A Low-Latency DMR Architecture with Efficient Recovering Scheme Exploiting Simultaneously Copiable SRAM

Go Matsukawa[1], Yohei Nakata[1], Yuta Kimi[1], Yasuo Sugure[2], Masafumi Shimozawa[3], Shigeru Oho[4], Hiroshi Kawaguchi[1], and Masahiko Yoshimoto[1,5]
[1]Graduate School of System Informatics, Kobe University, Kobe, Japan
[2]Central Research Laboratory, Hitachi, Ltd., Yokohama, Japan
[3]Hitachi Solutions, Ltd., Yokohama, Japan
[4]Department of Electrical and Electronics Engineering, Nippon Institute of Technology, Minamisaitama, Japan
[5]Japan Science and Technology Agency (JST) CREST, Tokyo, Japan

***Abstract*—This paper presents a novel architecture for a fault-tolerant high-performance system using a checkpoint/restart approach with dual modular redundancy (DMR). The proposed architecture can perform low-latency copy with instantaneously copiable SRAM. Furthermore, we can use an instantaneous comparison scheme that has more fault coverage than comparison with a cyclic redundancy check (CRC). Evaluation results show that, compared with the conventional checkpoint/restart DMR, the proposed architecture reduces the cycle time by 97.8% and achieves a 3.28% low-latency execution cycle even if a one-time fault occurs when executing the task.**

Keywords— dual modular redundancy; checkpointing; fault-tolerance

I. INTRODUCTION

Microprocessors are a key component used for widely various applications. Processors used in safety-critical systems such as vehicles and social infrastructure must operate with high reliability. Nevertheless, processors are increasingly sensitive to soft errors and power supply noise with technology scaling, which increases threshold voltage (V_{th}) variation because of random dopant fluctuation. The V_{th} variation degrades processor reliability. These factors engender faults in the processor. Consequently, reliable processors are necessary to detect faults and to recover from a faulty state even if faults occur in the processor.

To detect faults, Dual Modular Redundancy (DMR) with a recovery scheme has been applied to reliable processors [1–3]. Two cores in the DMR processor execute the same task simultaneously in parallel. The processor detects faults by comparing states of the cores (register values and stored data in the memory) at every checkpoint interval. If the states match, then the register values of the cores are copied into backup shadow registers for subsequent recovery. If the states differ, then the cores load the shadow register values to recover the recent fault-free state. The checkpoint and recovery technique enables detection of faults and recovery from a faulty state. However, the copy of register values at the checkpoint and loading of these values for recovery impose additional latency. Unless the latency is low, the technique cannot be applied to systems such as vehicle control systems. A conventional DMR architecture uses a Cyclic Redundancy Check (CRC) code for comparison and bus transfer for the copying of register values [4]. Fault-detection capability of the comparison of CRC code depends on the fault location. The bus transfer is processed sequentially. The bus transfer latency increases along with the number of registers.

Herein, we propose a DMR architecture that has a low latent recovery scheme. To realize low latency, the proposed DMR architecture exploits block-level simultaneously copiable SRAM. In addition, exploiting block-level instantaneous comparison SRAM improves the fault detection capability.

II. PROPOSED DMR ARCHITECTURE WITH A RECOVERY SCHEME

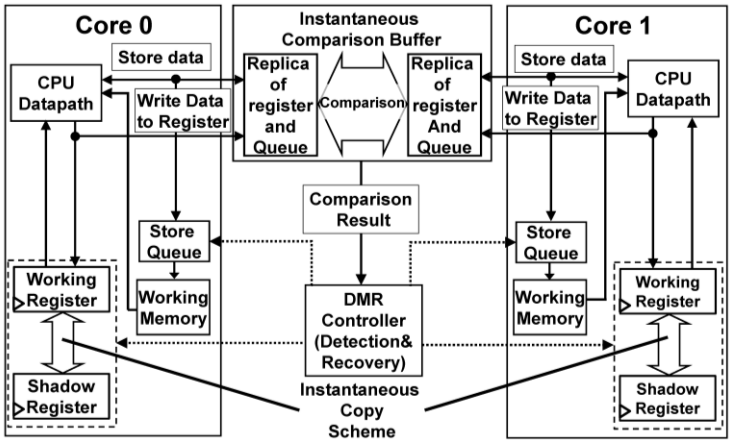


Figure 1. Proposed DMR Architecture with recovery scheme.

In this section, we present an overview of the proposed DMR architecture with a recovery scheme. Then the checkpoint/restart scheme is explained. Finally we explain differences from conventional DMR. Figure 1 portrays the proposed DMR architecture, which consists of two cores, an instantaneous comparison buffer, and a DMR controller. The proposed architecture calculates or controls using working

memory, store queue, and working register data in normal operation. Then, output data are written to the store queue or working register, and are written simultaneously in the instantaneous comparison buffer. As shown in Figure 2, normal operations begin during checkpoint intervals. The instantaneous comparison buffer compares both replicas of the register and queue in the core. Figure 3(a),(b) depicts a data transfer between Store Queue and Working Memory. The comparison result is transferred to the DMR controller. If results of the comparison indicate a match, then data of the store queue are written in working memory in the next normal operation as shown in Figure 3(a). Data of the working register are copied in the shadow register using simultaneous data copy. If the result of comparison is a mismatch, then the store queue data are erased and data of the shadow register are transferred to the working register. Then the process restarts from the last checkpoint.

Conventional copying between the working register and shadow register is executed via the shared bus, but latency rises proportionately to the number of registers. Moreover, normal comparison data between the dual cores is performed via the bus, so latency increases along with the data size. Although the comparison of CRC is also often used to reduce the number of comparison cycles, CRC might be unable to detect multibit error.

The proposed architecture is valid for transient fault such as soft error. On the other hand, permanent fault can not be covered because if permanent fault is occurred this architecture will repeat fault detection and recovery during execution.

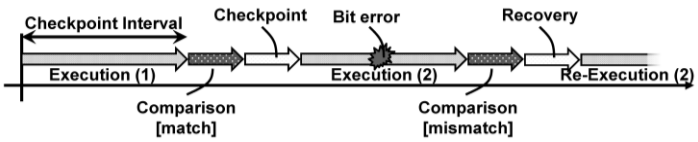
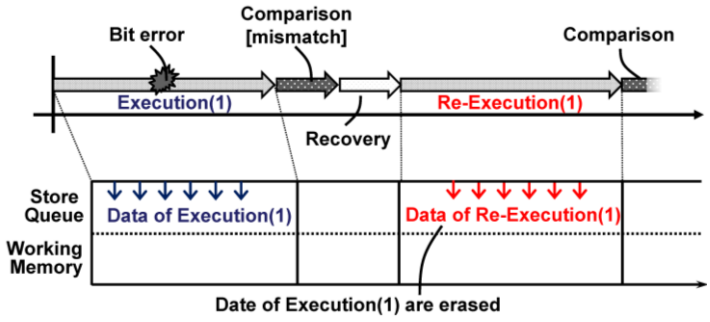
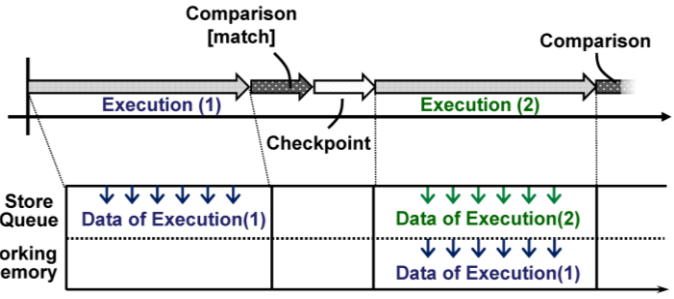


Figure 2. Execution with checkpoint and recovery.



(a)



(b)

Figure 3. Date between Store Queue and WorkingMemory: (a) comparison match and (b) comparison mismatch

III. INSTANTANEOUS COMPARISON AND SIMULTANEOUS COPY WITH 7T MEMORY CELL

The proposed DMR architecture has instantaneous comparison and a simultaneous copy scheme structure with 7T SRAM. Herein, we explain the instantaneous comparison and the simultaneous copy scheme.

A. 7T SRAM

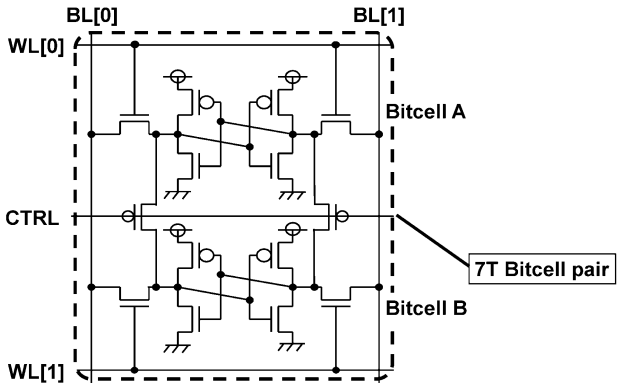


Figure 4. Schematic of 7T bitcell pair.

Figure 4 presents a schematic of a doubled 7T SRAM cell, which is based on a simultaneous copy scheme and instantaneous comparison mechanism [5]. The 7T bitcells mutually connect their internal nodes using pMOS transistors. The area overhead of 7T bitcell is greater than that of the conventional 6T bitcell by 11%. When one bit of data is stored in the two bitcells, the 7T SRAM can improve the reliability because write margin and read margin of SRAM are improved.

B. Instantaneous comparison

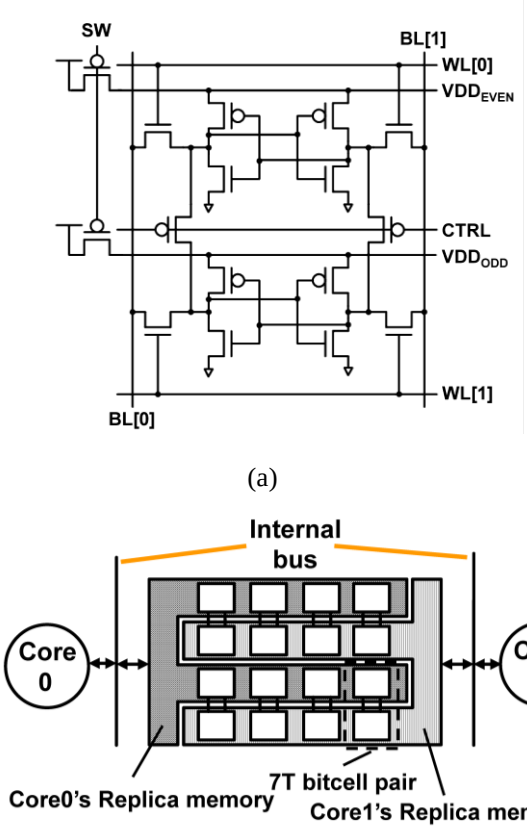


Figure 5. Instantaneous comparison structure: (a) instantaneous comparison feasible 7T bitcell pair and (b) deployment instaneous comparison buffer.

Figure 5(a) shows a schematic of the 7T SRAM, which can achieve high-speed comparison. In comparing data, the connecting pMOSes are turned on by lowering the CTRL signal [6]. If a 7T bitcell pair retains different data, then supply current flows into a ground. However, if it retains the same data, then the supply current does not draw through a bitcell pair because no current path exists. Therefore, results of the comparison depend on whether the supply voltage is high or not. Moreover, a comparison is made instantly by deploying a replica of the register and queue as shown in Figure 5(b). It takes four cycles to produce an instantaneous comparison independent of the size of the comparison buffer. The cycle time by normal comparison via the shared bus becomes longer as the number of registers increases. Although comparison with the CRC can accomplish comparison in two cycles, the proposed comparison scheme has higher fault coverage than comparison with the CRC. Furthermore, the instantaneous comparison scheme consumes 99.79% less energy than comparison with a CRC comparison circuit [6]. However, drawback of this comparison scheme is that area overhead increases accordingly size of the instantaneous comparison buffer.

C. Simultaneous copy scheme

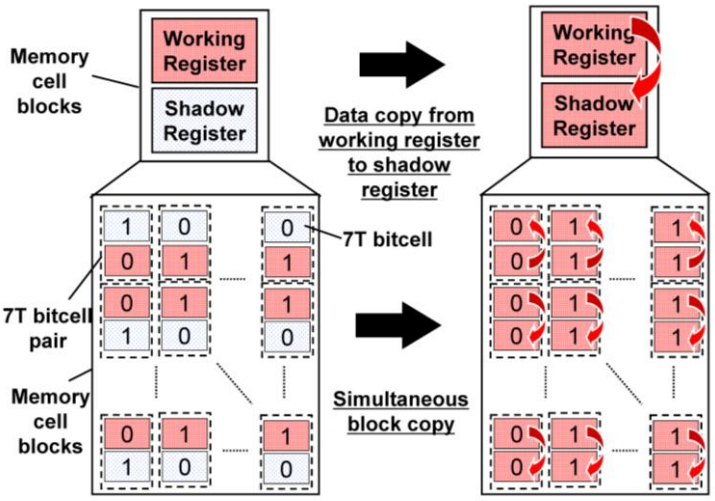


Figure 6. Simultaneous copy scheme.

The simultaneous copy scheme is extremely effective for the reduction of latency for the checkpoint and recovery state. Figure 6 depicts the working register value back up into the shadow register at the block-level using the simultaneous copy scheme. Figure 7 depicts a schematic of the simultaneous copiable 7T SRAM that performs data copying between the bitcells [7]. The 7T bitcells connect their internal node mutually with pMOS transistors in the same way as that for 7T SRAM, as explained briefly in section 3.B. For the case in which a datum in bitcell A is copied to bitcell B (upper to lower), first, the datum in bitcell B is destroyed and prepared for copying. Then, the datum in bitcell A is copied to bitcell B through connecting pMOSes. All 7T bitcell pairs can simultaneously back up the checkpoint data or restore them without a shared bus. Consequently, this 7T SRAM can produce a high-speed copy that requires only four cycles.

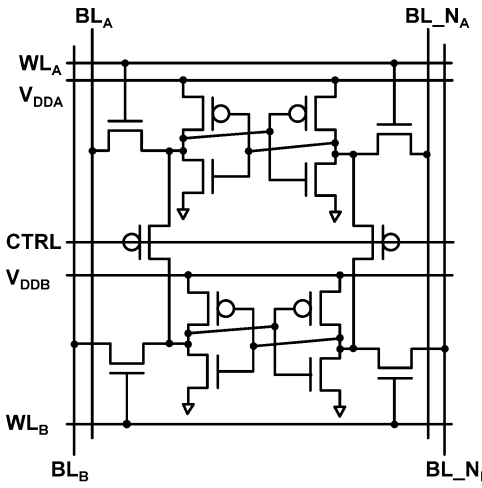


Figure 7. Schematic of simultaneous copiable SRAM.

IV. RESULT

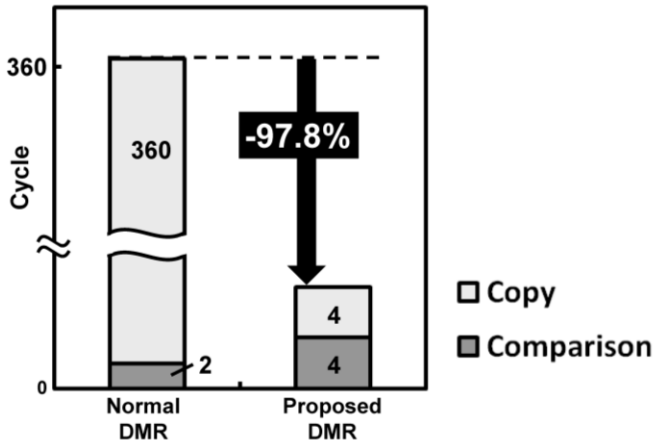
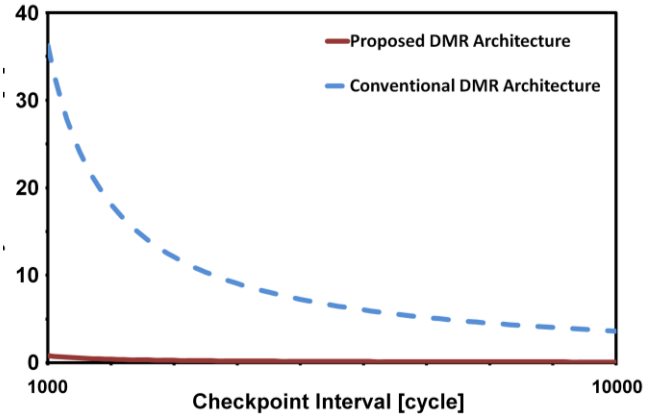


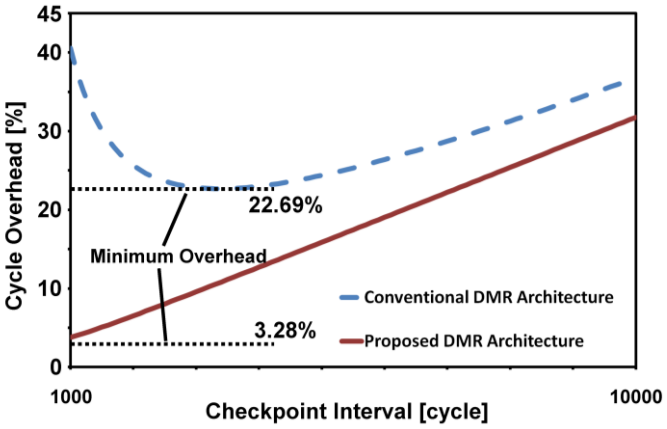
Figure 8. Measured copy cycle, where the number of working registers is 360.

The evaluated cycle time in the DMR phase is presented in Figure 8. Assuming the number of working registers as 360, we can estimate the cycle time in the DMR phase. The conventional DMR produces a comparison using CRC. Thereby it takes two cycles to compare CRCs. However, instantaneous comparison requires four cycles. Although copying between the registers via a shared bus requires 360 cycles, it takes four cycles to copy using a simultaneous copy scheme with 7T SRAM: the conventional DMR architecture needs 362 cycles in the DMR phase. In contrast, the proposed architecture requires eight cycles for comparison and copy and reduces clock cycles by 97.8% compared to conventional copying via the shared bus.

As shown in Figures 9(a) and 9(b), the y-axis shows the cycle overhead with the checkpoint/restart approach. The x-axis shows the checkpoint interval, which is the number of cycles between two consecutive checkpoints. Assuming that the execution time of the task is 50 μs, one cycle time is 16 ns. Figure 9(a) represents the cycle overhead of execution of a task that is fault-free. Figure 9(b) shows that a fault occurs in executing the task once. In Figure 9(a), as the checkpoint interval decreases, the cycle overhead increases rapidly because comparison and copy are performed frequently. In the proposed architecture, cycle overhead increases slightly when the checkpoint period is short because the proposed architecture dramatically reduces the cycle time in copy operation. When the checkpoint interval increases, both the cycle overheads of conventional and proposed architectures are low. However, if a failure occurs in the working register and the results of comparison are mismatched, then the DMR architecture must make a rollback checkpoint period. Figure 9(b) shows that the cycle overhead increases as the checkpoint interval becomes longer because the reexecution time is proportional to the checkpoint interval. The minimum cycle overhead of the conventional DMR and proposed DMR are 22.6% and 3.28%, respectively, in this case. Moreover, the difference of cycle overhead increases in an environment in which faults often occur. Consequently, the proposed architecture can maintain low latency by setting the checkpoint interval low.



(a)



(b)

Figure 9. Cycle overhead with respect to the checkpoint interval: (a) fault-free and (b) occurrence fault once.

V. CONCLUSION

We proposed a DMR architecture that can conduct instantaneous comparison with a simultaneous copy scheme using 7T SRAM. The proposed DMR architecture performs high-speed copying using a simultaneous copy scheme and high fault coverage using instantaneous comparison. The proposed architecture can execute operations with low latency even if the checkpoint interval becomes short. Consequently, the proposed DMR architecture provides benefits for real-time systems. When a user uses the proposed architecture, area overhead due to using 7T SRAM and instantaneous comparison buffer must be considered.

REFERENCES

[1] J. Teifel, "Self-voting dual-modular-redundancy circuits for single-event-transient mitigation," IEEE Transactions on Nuclear Science, vol.55, pp.3435–3439, 2008.

[2] A. Ziv and J. Bruck, "Performance Optimization of Check-pointing Schemes with Task Duplication," IEEE Transactions on Computers, pp. 1381–1386, 1997.

[3] Y. Zhang and K. Chakrabarty, "Energy-aware adaptive checkpointing in embedded real-time systems," DATE, vol.1, pp.918-923, 2003.

[4] J. C. Smolens, B. T. Gold, B. Falsafi, and J. C. Hoe, "Reunion: Complexity-Effective Multicore Redundancy," MICRO, pp. 223–234, 2006.

[5] H. Fujiwara, S. Okumura, Y. Iguchi, H. Noguchi, H. Kawaguchi, M. Yoshimoto, "A 7T/14T Dependable SRAM and its Array Structure to Avoid Half Selection," VLSI Design 2009, pp. 295–300, January, 2009.

[6] S. Okumura, Y. Nakata, K. Yanagida and Y. Kagiyama, "Low-power block-level instantaneous comparison 7T SRAM for dual modular redundancy," IEEE CICC, 2011.

[7] S. Okumura, S. Yoshimoto, K. Yamaguchi, Y. Nakata, H. Kawaguchi and M. Yoshimoto, "7T SRAM enabling low-energy simultaneous block copy," IEEE CICC, 2010.