Hindawi
VLSI Design
Volume 2017, Article ID 1030249, 9 pages
https://doi.org/10.1155/2017/1030249

Research Article

State-Transition-Aware Spilling Heuristic for MLC STT-RAM-Based Registers

Yuanhui Ni,[1] Zhiyao Gong,[1] Weiwen Chen,[1] Chengmo Yang,[2] and Keni Qiu[1,3]

[1] *College of Information Engineering, Capital Normal University, Beijing 100048, China*
[2] *Department of Electrical and Computer Engineering, University of Delaware, Newark, DE, USA*
[3] *Beijing Advanced Innovation Center for Imaging Technology, Beijing, China*

Correspondence should be addressed to Keni Qiu; qiukn@cnu.edu.cn

Received 6 June 2017; Accepted 22 October 2017; Published 22 November 2017

Academic Editor: Chien-In Henry Chen

Multilevel Cell Spin-Transfer Torque Random Access Memory (MLC STT-RAM) is a promising nonvolatile memory technology to build registers for its natural immunity to electromagnetic radiation in rad-hard space environment. Unlike traditional SRAM-based registers, MLC STT-RAM exhibits unbalanced write state transitions due to the fact that the magnetization directions of hard and soft domains cannot be flipped independently. This feature leads to nonuniform costs of write states in terms of latency and energy. However, current SRAM-targeting register allocations do not have a clear understanding of the impact of the different write state-transition costs. As a result, those approaches heuristically select variables to be spilled without considering the spilling priority imposed by MLC STT-RAM. Aiming to address this limitation, this paper proposes a state-transition-aware spilling cost minimization (SSCM) policy, to save power when MLC STT-RAM is employed in register design. Specifically, the spilling cost model is first constructed according to the linear combination of different state-transition frequencies. Directed by the proposed cost model, the compiler picks up spilling candidates to achieve lower power and higher performance. Experimental results show that the proposed SSCM technique can save energy by 19.4% and improve the lifetime by 23.2% of MLC STT-RAM-based register design.

1. Introduction

Electromagnetic radiation effects can cause several types of errors on traditional SRAM-based registers and DRAM-based memory such as single event upset (SEU) and single event functional interrupt (SEFI). Especially in aerospace where radiation is quite intense, the stability and correctness of systems are strongly affected. It is therefore essential to make electronic components and systems resistant to damage or malfunctions caused by ionizing radiation. Previous studies have shown that nonvolatile memories such as Spin-Torque Random Access Memory (STT-RAM), Phase Change Memory (PCM), Domain Wall Memory (DWM), and Flash memories [1–3] exhibit the appealing feature of soft-error immunity. Different from charge-based memories such as SRAM, NVMs such as STT-RAM, PCM, DWM, and Flash memories store data as a change in physical state. Since write operations involve changing the physical state, NVMs are resilient to radiations in the harsh space environment. Among these technologies, STT-RAM has the shortest access policy and can potentially be used to build registers. Multilevel cell STT-RAM (MLC STT-RAM) offers high storage density, and recent studies have shown that the write latency of STT-RAM can be greatly reduced by modifying the bit-cell structure or increasing write current [4]. In this paper, we consider to build a full electromagnetic-immunity memory hierarchy consisting of MLC STT-RAM-based registers and nonvolatile main memory. The goal of this work is to effectively allocate MLC STT-RAM-based registers.

During compilation, the decision of which variables to be kept in registers at each point in the generated code is called *register allocation*. Typically, register allocation is modeled as a graph coloring program which is aimed at finding a *k-optimal-coloring* solution for the interference graph. In Chaitin's coloring [5], when the physical registers are insufficient to hold all the variables, that is, when a

node in graph cannot be provably colored, several live ranges must be selected to spill. Since the cost to write different values in SRAM is uniform, traditional register allocators [5, 6] heuristically select potential spilling candidates without considering state-transition costs. When applied to STT-RAM-based registers, however, those techniques produce inferior spilling decisions.

For MLC STT-RAM-based registers, the programming costs of variables with different state transitions vary significantly [7]. To minimize the overall programming energy, we propose a write state-transition-aware spilling cost minimization (SSCM) technique. First, a spilling cost model needs to be built. Then, the spilling priority order is derived based on the cost model to make a better allocation decision. In particular, this paper tends to select the potential spilling nodes with larger spilling costs. The main contributions of our paper are summarized as follows.

(i) To the best of our knowledge, this is the first work which integrates the write state-transition cost of MLC STT-RAM into the spilling policy of register allocation.

(ii) A cost model is proposed to quantify the spilling cost of variables in the potential spilling list.

(iii) A SSCM algorithm is proposed to select the best spilling candidate with the goal of reducing the overall programming energy of MLC STT-RAM.

(iv) Experiments are conducted to quantitatively evaluate the effectiveness of the proposed approach.

The rest of this paper is organized as follows. The background of STT-RAM and register allocation are introduced in Section 2. Section 2.3 presents the motivation of this work. Section 3 derives the spilling cost model and presents the algorithm of SSCM-aware register allocation. A set of experiments is conducted to evaluate the proposed methods in Section 4. Finally, Section 5 concludes the paper.

2. Background Information

This section firstly describes the resistance state transition of MLC STT-RAM and its nature of antielectromagnetic radiation and then presents the traditional graph coloring algorithm for register allocation. Finally, previous spilling heuristic is discussed.

2.1. MLC STT-RAM Preliminaries. Among all the emerging NVMs, the spin-transfer torque RAM (STT-RAM) is considered as a promising candidate for on-chip memory because of its advantages, such as low leakage, high density, fast read speed, nonvolatility, and immunity to radiation-induced soft errors [8]. It features much better endurance and performance than other magnetic memory technologies. Compared to SRAM, it is up to 4 times denser and has much lower leakage energy. This enables the implementation of very large on-chip memories with near-zero static consumption, which alleviates both main memory stress and power consumption. High TMR (tunneling magnetoresistance ratio) motivated the research on multilevel cell (MLC) STT-RAM. In a MLC STT-RAM bit, n bits are represented by 2^n states, that is, resistance. By doing so, MLC technology can effectively improve the memory density and power efficiency.

Table 1: Parameters of SRAM, SLC STT-RAM, and MLC STT-RAM.

Parameters	SRAM	SLC STT-RAM	MLC STT-RAM
Read latency (cycles)	7.43	9.08	S:6.73, H:9.80
Read dyn. eng. (nJ)	0.161	0.216	S:0.22, H:0.43
Write latency (cycles)	5.78	25.58	S:25.31, H:56.50
Write dyn. eng. (nJ)	0.156	0.839	S:0.843, H:2.502
Leakage power (mW)	295.58	18.39	7.02
Array area (mm^2)	7.28	1.86	1.01

In a (SLC) STT-MRAM device, the spin of the electrons is changed using a spin-polarized current. This effect is achieved in a magnetic tunnel junction (MTJ). An MTJ device consists of a reference layer and a free layer. The magnetization direction (MD) of reference layer is unchanged while the MD of free layer can be flipped by applying a current through the MTJ. The MLC STT-RAM comprises 2-bit MLC cell which is adopted in this work. Two MTJs with different sizes are stacked vertically atop an NMOS transistor. The four resistance states are defined by the four combinations of different MDs of the two MTJs [9].

For comparison, Table 1 shows the parameters of SRAM, SLC (Single-Level Cell) STT-RAM, and MLC (multilevel cell) STT-RAM [10]. It is known that registers are frequently written component in a system. When architecting STT-RAM for registers, the long write latency will impose great impact on both performance and energy of architectural components.

In conventional random access memory (RAM) technologies, data are stored as electric charge or current flows. For STT-RAM, data are stored by magnetic storage elements-magnetic tunnel junctions (MTJs). Since STT-RAM cell does not carry electric charge, it is resilient to radiations. *Such natural immunity to electromagnetic makes it an ideal candidate to replace the traditional SRAM technology and be used as registers in the harsh space environment* [11]. Samples were exposed to 2 MeV and 220 MeV protons and showed no changes in bit-state or write performances. Radiation testing results show that STT-RAM will not suffer SEUs when used in space [12]. Thanks to its easy integration with CMOS and infinite endurance, STT-RAM has been proposed to be widely used in order to overcome the power challenge of conventional CMOS circuits [13]. Therefore, in many harsh environments like aerospace, STT-RAM is an ideal candidate to build registers. In fact, STT-RAM-based register file has been used in [14–16] to achieve lower dynamic and leakage energy consumption. Recently, IBM researchers in collaboration with Samsung researchers demonstrated 11 nm STT-RAM junction, which is a significant achievement on the way to substitute DRAM with STT-RAM [17]. This work proposes to build STT-RAM-based registers for embedded systems in rad-hard environment.

Table 2: Write transitions.

Zero transition (ZT)	$R00 \rightarrow R00$	Require no current
	$R01 \rightarrow R01$	
	$R10 \rightarrow R10$	
	$R11 \rightarrow R11$	
Soft transition (ST)	$R01 \leftrightarrow R00$	Require small current
	$R10 \leftrightarrow R11$	
Hard transition (HT)	$R00 \leftrightarrow R11$	Require large current
	$R01 \rightarrow R11$	
	$R10 \rightarrow R00$	
Two-step transition (TT)	$R00 \rightarrow R10$	Require sum of two-step current
	$R01 \rightarrow R10$	
	$R10 \rightarrow R01$	
	$R11 \rightarrow R01$	

The resistance of an MTJ can be changed by injecting a switching current. In particular, MLC STT-RAM has two domains, a hard domain and a soft domain. The magnetic direction of the soft domain can be changed by a small current, while applying a larger current to MTJ affects both hard and soft domains. In this paper, the first bit of a 2-bit data indicates the magnetization direction of the hard domain and the second bit indicates the magnetization direction of the soft domain. States transitions of MTJ resistance can be presented in Table 2 with the following four types [18], where "$R00$" represents that the soft-bit and hard-bit are both low resistance. Similarly, "$R01$" stands for the soft-bit with low resistance while hard-bit is high resistance. And "$R10$" represents the soft-bit with high resistance while hard-bit is soft resistance. "$R11$" represents the soft-bit and hard-bit being both high resistance.

(i) Zero transition (ZT): neither bit is changed.

(ii) Soft transition (ST): only the magnetic orientation of the soft domain is switched.

(iii) Hard transition (HT): the magnetic orientation of the hard domain is switched, and two domains have the same orientation.

(iv) Two-step transition (TT): transition completes with two steps, including one HT followed by one ST.

Table 3 presents the rated current required to switch the state of MLC STT-RAM for each transition [18]. When the current is larger than the rated current, the state can switch to the other. A negative value sets the current in the reverse direction, and "—" represents that a state cannot be directly converted into the other state. It can be seen that switching a hard domain requires a larger current than switching a soft domain. For a two-step transition, the required current is the sum of the absolute currents of both steps.

It can be seen from Table 3 that changing states has significant impact on the energy consumption of MLC STT-RAM. It is therefore preferable to spill variables with higher programming energy to save register access energy during program execution. To achieve this goal, a spilling policy taking state-transition costs into account is proposed in this paper for MLC STT-RAM-based registers.

Table 3: Switching currents of MLC STT-RAM cell (μA).

	To			
From	$R00$	$R01$	$R10$	$R11$
$R00$	0	−38.3	—	−56.7
$R01$	26.3	0	—	−56.7
$R10$	66.4	—	0	−9.1
$R11$	66.4	—	39.3	0

2.2. Graph Coloring Based Register Allocation. A graph coloring based register allocation approach was designed by Chaitin et al. [5]. Its basic data structure is the interference graph $G = (V, E)$ [19]. The node in G represents live ranges, and the edge between nodes corresponds to interferences. Adjacent nodes are not allowed to simultaneously live and share the same physical register. The *k-coloring* problem assigns one of k colors (physical registers) to the node of G. Various phases of the process are described as follows.

Build. Construct the interference graph $G = (V, E)$ by scanning the entire program.

Simplify. After *build*, the nodes in G are, respectively, examined. Each node $v \in V$ with a degree $< C$ (less than C neighbors) is removed from G and pushed onto the stack. Relevant edges are also removed from graph G.

Spill. If there exists a node with degree $\geq C$, it will be chosen as a potential spill candidate. Once a node is marked for spilling, the node is then deleted from the graph G and pushed onto the stack.

Select. Repeatedly pop the nodes from stack and reinsert them into G. If v is not a potential spilling candidate, v can be assigned a free color. If v is a potential spilling, v may be trivially colorable; that is, it will get assigned a color. Otherwise, the node is marked for an actual spilling and remained uncolored.

Start Over. If v is marked for spilling, an additional store is inserted after every definition, and a load is inserted before every use. The whole graph coloring process is started all over again.

A critical issue of register allocation is which node v should be selected as a potential spilling candidate. Several approaches have been proposed to make decisions according to the sequence which registers, the degree and the number of operation o, respectively (use or define v) [19]. However, these spilling policies assume uniform write distribution and hence will fail to choose the most energy-efficient node from the potential spilling list if MLC STT-RAM is employed as register. In this paper, considering unbalanced write distribution of MLC STT-RAM, a cost model estimating node spilling cost is proposed to derive a highly efficient register allocation approach.

Table 4: A statistic study of two consecutive writes to registers of eight 2-bit MLLC STT-RAM cells.

	Node *a*	Node *b*	Node *c*
Old value	0001000100010110	0001100010100001	0100000010000001
New value	1010110100011110	0011100011000000	0000010011110001
Number of transactions			
ZT	4	5	4
ST	0	1	3
HT	2	2	1
TT	2	0	0
Statistics of transactions			
Soft transitions	**2**	1	3
Hard transitions	**4**	2	1

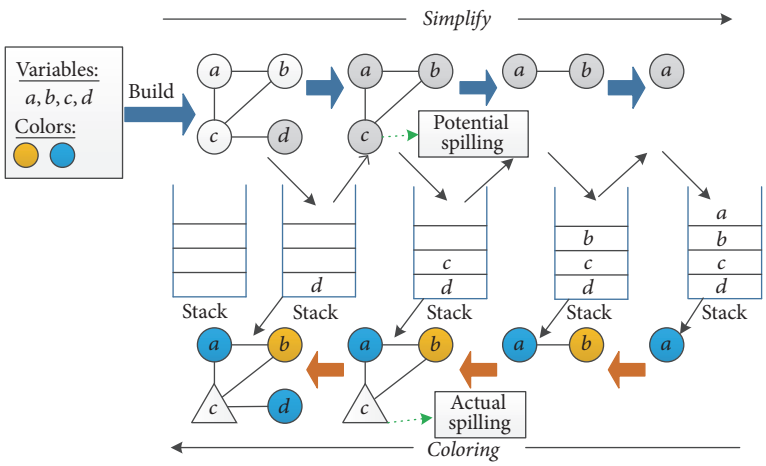

Figure 1: A 2-coloring example in conventional register allocation.

2.3. A Motivational Example. In this section, a motivational example is presented to show how the unbalanced costs of different write state transitions impact the spilling decision for MLC STT-RAM-based registers.

The example in Figure 1 shows a 2-coloring problem in a manner of conventional register allocation. It is assumed that four variables should be allocated with two registers. In the *Simplify* phase, the node *d* will be first deleted from the interference graph and pushed onto the stack. Then there does not exist any node with a degree less than two. In this case, any one of the *a*, *b*, *c* nodes can be selected for potential spilling. In the conventional approach, the three nodes are all added into the spilling list, and the compiler chooses the to-be-spilled nodes without any priority. In the example in Figure 1, node *c* is chosen as the potential spilling target in the *Simplify* phase and is spilled one in the *coloring* phase.

In this work, since we consider registers built by MLC STT-RAM where writes with different state-transitions cost different energy, the conventional approach is not appropriate any more. Table 4 presents an example of programming a 16-bit MLC STT-RAM. It is assumed that the old value of node *a* is "00 01 00 01 00 01 01 10" and the new value to-be-written is "10 10 11 01 00 01 11 10". The old values and new written values in nodes *b* and *c* are given in Table 4 as well. We also collect the numbers of the aforementioned state transitions. It has been presented that a TT implies one ST and one HT and a ZT for no transitions. As such, we can convert the above transitions by counting soft transitions and hard transitions as shown in the lower right part of Table 4. The results indicate that writing node *a* costs the highest energy. It is therefore preferable to spill node *a*.

The observation indicates the impact of different state-transition costs on the potential spilling decision during register allocation. *Different from conventional register allocation policies, the spilling costs with different state transitions are nonuniform in MLC STT-RAM-based registers.* Motivated by this consideration, a spilling policy guided by state-transition cost analysis is proposed so as to reduce energy consumption in MLC STT-RAM.

3. A State-Transition-Aware Spilling Heuristic

This section first describes the framework overview of the proposed approach and then presents the spilling cost model driven by state transition of MLC STT-RAM. Finally, the algorithm for SSCM-based register allocation is presented.

3.1. Framework Overview. Previous heuristics as described in Section 2.2 usually employ simple spilling principles. Due to the lack of a formal cost model, these heuristics fail

to estimate the impact of a spilling decision on program code quality. Furthermore, since they all target SRAM-based registers where write cost of different values is uniform, none of them examine the write operation state. In other words, spilling decisions are independent of the actual cost model.

In this paper, we propose a cost-based method to choose spilling variables when MLC STT-RAM is employed as the register. In order to build a formal spilling cost model, we explore the unbalanced writes to the hard domain and soft domain of MLC STT-RAM cells and the exact state-transition cost to identify the spilling cost of each node. Then, spilling candidates are selected according to their spilling costs in the *spill* phase. In the following subsections, a qualitative state-transition model is first constructed for cost assessments. Then, the heuristic of SSCM-based register allocation is depicted. This algorithm extends the capability of Chaitin's algorithm [5] in spilling-optimization ways. Compared to traditional Chaitin's register allocation, SSCM-based register allocation can retain more cost-efficient variables in registers, thus delivering promising reduction in terms of energy consumption.

3.2. A Spilling Cost Model. In this subsection, a spilling cost model is presented to illustrate the spilling priority, determined based on state-transition profiling information of MLC STT-RAM.

We assume that the write frequency or the number of transitions of each state can be obtained through profiling. Considering a MLC STT-RAM with 2 bits per cell, the state S contains $2^2 = 4$ states. The write frequency of state set S can be calculated as follows:

$$F = \sum_j S_{ji}, \quad i, j \in [0, 3], \tag{1}$$

where S_{ji} represents the number of transitions from state S_i to state S_j.

The number of the four state transitions can be collected by the following model:

$$F(\mathrm{ZT}) = \sum_j S_{ji}(\mathrm{ZT}), \quad i, j \in [0, 3], \ i \neq j. \tag{2}$$

The other three states can be obtained in a similar way.

Subsequently, the cost model of a variable can be constructed as the linear combination of F(ZT), F(ST), F(HT), and F(TT), represented by

$$\begin{aligned}\mathrm{Cost} = {} & \alpha \cdot F(\mathrm{ZT}) + \beta \cdot F(\mathrm{ST}) + \gamma \cdot F(\mathrm{HT}) + \delta \\ & \cdot F(\mathrm{TT}),\end{aligned} \tag{3}$$

where α, β, γ, δ are defined as the weight of every state-transition frequency. In this paper, since we focus on the dynamic energy saving, the weight is defined as the execution energy of different state transition. The dynamic energy of state transition E_{XT} is calculated in direct proportion to the product of the square of every transition's average switching current and the pulse duration:

$$E_{\mathrm{XT}} \propto I_{\mathrm{wirte(XT)}}{}^2 \cdot t_{\mathrm{pulse}}, \quad \mathrm{XT} \in [\mathrm{ZT}, \mathrm{ST}, \mathrm{HT}, \mathrm{TT}]. \tag{4}$$

Here, $I_{\mathrm{wirte(XT)}}$ denotes the required average switching current of every state transition XT and can be obtained by Table 3, while t_{pulse} denotes the pulse duration. Then weights α, β, γ, δ can be obtained by normalizing E_{XT} to 0-1.

We calculate the write energy of every energy in MLC STT-RAM at 45 nm technology node based on data reported in [18, 20] and assume that 10 ns pulse duration is applied. By profiling the frequencies of the four transition events, α, β, γ, and δ can be obtained by normalizing the average energy of ZT, ST, HT, TT to 0-1. In this way, the spilling cost model can be constructed according to (3).

Once the parameters have been finalized, we can obtain the cost for each node in graph G according to (3). Then the nodes with degree greater than k are sorted based on their write cost in descending order. Finally, the node with the highest cost is selected as the spilling candidate. In this way, the model for spilling cost minimization can be constructed. We use the same cost model as the measurement of spilling priority for every remaining node. If a node with the highest priority is spilled, the register energy pressure can be reduced. In this way, the allocator can make a better decision on register assignment based on the exact STT-RAM register state-transition usage information.

Overall, the procedure of building spilling cost model is shown in Figure 2, while the entire implementation process is depicted in Algorithm 1.

The spilling cost model provides a sound basis for selecting potential spilling nodes. By keeping the node (variable) with less transition energy in register instead of memory, it helps avoid expensive spills when considering the state-transition costs of MLC STT-RAM.

3.3. Algorithm Description. This subsection describes the proposed SSCM-based register allocation algorithm. The basic idea is to choose the potential spilling candidate with the high spilling priority which is determined by the variable's write transition cost. The goal is to spill the node with relatively expensive write cost to memory so as to relieve the register pressure and maximize energy saving during program execution. The SSCM-based register allocation mainly consists of four steps.

Step 1. An interference graph G is employed as the basic data structure for graph coloring. Then repetitively, the variable v with degree $\leq k$ is deleted from the interference graph G, until no node with degree $\leq k$ remains.

Step 2. It is assumed that G' is the graph resulting from G by successively deleting nodes with degree less than k. If G' is empty, then color the variables in reverse order of deleting.

Step 3. The number of different write state transitions of the remaining nodes is counted through profiling. Then the cost of each variable can be obtained by (3). Subsequently, the variables are sorted in descending order of spilling cost. The variable with the greatest spilling cost is marked for a potential spilling. Then the allocator gets the variable colored.

(1) The write numbers for each state are first collected;
(2) The frequency of every state transition can be calculated by Equation (2);
(3) The spilling cost of every node (variable) can be obtained based on frequency analysis;
(4) The node is sorted based on write cost in descending order;
(5) The highest cost node is selected for spilling;

Algorithm 1: The procedure of building spilling cost model.

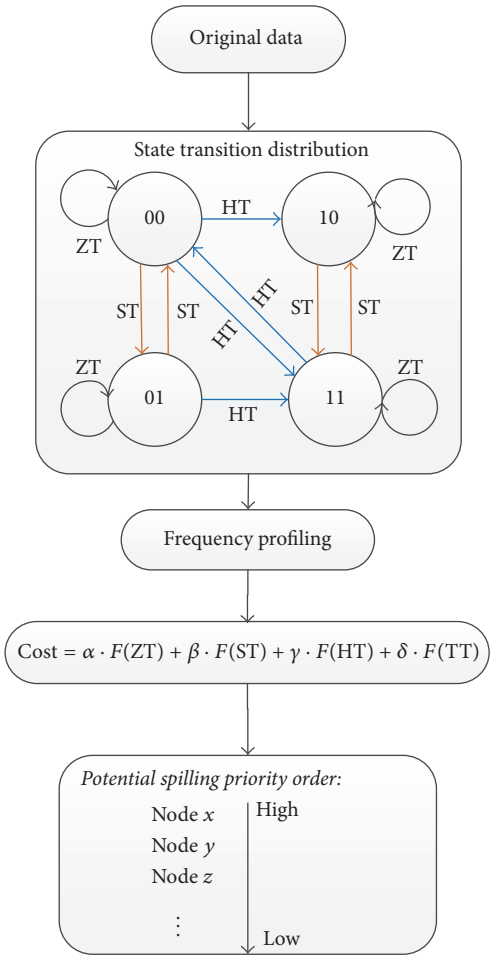

Figure 2: The procedure of building spilling cost model.

Step 4. If no color is available for the spilled variable, then stop. Otherwise, the allocator will insert the spill node, rebuild the interference graph, and start over.

The algorithm is shown in Algorithm 2 in detail. When the algorithm cannot find a variable that is trivially colorable, some variables need to be spilled (line (4)). The algorithm chooses the variable with the highest spilling cost as the potential spilling candidate (line (7)). If the variable is not colored, it is marked for an actual spilling (line (14)).

As discussed previously, the proposed optimistic coloring can lead to more energy-efficient register allocation by considering the nonuniform state transitions of MLC STT-RAM-based registers.

3.4. Discussion Regarding Input-Dependence. One typical concern with most profiling-based optimizations is input-dependence, that is, whether the optimizations made for a specific set of inputs will be preserved for other inputs of the same application. For the proposed SSCM scheme, it is clear that the spilling cost models are fixed given a specific programming strategy, while the write state-transition frequencies of each state vary across different applications and across different inputs. However, the optimality of SSCM depends not on the values of F, but only the descending order of node write costs. In other words, once a spilling decision is made based on a set of input, this decision preserves the maximal cost reduction for other inputs as long as the descending order of nodes remains the same, even with various frequency values. In addition, the previous work [21] focusing on workload characterization showed the workload characterization strategies provide potential to improve the accuracy of offline prediction of the proposed SSCM policy.

In the experimental evaluation, this paper, same as the work in [22], assesses all the test benches with various inputs and studies the differences in cost reduction. Regarding the proposed SSCM, two cases are evaluated: *SSCM_ideal* and *SSCM_practical*. *SSCM_ideal* customizes the spilling decision for different inputs of the same program, while *SSCM_practical* makes the spilling decision for one input and applies it to other input configurations. A comparison between the two cases shows that the impact of input variations on the optimality of SSCM is negligible, thus confirming that profiling can be done on one specific input and *SSCM_practical* can be employed.

4. Experiment

In this section, the experimental setup is introduced first. Then, the experimental results for evaluating the efficacy of proposed SSCM methods are presented.

4.1. Experimental Setup. We evaluate how the proposed SSCM impacts on dynamic energy and lifetime of MLC STT-RAM. The architectural parameters of the MLC STT-RAM registers are listed in Table 5 [23].

Benchmarks are selected from DSP programs and Livermore benchmarks in the experiments. Using the LLVM [24], the corresponding assembly code and the register write state-transition profiling can be obtained. Then the cost model can be built to guide the proposed state-transition-aware spilling heuristic in register allocation. All the experiments are implemented with the *SSCM_practical* deployment.

```
(1)  while G not empty do
(2)      if there is an v with degree ≤ k then
(3)          delete v
(4)      else
(5)          obtain the frequency set F by offline profiling
(6)          sort variables based on the descending write cost
(7)          choose v with MAX COST
(8)          add v to spilling_list
(9)          delete v
(10)     end if
(11)     if no variable has been spilled then
(12)         color the variables in reverse order of deleting
(13)     else
(14)         spill each v ∈ spilling_list everywhere
(15)         rebuild the interference graph and repeat the procedure
(16)     end if
(17) end while
```

Algorithm 2: SSCM-based register allocation algorithm.

Table 5: The configurations of MLC STT-RAM.

	MLC STT-RAM
Total read time (ns)	S:1.25 H:1.63
Total write time (ns)	S:7.18 H:14.86
Read energy	S:0.018 H:0.023
Write energy	S:0.087 H:0.14
Wearing/per write	Hard domain: 0, 0, 1, 1 for ZT, ST, HT and TT, respectively
	Soft domain: 0, 1, 1, 2 for ZT, ST, HT and TT, respectively

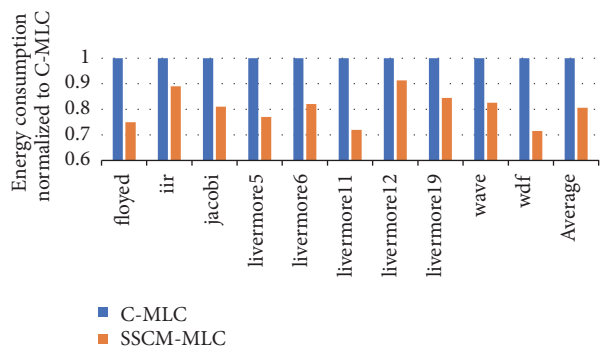

Figure 3: Energy evaluation under C-MLC and SSCM-MLC register design.

4.2. Experimental Results. Typically, a register file is accessed in a single cycle. The cycle length is sized for the worst case. Thus, all accesses take the same amount of time. In this section, the proposed SSCM-MLC STT-RAM scheme is evaluated against the MLC STT-RAM with traditional register allocation in terms of energy efficiency and lifetime.

4.2.1. Dynamic Energy. The consumed energy is accumulated by each 2-bit state transition in the register. Each register is 64-bit long and the bits in the same register can be programmed simultaneously [7]. For every register, the overall energy consumption is determined by the product of each state to program and the energy of each state. So the energy improvement is impacted from the number and type of state transitions. Figure 3 presents the results of energy consumption of the SSCM scheme (SSCM-MLC) compared with conventional register allocation applied to MLC STT-RAM without considering the spilling priority (C-MLC). The results shown in Figure 3 are normalized to the C-MLC scheme. As is shown in Figure 3, for all benchmarks, *wdf* achieves the highest energy reduction. The reason lies that the hard/two-step transitions variables of *wdf* are spilled to memory and low energy zero/soft transitions variables are kept in register. It can be seen that the benchmark *livermore12* is smaller than others. The underlying reason is that *livermore12* has more soft transitions and zero transition. And the zero/soft transition consumes less energy than hard/two-step transition. The proposed SSCM policy spills a large amount of the zero/soft transition variable of *livermore12*. As a result, the overall energy consumption of *livermore12* is minimal. On average, the proposed SSCM saves energy by 19.4% over C-MLC. This is mainly due to the fact that the proposed SSCM policy is able to retain the energy-efficiency variable in the register, thus saving more write energy.

4.2.2. Lifetime Evaluation. The best endurance test result for SLC STT-RAM devices so far is less than 4×10^{15} cycles [25]. For MLC STT-RAM, the larger write current exponentially degrades the lifetime of register as a result of dielectric breakdown. Furthermore, the frequent access to registers also attribute to lifetime reduction. For two registers with

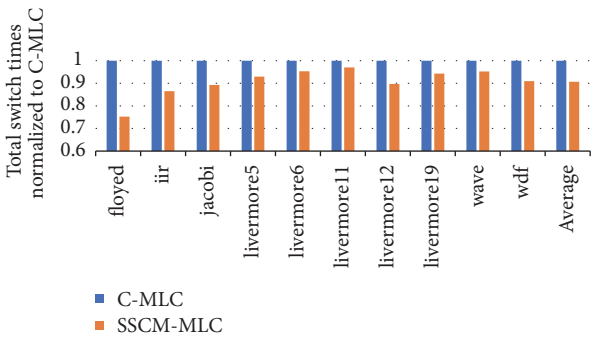

Figure 4: Normalized total write times under C-MLC and SSCM-MLC register design.

the physical properties, their lifetimes are decided by the number of writes (switch times under a write operation). The switch times of an MLC cell under a write operation are counted in hard domain and soft domain, separately. In Figure 4, the total number of switches represents the sum of the soft domain and the hard domain. The results show that the proposed SSCM design achieves greater switch reduction than C-MLC. Specifically, the total number of switches to soft and hard domains is reduced by 9.35%, on average. This is mainly because the SSCM scheme spills more variables with two-step state transition to memory, thus reducing the total number of switches. Overall, the MLC STT-RAM lifetime is improved by 23.2% compared to C-MLC design. As is shown in Figure 4, the switching time of the benchmark *floyed* is smaller than others. This is mainly because there are more two-step state transitions in the benchmark *floyed* so that the SSCM scheme spills more variables with the two-step state transition to memory, thus reducing the total number of switches. It can be observed that the switching time of the benchmark *livermore11* is larger than others in Figure 4. The reason lies that there are more zero state transitions in the benchmark *livermore11*. The proposed SSCM scheme spills more variables with the zero state transition to memory, thus reducing less number of switches than others.

5. Conclusions

This paper has proposed a state-transition-aware spilling cost minimization (SSCM) scheme for energy reduction in MLC STT-RAM-based register design. First an energy cost model is built to quantitatively calculate spilling cost of each variable with a degree larger than k colors. Then the algorithm for SSCM-based register allocation is presented to choose the variable with the highest write cost to be spilled and assign the physical register to other variables. Experimental results show that the proposed SSCM scheme can achieve promising cost reduction in terms of energy consumption of registers and enlarge MLC STT-RAM lifetime as well.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is supported by the grants of Beijing Advanced Innovation Center for Imaging Technology, National Natural Science Foundation of China [Project no. 61502321], and the Project of Beijing Municipal Education Commission [Project no. KM201710028016].

References

[1] M. Ranjbar Pirbasti, M. Fazeli, and A. Patooghy, "Phase Change Memory lifetime enhancement via online data swapping," *Integration, the VLSI Journal*, vol. 54, pp. 47–55, 2016.

[2] S. Mittal, "A survey of techniques for architecting processor components using domain-wall memory," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 13, no. 2, article no. 29, 2016.

[3] Q. Li, L. Shi, C. J. Xue et al., "Access characteristic guided read and write cost regulation for performance improvement on flash memory," in *Proceedings of the in USENIX Conference on File and Storage Technologies (FAST 16)*, pp. 125–132, Santa Clara, Calif, USA, 2016.

[4] R. Bishnoi, M. Ebrahimi, F. Oboril, and M. B. Tahoori, "Improving Write Performance for STT-MRAM," *IEEE Transactions on Magnetics*, vol. 52, no. 8, 2016.

[5] G. J. Chaitin, M. A. Auslander, A. K. Chandra, J. Cocke, M. E. Hopkins, and P. W. Markstein, "Register allocation via coloring," *Computer Languages*, vol. 6, no. 1, pp. 47–57, 1981.

[6] K. D. Cooper and A. Dasgupta, "Tailoring graph-coloring register allocation for runtime compilation," in *Proceedings of the 4th International Symposium on Code Generation and Optimization, CGO 2006*, pp. 39–49, USA, March 2006.

[7] M. Zhao, Y. Xue, C. Yang, and C. J. Xue, "Minimizing MLC PCM write energy for free through profiling-based state remapping," in *Proceedings of the 2015 20th Asia and South Pacific Design Automation Conference, ASP-DAC 2015*, pp. 502–507, Japan, January 2015.

[8] C. J. Xue, Y. Zhang, Y. Chen, G. Sun, J. J. Yang, and H. Li, "Emerging non-volatile memories: Opportunities and challenges," in *Proceedings of the Embedded Systems Week 2011, ESWEEK 2011 - 9th IEEE/ACM International Conference on Hardware/Software-Codesign and System Synthesis, CODES+ ISSS'11*, pp. 325–334, Taiwan, October 2011.

[9] W. Wen, Y. Zhang, M. Mao, and Y. Chen, "State-restrict MLC stt-ram designs for high-reliable high-performance memory system," in *Proceedings of the 51st Annual Design Automation Conference, DAC 2014*, USA, June 2014.

[10] X. Chen, N. Khoshavi, J. Zhou et al., "AOS: Adaptive overwrite scheme for energy-efficient MLC STT-RAM cache," in *Proceedings of the 53rd Annual ACM IEEE Design Automation Conference, DAC 2016*, USA, June 2016.

[11] D. Chabi, W. Zhao, J.-O. Klein, and C. Chappert, "Design and analysis of radiation hardened sensing circuits for Spin transfer torque magnetic memory and logic," *IEEE Transactions on Nuclear Science*, vol. 61, no. 6, pp. 3258–3264, 2014.

[12] G. Tsiligiannis, L. Dilillo, A. Bosio et al., "Testing a commercial MRAM under neutron and alpha radiation indynamic mode," *IEEE Transactions on Nuclear Science*, vol. 60, no. 4, pp. 2617–2622, 2013.

[13] Y. Lakys, W. S. Zhao, J.-O. Klein, and C. Chappert, "Hardening techniques for MRAM-based nonvolatile latches and Logic,"

IEEE Transactions on Nuclear Science, vol. 59, no. 4, pp. 1136–1141, 2012.

[14] N. Goswami, B. Cao, and T. Li, "Power-performance co-optimization of throughput core architecture using resistive memory," in *Proceedings of the 19th IEEE International Symposium on High Performance Computer Architecture, HPCA 2013*, pp. 342–353, China, February 2013.

[15] J. Wang and Y. Xie, "A write-aware STTRAM-based register file architecture for GPGPU," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 12, no. 1, article no. 6, 2015.

[16] H. Zhang, X. Chen, N. Xiao, and F. Liu, "Architecting energy-efficient STT-RAM based register file on GPGPUs via delta compression," in *Proceedings of the 53rd Annual ACM IEEE Design Automation Conference, DAC 2016*, USA, June 2016.

[17] http://www.mram-info.com/tags/companies/ibm.

[18] Y. Chen, X. Wang, W. Zhu et al., "Access scheme of multi-level cell spin-transfer torque random access memory and its optimization," in *Proceedings of the 53rd IEEE International Midwest Symposium on Circuits and Systems, MWSCAS 2010*, pp. 1109–1112, USA, August 2010.

[19] H. Falk, "WCET-aware register allocation based on graph coloring," in *Proceedings of the 2009 46th ACM/IEEE Design Automation Conference, DAC 2009*, pp. 726–731, usa, July 2009.

[20] X. Lou, Z. Gao, D. V. Dimitrov, and M. X. Tang, "Demonstration of multilevel cell spin transfer switching in MgO magnetic tunnel junctions," *Applied Physics Letters*, vol. 93, no. 24, Article ID 242502, 2008.

[21] P. Bogdan, "Mathematical modeling and control of multi-fractal workloads for data-center-on-a-chip optimization," in *Proceedings of the 9th IEEE/ACM International Symposium on Networks-on-Chip, NOCS 2015*, Canada, September 2015.

[22] M. Zhao, Y. Xue, J. Hu et al., "State Asymmetry Driven State Remapping in Phase Change Memory," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 1, pp. 27–40, 2017.

[23] X. Liu, M. Mao, X. Bi, H. Li, and Y. Chen, "An efficient STT-RAM-based register file in GPU architectures," in *Proceedings of the 2015 20th Asia and South Pacific Design Automation Conference, ASP-DAC 2015*, pp. 490–495, Japan, January 2015.

[24] C. Lattner and V. Adve, "LLVM: a compilation framework for lifelong program analysis & transformation," in *Proceedings of the International Symposium on Code Generation and Optimization (CGO '04)*, pp. 75–86, March 2004.

[25] H. Luo, J. Hu, L. Shi, C. J. Xue, and Q. Zhuge, "Two-step state transition minimization for lifetime and performance improvement on MLC STT-RAM," in *Proceedings of the 53rd Annual ACM IEEE Design Automation Conference, DAC 2016*, USA, June 2016.