cofactors. This increase is due to: 1) duplication of logic between the two cofactors and 2) insertion of the supply gating transistor. An estimate of this increase for standard benchmark circuits is provided in Section V.

IV. APPLICATION MAPPING USING PREFERENTIAL STORAGE

The observation that the power requirement for the CLB operation in the original design is intermediate to that required for the H–L and the L–H configuration leads to the concept of preferential storage. The idea is to map the logic inside each LUT in such a manner that the truth table stored has more logic zeros than logic ones, or *vice versa*. The concept of preferential storage has been explored for diverse reasons in the context of FPGAs. Srinivasan *et al.* [34] attempted to improve the soft-error tolerance of FPGA configuration bits by storing more logic zeros in the LUTs. Anderson and Najm [35] have proposed to reduce the leakage power consumption in FPGA by correct selection of polarity for the configuration bits. We have developed a simple software routine that optimizes the circuit after it has been technology-mapped using standard algorithms such as Flowmap or Cutmap [36].

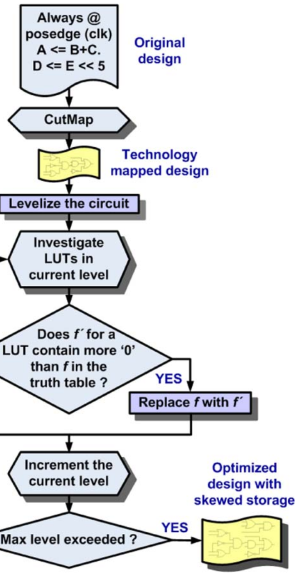

Always @
posedge (clk)
A <= B+C.
D <= E << 5
Original design
CutMap
Technology mapped design
Levelize the circuit
Investigate LUTs in current level
Does f′ for a LUT contain more '0' than f in the truth table ?
NO
YES
Replace f with f′
Increment the current level
Optimized design with skewed storage
NO
Max level exceeded ?
YES

Fig. 8. Flowchart depicting the basic steps for a preferential-storage-based application mapping.

Fig. 8 shows the basic steps of our routine. The input to the routine is a "blif" file containing the optimized programmable logic array (PLA) representations for each LUT obtained from the Flowmap algorithm. The routine then evaluates the "zero" and "one" content of the truth table given by its PLA representation. It retains or replaces the original function by its complement depending on the direction in which the number of "zeros" increase in the truth table. The prior step is repeated for every level of the mapped circuit until the last level is reached. The routine was run on several ISCAS'85 and 89 benchmarks [37]. Table IV summarizes the savings in average dynamic and static power compared to the unoptimized design. As seen in Table IV, the proposed routine achieves, on average, a 9.78% improvement in average dynamic power per CLB and a 21.95% improvement in average static power per CLB for a 103.23% increase in the number of CLB outputs with "0" polarity.

V. SIMULATION RESULTS

The optimized CMOS–STTRAM hybrid FPGA framework has been validated by comparing it with an Altera-Stratix at 130 nm FPGA model. The comparison has been performed by mapping a set of ISCAS'85 and 89 benchmark circuits to both the

TABLE IV
SAVINGS IN DYNAMIC AND STATIC POWER ACHIEVED THOUGH PREFERENTIAL STORAGE IN CMOS–STTRAM HYBRID LUTS

ISCAS Benchmarks	Ratio of 0's and 1's as CLB output		% Change in CLB output polarity	Avg. Dynamic power per CLB (μ W)		% Change in avg. dynamic power per CLB	Avg. Static power per CLB (μ W)		% Change in avg. static power per CLB
	Pre-opt.	Post-opt.		Pre-opt.	Post-opt.		Pre-opt.	Post-opt.	
c1355	1.03	1.39	34.52	52.94	50.58	4.45	26.41	23.77	10.00
c1908	0.97	2.16	122.95	53.43	47.28	11.50	26.96	20.08	25.53
c2670	1.28	2.39	86.86	51.22	46.61	9.01	24.49	19.32	21.12
c3540	0.93	2.37	153.66	53.75	46.67	13.16	27.32	19.39	29.01
c5315	0.77	2.08	169.03	55.23	47.55	13.91	28.98	20.38	29.69
c6288	0.92	1.42	54.61	53.88	50.41	6.44	27.47	23.58	14.15
c7552	0.86	1.52	75.75	54.38	49.90	8.23	28.03	23.01	17.90
s1423	1.39	2.46	76.55	50.56	46.42	8.18	23.75	19.11	19.51
s5378	1.06	2.36	123.34	52.74	46.68	11.48	26.19	19.40	25.90
s9234	0.97	2.43	151.80	53.47	46.50	13.05	27.01	19.19	28.94
s13207	1.09	2.78	155.31	52.52	45.64	13.10	25.95	18.24	29.71
s15850	1.16	2.34	101.12	51.97	46.75	10.05	25.33	19.47	23.12
s35932	1.20	1.64	36.54	51.70	49.28	4.69	25.03	22.31	10.85

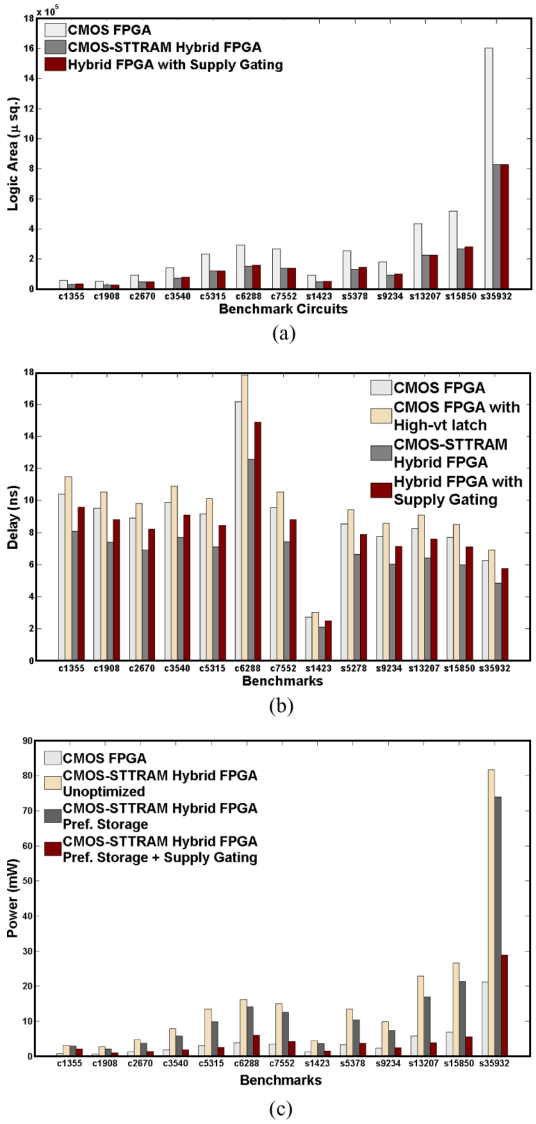
TABLE V
MAPPING STATISTICS OF ISCAS'85 AND 89 BENCHMARKS TO THE ALTERA–STRATIX FPGA PLATFORM

Index	Benchmark	# of CLB	Total delay (ns)	Logic delay (ns)
1	c1355	106	16.34	10.41
2	c1908	91	19.74	9.53
3	c2670	168	18.25	8.91
4	c3540	254	24.15	9.88
5	c5315	415	20.75	9.16
6	c6288	526	45.25	16.15
7	c7552	478	25.29	9.56
8	s1423	164	9.18	2.73
9	s5378	454	12.98	8.55
10	s9234	325	10.84	7.75
11	s13207	781	15.82	8.24
12	s15850	930	16.20	7.70
13	s35932	2880	10.29	6.25

frameworks. The number of CLBs required to map a given application has been obtained by mapping these benchmark circuits on a Stratix FPGA using Altera-Quartus v7.0 software. Results for the logic delay, power, and resource utilization for the CMOS FPGA framework were obtained by mapping the benchmark circuits to the Altera-Stratix at 130 nm FPGA platform. Table V presents the results for the CMOS FPGA platform. From the mapping statistics presented in Table V and the area, delay, and power estimates for a single CLB following the optimizations in Sections III and IV, we have calculated the design overhead for the proposed CMOS–STTRAM hybrid FPGA framework. Fig. 9(a)–(c) shows the design overhead estimates for the benchmark circuits when mapped to both the CMOS FPGA and the proposed CMOS–STTRAM hybrid FPGA frameworks.

A. Improvement in Area

The proposed architecture minimizes the number of CMOS back-to-back inverters per CLB. This leads to a substantial improvement in the area dedicated to the CMOS logic inside the CLB (see Table I). Fig. 9(a) shows the total logic area required for mapping the benchmarks in the following two cases: 1) CMOS FPGA and 2) CMOS–STTRAM hybrid FPGA. The estimates for the total logic area in the CMOS FPGA and the CMOS–STTRAM hybrid FPGA was obtained by multiplying



(a)

(b)

(c)

Fig. 9. Comparison of design overheads between the CMOS FPGA and the CMOS–STTRAM hybrid FPGA using the proposed optimization approaches. (a) Area. (b) Delay. (c) Power.

the total number of CLBs required to map a given design with the corresponding area estimates presented in Table I. From these results, we note that the proposed scheme leads to an average 48.39% savings in logic area compared to the conventional platform. The results are presented in Fig. 9(a), from where we note that the Shannon-decomposition-based supply gating incurs an average overhead of 4% for the benchmark circuits. Thus, the actual improvement in logic area over a conventional CMOS FPGA design is ∼44%. Since programmable interconnects occupy a significant portion (∼75%) [38], [39] of the FPGA die-area, the overall area improvement in the proposed nonvolatile FPGA framework is expected to be ∼11%.

B. Improvement in Delay

Fig. 9(b) shows the logic delay of the chosen benchmark set for four different scenarios:

1) normal CMOS FPGA;
2) low-power CMOS FPGA employing high-V_t transistors for the configuration bits;
3) proposed CMOS–STTRAM hybrid FPGA;
4) proposed FPGA design with supply gating transistors.

Delay for the CMOS–STTRAM hybrid framework was calculated as

$$\text{Path Delay} = \text{Interconnect Delay} + (\text{\# of Logic Stages}) \times (\text{Delay through each stage}).$$

In this expression, each stage refers to a single CLB. In Table I, we have already estimated the delay through a CMOS–STTRAM hybrid CLB structure. From this estimate and the mapping statistics presented in Table V, we have calculated the delay for the benchmark circuits mapped to the proposed FPGA platform. As seen from Fig. 9(b), compared to the conventional CMOS FPGA design, the proposed architecture achieves an average logic delay improvement of 22.28%. For a framework employing low-power techniques, the average improvement over a complete CMOS-based design is 19.77%. Since the programmable interconnects contribute to more than 50% of the path delay (refer to the logic delay contribution in Table V), the average improvement in the overall path delay is 11.08%.

C. Improvement in Power

The resistor–divider circuit optimization, preferential storage, and the dynamic supply gating scheme developed in earlier sections significantly reduce the dynamic and static current contributions from the CLB in the proposed nonvolatile FPGA framework. Fig. 9(c) compares the power requirement of the optimized and the unoptimized designs with the conventional CMOS-based FPGA framework. From Fig. 9(c), we note that both preferential storage and Shannon-decomposition-based dynamic supply gating helps in achieving a significant reduction in the total power requirement for the CMOS–STTRAM hybrid FPGA framework. On average, the optimized STTRAM hybrid framework consumes 16.1% less power than the CMOS design.

VI. Conclusion

In this paper, we have presented a novel nonvolatile CMOS–STTRAM hybrid FPGA platform, identified key design challenges, and proposed optimization techniques at multiple levels of design abstraction. The proposed architecture leverages on the high integration density of emerging STTRAM devices and minimizes the total logic area by reducing the contribution of CMOS cross-coupled inverters serving as configuration bits. An efficient resistor–divider design, an application-mapping methodology based on preferential storage, and Shannon-decomposition-based power gating significantly reduced the power dissipation in the proposed framework. Simulation results for a set of combinational and sequential benchmark circuits show that the proposed nonvolatile FPGA provides a promising reconfigurable computing platform in future technology generations. Future research involves development of novel CLB and programmable interconnect architecture, which will exploit the high integration density of 2-D STTRAM arrays.

References

[1] S. Hauck, "The roles of FPGA's in reprogrammable systems," *Proc. IEEE*, vol. 86, no. 4, pp. 615–638, Apr. 1998.

[2] K. Morris, "Power, suddenly we care," *FPGA Programmable Logic J.*, Apr. 2005.

[3] International Technology Roadmap for Semiconductors. (2007). Emerging Research Devices [Online]. Available: http://www.itrs.net/Links/2007ITRS/2007_Chapters/2007_ERD.pdf.

[4] K. Kim and Y. J. Song, "Current and future high-density FRAM technology," *Integr. Ferroelectr.*, vol. 61, pp. 3–15, 2004.

[5] T. W. Andre, J. J. Nahas, C. K. Subramanian, B. J. Garni, H. S. Lin, A. Omair, and W. L. Martino, "A 4-Mb 0.18-μm 1T1MTJ toggle MRAM with balanced three input sensing scheme and locally mirrored unidirectional write drivers," *IEEE J. Solid-State Circuits*, vol. 40, no. 1, pp. 301–309, Jan. 2005.

[6] W. Y. Cho, B. H. Cho, B. G. Choi, H.R. Oh, S. Kang, K. Kim, K. H. Kim, D. Kim, C. Kwak, H. Byun, Y. Hwang, S. Ahn, G. Koh, G. Jeong, H. Jeong, and K. Kim, "A 0.18-μm 3.0-V 64-Mb non-volatile phase-transition random access memory (PRAM)," *IEEE J. Solid-State Circuits*, vol. 40, no. 1, pp. 293–300, Jan. 2005.

[7] M. Hosomi, H. Yamagishi, T. Yamamoto, K. Bessho, Y. Higo, K. Yamane, H. Yamada, M. Shoji, H. Hachino, C. Fukumoto, H. Nagao, and H. Kano, "A novel non-volatile memory with spin torque transfer magnetization switching: Spin-RAM," in *Proc. IEEE Int. Electron Devices Meeting*, 2006, pp. 459–462.

[8] J. Hayakawa, S. Ikeda, Y. M. Lee, R. Sasaki, T. Meguro, F. Matsukura, H. Takahashi, and H. Ohno, "Current-driven magnetization switching in CoFeB/MgO/CoFeB magnetic tunnel junctions," *Jpn. J. Appl. Phys.*, vol. 44, no. 41, pp. L1267–L1270, 2005.

[9] T. Kawahara, R. Takemura, K. Miura, J. Hayakawa, S. Ikeda, Y. M. Lee, R. Sasaki, Y. Goto, K. Ito, T. Meguro, F. Matsukura, H. Takahashi, H. Matsuoka, and H. Ohno, "2 Mb spin-transfer torque RAM (SPRAM) with bit-by-bit bidirectional current write and paralilelizing-direction current read," in *Proc. IEEE Int. Solid-State Circuits Conf.*, 2007, pp. 480–617.

[10] P. Chow, S. O. Seo, J. Rose, K. Chung, G. Paez-Monzon, and I. Rahardja, "The design of an SRAM based field-programmable gate array. Part II: Circuit design and layout," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 7, no. 2, pp. 191–197, Jun. 1999.

[11] D. Choi, K. Choi, and J. D. Villasenor, "New non-volatile memory structures for FPGA architectures," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 16, no. 7, pp. 874–881, Jul. 2008.

[12] Actel ProASICPLUS, Actel's 2nd Generation Reprogrammable flash FPGAs [Online]. Available at: http://www.actel.com/products/proasicplus/default.aspx

[13] Xilinx Spartan-3AN FPGAs [Online]. Available: http://www.xilinx.com/prs_rls/2007/silicon_spart/0734_spartan3an.htm

[14] S. Tehrani, J. M. Slaughter, M. Deherrera, B. N. Engel, N. D. Rizzo, J. Salter, M. Durlam, R. W. Dave, J. Janesky, B. Butcher, K. Smith, and

G. Grynkewich, “Magnetoresistive random access memory using magnetic tunnel junctions,” *Proc. IEEE*, vol. 91, no. 5, pp. 703–717, May 2003.

[15] W. Zhao, E. Belhaire, Q. Mistral, E. Nicolle, T. Devolder, and C. Chappert, “Integration of spin-RAM technology in FPGA circuits,” in *Proc. Int. Conf. Solid-State Integr. Circuit Technol.*, 2006, pp. 799–802.

[16] N. Bruchon, L. Torres, G. Sassatelli, and G. Cambon, “New non-volatile FPGA concept using magnetic tunneling junction,” in *Proc. IEEE Comput. Soc. Annu. Symp. Emerg. VLSI Technol. Arch.*, 2006, pp. 269–276.

[17] [Online]. Available: http://www.eetimes.com/news/semi/showArticle.jhtml?articleID=10800905

[18] J. Li, C. Augustine, S. Salahuddin, and K. Roy, “Modeling of failure probability and statistical design of spin–torque transfer magnetic random access memory (STT MRAM) array for yield enhancement,” in *Proc. Des. Autom. Conf.*, 2008, pp. 278–283.

[19] Y. Chen, X. Wang, H. Li, H. Liu, and D. V. Dimitrov, “Design margin exploration of spin–torque transfer RAM (SPRAM),” in *Proc. Int. Symp. Qual. Electron. Des.*, 2008, pp. 684–690.

[20] X. Dong, X. Wu, G. Sun, Y. Xie, H. Li, and Y. Chen, “Circuit and microarchitecture evaluation of 3D stacking magnetic RAM (MRAM) as a universal memory replacement,” in *Proc. Des. Autom. Conf.*, 2008, pp. 554–559.

[21] S. P. Ferrera and N. P. Carter, “A magnetoelectronic macrocell employing reconfigurable threshold logic,” in *Proc. Int. Symp. Field Programmable Gate Arrays*, 2004, pp. 143–151.

[22] W. Xu, T. Zhang, and Y. Chen, “Spin-transfer torque magnetoresistive content addressable memory (CAM) cell structure design with enhanced search noise margin,” *Proc. IEEE Int. Symp. Circuits Syst.,* 2008, pp. 1898–1901.

[23] Lattice XP FPGA [Online]. Available: http://www.latticesemi.com/products/fpga/xp/nonvolatilereconfigurabil.cfm?jsessionid=ba30dc495dcc$3F$3FID

[24] U. K. Klostermann, M. Angerbauer, U. Griming, F. Kreupl, M. Ruhrig, F. Dahmani, M. Kund, and G. Miiller, “A perpendicular spin torque switching based MRAM for the 28 nm technology node,” in *Proc. IEEE Int. Electron Devices Meeting*, 2007, pp. 187–190.

[25] TSMC 130 nm models [Online]. Available: www.mosis.org

[26] Altera Stratix FPGA devices: Documentation [Online]. Available: http://www.altera.com/products/devices/stratix/stx-index.jsp

[27] W. Zhao, E. Belhaire, and Q. Mistral, “Macro-model of spin-transfer torque based magnetic tunnel junction device for hybrid magnetic-CMOS design,” in *Proc. IEEE Int. Behav. Model. Simul. Workshop*, 2006, pp. 40–43.

[28] V. Javerliac, “SPINTEC,” private communication, 2005.

[29] S. S. P. Parkin, C. Kaiser A. Panchula, P. M. Rice, B. Hughes, M. Samant, and S. Yang. Giant tunneling magnetoresistance at room temperature with MgO (1 0 0) tunnel barriers [Online]. Available: www.nature.com/naturematerials

[30] Z. Diao, Z. Li, S. Wang, Y. Ding, A. Panchula, E. Chen, L. Wang, and Y. Huai, “Spin-transfer torque switching in magnetic tunnel junction and spin-transfer torque random access memory,” *J. Phys.: Cond. Matter*, vol. 19, pp. 165209.1–165209.13, Apr. 2007.

[31] H. Kudo, K. Yoshie, S. Yamaguchi *et al.*, “Copper dual damascene interconnects with very low-k dielectrics targeting for 130 nm node,” in *Proc. Int. Conf. Interconnect Technol.*, 2000, pp. 270–272.

[32] L. Leinweber and S. Bhunia, “Fine-grained supply gating through hypergraph partitioning and Shannon decomposition for active power reduction,” in *Proc. Des., Autom. Test Eur.*, 2008, pp. 1–6.

[33] J. W. Tschanz, S. G. Narendra, Y. Ye, B. A. Bloechel, S. Borkar, and V. De, “Dynamic sleep transistor and body bias for active leakage power control of microprocessors,” *IEEE J. Solid-State Circuits*, vol. 38, no. 11, pp. 1838–1845, Nov. 2003.

[34] S. Srinivasan, A. Gayasen, N. Vijaykrishnan, M. Kandemir, Y. Xie, and M. J. Irwin, “Improving soft-error tolerance of FPGA configuration bits,” in *Proc. Int. Conf. Comput.-Aided Des.*, 2004, pp. 107–110.

[35] J. H. Anderson and F. N. Najm, “Active leakage power optimization for FPGAs,” *IEEE Trans. Comput.-Aided Des.*, vol. 25, no. 3, pp. 423–437, Mar. 2006.

[36] J. Cong and Y. Ding, “FlowMap: An optimal technology mapping algorithm for delay optimization in lookup-table based FPGA designs,” *IEEE Trans. Comput.-Aided Des.*, vol. 13, no. 1, pp. 1–12, Jan. 1994.

[37] ISCAS 85 and 89 benchmarks [Online]. Available: http://www.fm.vslib.cz/∼kes/asic/iscas/

[38] S. M. Trimberger, *Field-Programmable Gate Array Technology*. Norwell, MA: Kluwer, 1994.

[39] A. Rahman, S. Das, A. P. Chandrakasan, and R. Reif, “Wiring requirement and three-dimensional integration technology for field programmable gate arrays,” *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 11, no. 1, pp. 44–54, Feb. 2003.

Somnath Paul (S’07) received the B.E. degree in electronics and telecommunication engineering from Jadavpur University, Kolkata, India, in 2005. He is currently working toward the Ph.D. degree in computer engineering at Case Western Reserve University, Cleveland, OH.

He was a Design Engineer with Advanced Micro Devices, Bangaluru, India. He has also held internship positions at Intel and Qualcomm. His current research interests include development of novel hardware frameworks for reconfigurable architectures and hardware/software codesign for yield improvement in nanoscale technologies.

Saibal Mukhopadhyay (S’99–M’07) received the B.E. degree in electronics and telecommunication engineering from Jadavpur University, Kolkata, India, in 2000, and the Ph.D. degree in electrical and computer engineering from Purdue University, West Lafayette, IN, in 2006.

He was a Research Staff Member with the IBM T. J. Watson Research Center, Yorktown Heights, NY, where he was engaged in high-performance circuit design and technology-circuit codesign, focusing primarily on static RAMs. He is currently an Assistant Professor in the Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta. His research interests include analysis and design of low-power and robust circuits in nanometer technologies. He has authored or coauthored more than 90 papers published in refereed journals and conferences.

Dr. Mukhopadhyay received the IBM Faculty Partnership Award in 2009, the Semiconductor Research Corporation (SRC) Inventor Recognition Award in 2009, the SRC Technical Excellence Award in 2005, the IBM Ph.D. Fellowship Award for 2004–2005, the Best in Session Award at the 2005 SRC Technology and Talent for the 21st Century Conference, and the Best Paper Awards at the 2003 IEEE Nano and the 2004 International Conference on Computer Design.

Swarup Bhunia (S’00–M’05–SM’09) received the B.E. (Hons.) degree from Jadavpur University, Kolkata, India, the M.Tech. degree from the Indian Institute of Technology (IIT), Kharagpur, India, and the Ph.D. degree from Purdue University, West Lafayette, IN, in 2005.

He has worked in the semiconductor industry on Register Transfer Level (RTL) synthesis, verification, and low-power design for about three years. He is currently an Assistant Professor of electrical engineering and computer science at Case Western Reserve University, Cleveland, OH. He has authored or coauthored more than 80 papers published in peer-reviewed journals and premier conferences in the area of very large scale integration (VLSI) design and test techniques. His research interests include low-power and robust design, hardware security and protection, adaptive nanocomputing, and novel test methodologies.

Dr. Bhunia received the Semiconductor Research Corporation (SRC) Technical Excellence Award in 2005 as a team member, the Best Paper Award at the International Conference on Computer Design in 2004, the Best Paper Award at the Latin American Test Workshop in 2003, and the Best Paper Nomination at the Asia and South Pacific Design Automation Conference in 2006. He has been a member of the technical program committees of the Design Automation and Test in Europe during 2006–2009, the Hardware Oriented Trust and Security Workshop in 2008, the IEEE/International Federation of Information Processing International Conference on VLSI in 2008, the Test Technology Educational Program during 2006–2008, the International Symposium on Low Power Electronics and Design during 2007–2008, and the IEEE/Association for Computing Machinery Symposium on Nanoscale Architectures during 2007–2008. He was also a member of the program committee of the International Online Test Symposium in 2005.