Extended Abstracts of the 2017 International Conference on Solid State Devices and Materials, Sendai, 2017, pp977-978

PS-12-05

Energy-Efficient High-Performance Nonvolatile VLSI Processor with a Temporary-Data Reuse Technique

Masanori Natsui and Takahiro Hanyu
Tohoku University
2-1-1 Katahira, Aoba-ku, Sendai 980-8577, JAPAN, Phone: +81-22-217-5552, E-mail: natsui@riec.tohoku.ac.jp

Abstract

An instruction fetch acceleration technique for MRAM-embedded nonvolatile VLSI processor is proposed. The proposed technique realizes efficient instruction fetch by eliminating redundant memory access by considering the code length of the instruction to be fetched and the transition of the memory address to be accessed. Through the evaluation using a general purpose 32-bit microprocessor, it is demonstrated that the proposed technique increases the peak efficiency of the system up to 1.37 times, while achieves 4.6 times area reduction compared with cache-based one.

1. Introduction

Research and development for realizing a high performance / low power sensor node for IoT utilizing a nonvolatile memory element such as MTJ device [1] has been actively conducted in recent years. In order to realize high-speed operation in sensor nodes where wide operating temperature is assumed, it is important to eliminate the bottleneck of memory access. Figure 1 shows a shmoo plot for read / write operations of an experimental embedded STT-MRAM using 2T-2MTJ cell structure. Since the operating frequency of the MRAM designed using the advanced process is highly dependent on the temperature, raising the operating frequency will narrow the temperature guaranteeing the operation.

One of the most popular ways to solve the memory bottleneck is the introduction of caches [2]. However, this leads to an increase in area and power consumption in circuit implementation. Additionally, it is necessary to make the cache itself nonvolatile in order to utilize the benefit of nonvolatile power gating. Another way is to introduce a memory interleaving [3]. However, since most of the microprocessors in recent years have a variable-length instruction set, regular memory access which is indispensable for effective interleaving is not performed in many cases.

From these viewpoints, in this paper, we propose a technique for improving the processing speed in a microprocessor based on memory access multiplexing with a circuit technique that dynamically changes the frequency according to the code length of the executed instruction.

2. Instruction Fetch Acceleration Technique

Figure 2 shows a basic concept of the proposed technique. In this paper, we consider a system consisting a microprocessor based on ARM Cortex-M0 and an embedded STT-MRAM. This chip has an instruction set called Thumb-2, which contains both 16-bit instructions and 32-bit instructions. A memory access is always performed using a 32-bit bus (HADDR, HRDATA), and whether it is a 16-bit instruction or a 32-bit instruction is judged at the stage of instruction fetch. In the case of a 16-bit instruction, an instruction is executed using only either the upper 16 bits or the lower 16 bits of the 32-bit data received from the memory. In this case, half of the read data is wasted and redundant accesses will occur.

In the proposed architecture in Figure 2, the read data is temporarily held in registers in the accelerator module (reg0, reg1), and when access to the same memory address is repeated, the data held in the register is reused instead of the memory. Furthermore, in the case of 32-bit instructions allocated at consecutive memory addresses, it is also possible to double the speed of memory access by storing data of two 32-bit instructions into the registers by interleaving and applying the same control. As a result, it becomes possible to conceal the bottleneck in the memory access, and realize high speed instruction fetch.

3. Evaluation

Figure 3 shows a part of the test chip layout of an experimental nonvolatile VLSI Processor using a 40nm MOS/MTJ process, which is designed by using an automated design flow and cell libraries for the MTJ-based NV-LIM LSI [4, 5]. From the number of gates of each block, the area overhead due to introducing the proposed circuit is estimated to be about 15%, which achieves 4.6 times area reduction compared with a conventional implementation using cache. Note that each block is separately designed for overhead evaluation in this figure, however it is also possible to integrate these blocks and layout them as one circuit block. In that case the area overhead is expected to be even smaller.

Figure 4 shows a simulated waveform of the proposed accelerator unit. In this example, (1) 16-bit instructions allocated at consecutive memory addresses, (2) branch instructions to access a memory address not consecutive, and (3) 32-bit instructions allocated at consecutive memory addresses are sequentially executed. We can confirm that the operating frequency is dynamically changed depending on whether or not the transition of the memory address to be accessed satisfies the condition of instruction fetch acceleration.

Table I compares the performance of a microprocessor incorporating the proposed technique and that of the conventional ones. Although the power consumption increases due to the addition of the accelerator, the proposed technique can double the operating frequency of the CPU without changing the required performance to the memory. As a result, the efficiency (MIPS/mW) can be improved up to 1.37 times while guaranteeing memory read / write operation over a wide temperature range.

4. Conclusion

The proposed technique enables faster instruction fetch

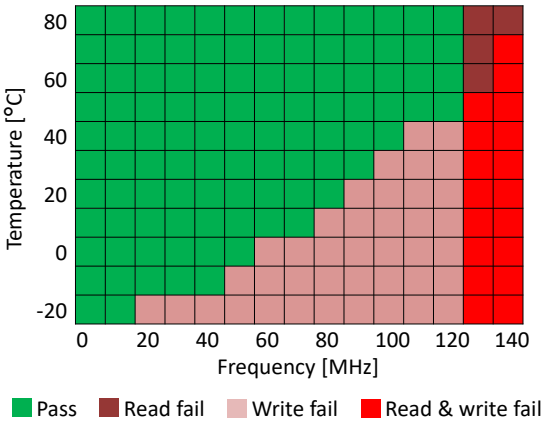


Fig. 1 Shmoo plot of an embedded MRAM.

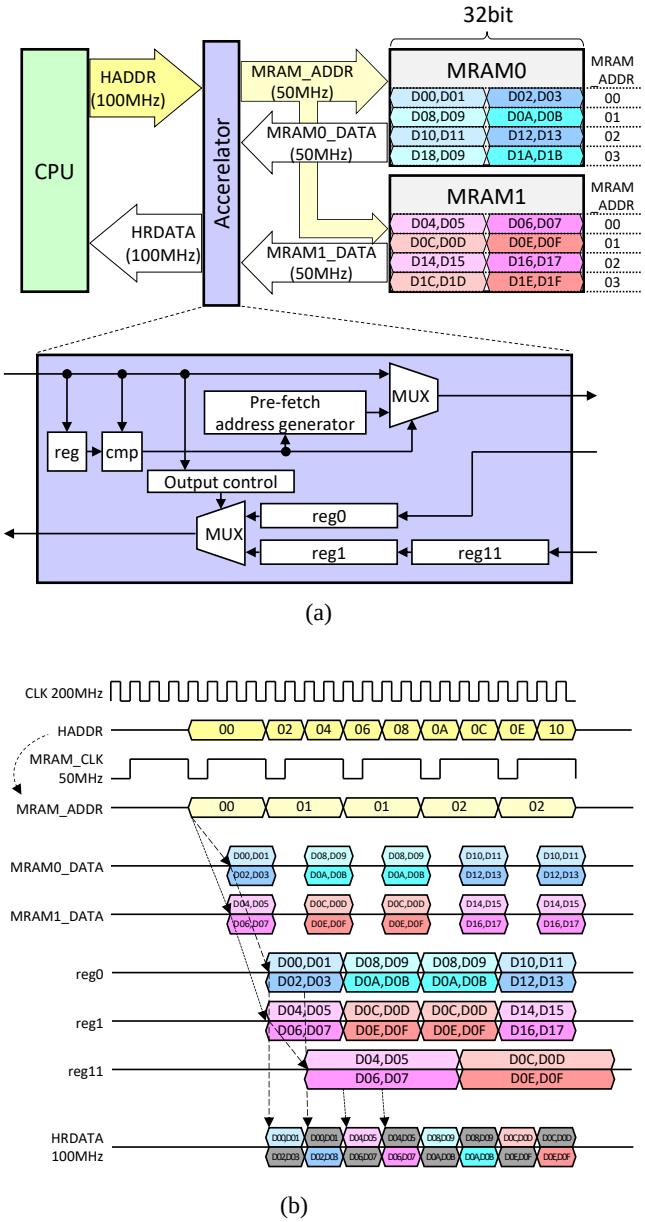
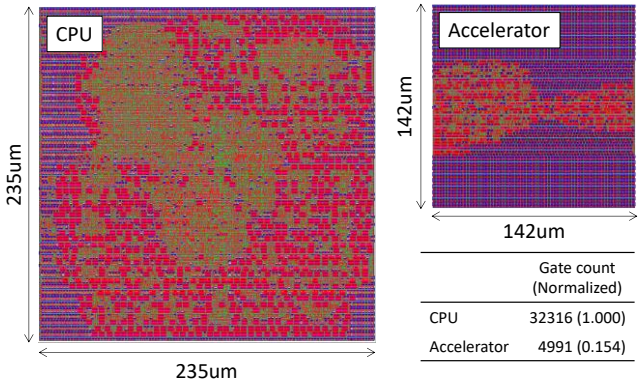


Fig. 2 Proposed architecture: (a) circuit architecture, (b) example data transition on a read operation of consecutive 16-bit instructions.



	Gate count (Normalized)
CPU	32316 (1.000)
Accelerator	4991 (0.154)

Fig. 3 Chip layout.

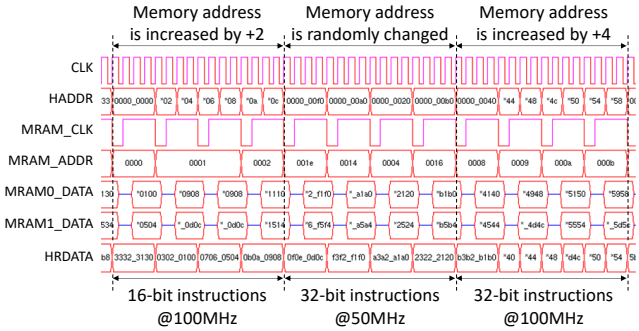


Fig. 4 Simulated waveform.

Table I Performance comparison.

	Conventional		Proposed
	w/o cache	w/ cache*	
Area ratio	1.00	7.08	1.54
Voltage [V]	1.1	1.1	1.1
Frequency [MHz]	50	50/100	50/100
Peak perf. [MIPS]	49.56	99.12	**99.12**
Power [mW]	0.334	0.571	0.487
Peak efficiency ratio	1.00	1.17	**1.37**

*Area and power of cache are estimated based on [6]

without changing required performance for embedded nonvolatile memory by utilizing the properties of microprocessor. In the future, we will show the effectiveness of the proposed technique through the performance evaluation of the entire VLSI system including MRAM, other peripherals, and interconnect bus among them.

Acknowledgements

The authors thank Y. Takako of Focal Agency for excellent technical assistance. Part of this research was supported by the JSPS IMPACT Program, R&D for Next-Generation IT of MEXT of Japan, and JSPS KAKENHI Grant Number 16KT0187.

References

[1] S. Ikeda, et al., Nat. Mater. 9, 721 (2010).
[2] A. J. Smith, ACM Computing Surveys 14(3), 473 (1982).
[3] K. Hwang et al., McGraw-Hill (1984).
[4] M. Natsui, et al., 2013 IEEE ISSCC, 194 (2013).
[5] M. Natsui, et al., IEEE JSSC 50, 476 (2015).
[6] T. Fukuda, et al., 2014 IEEE ISSCC, 236 (2014).