IEICE TRANSACTIONS on Electronics

VOL. E98-C NO. 4
APRIL 2015

A PUBLICATION OF THE ELECTRONICS SOCIETY
The Institute of Electronics, Information and Communication Engineers
Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3chome, Minato-ku, TOKYO, 105-0011 JAPAN

PAPER *Special Section on Solid-State Circuit Design—Architecture, Circuit, Device and Design Methodology*

A Low-Latency DMR Architecture with Fast Checkpoint Recovery Scheme

Go MATSUKAWA[†a)], ***Student Member***, **Yohei NAKATA**[†], **Yasuo SUGURE**[††], ***Members***, **Shigeru OHO**[†††], ***Nonmember***, **Yuta KIMI**[†], ***Student Member***, **Masafumi SHIMOZAWA**[††††], **Shuhei YOSHIDA**[†], ***Nonmembers***, **Hiroshi KAWAGUCHI**[†], ***and*** **Masahiko YOSHIMOTO**[†,†††††], ***Members***

SUMMARY This paper presents a novel architecture for a fault-tolerant and dual modular redundancy (DMR) system using a checkpoint recovery approach. The architecture features exploitation of SRAM with simultaneous copy and instantaneous compare function. It can perform low-latency data copying between dual cores. Therefore, it can carry out fast backup and rollback. Furthermore, it can reduce the power consumption during data comparison process compared to the cyclic redundancy check (CRC). Evaluation results show that, compared with the conventional checkpoint/restart DMR, the proposed architecture reduces the cycle overhead by 97.8% and achieves a 3.28% low-latency execution cycle even if a one-time fault occurs when executing the task. The proposed architecture provides high reliability for systems with a real-time requirement.

key words: *dual modular redundancy, checkpoint recovery, fault-tolerance*

1. Introduction

Microprocessors are key components used for widely diverse applications. Processors used in safety-critical systems such as vehicles and social infrastructure must operate with extremely high reliability. Nevertheless, processors are increasingly sensitive to software errors, power supply noise, and temperature fluctuations with technology scaling. These factors engender faults in the processor. Consequently, reliable processors are being sought which can detect faults and recover from a faulty state even if faults in the processor occur.

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 enable fault detection 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 is inapplicable 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 capabilities of the comparison of CRC code depend on the fault location. The bus transfer is processed sequentially. Therefore, the bus transfer latency increases along with the number of registers.

Several time-redundancy methods have been proposed for fault tolerant systems [5]–[6]. Although the use of time-redundancy is superior to modular redundancy such as DMR and triple module redundancy (TMR) with respect to area overhead, time-redundancy methods incur a large cycle penalty. Consequently, it is difficult to apply time-redundancy methods to microprocessors for real-time systems. Actually, DMR approaches with a recovery scheme have also been proposed [7]. However, in a recovery phase, copying for rollback must be performed via a shared bus. The latency of copying increases proportionately with the number of registers.

We propose a DMR architecture that has a low latency recovery scheme. To realize low latency, the proposed DMR architecture exploits block-level simultaneously copiable SRAM. Moreover, exploiting block-level instantaneous comparison SRAM improves the fault-detection capability. A preliminary version of this report was published earlier [8]. Compared to that earlier work, this paper includes several more evaluations to demonstrate that the proposed architecture provides benefits for real-time systems and explanation about the control flow in the proposed DMR architecture in more detail.

2. Proposed DMR Architecture with a Recovery Scheme

This section presents an overview of the proposed DMR architecture with a recovery scheme. Then the checkpoint/restart scheme is explained. Finally, we explain the differences from conventional DMR. Figure 1 portrays the proposed DMR architecture, which consists of two cores,

Manuscript received July 21, 2014.
Manuscript revised January 7, 2015.

[†]The authors is with Kobe University, Kobe-shi, 657-8501 Japan.
[††]The author is with Central Research Laboratory, Hitachi, Ltd., Yokohama-shi, 105-0011 Japan.
[†††]The author is with Nippon Institute of Technology, Minamisaitama, 345-8501 Japan.
[††††]The author is with Hitachi Solutions, Ltd., Yokohama-shi, 244-0801 Japan. The author is also affiliated with Nippon Institute of Technology, Minamisaitama, 345-8501 Japan.
[†††††]The author is with JST CREST, Chiyoda-ku Tokyo, 102-0076 Japan.
a) E-mail: matsukawa@cs28.cs.kobe-u.ac.jp

DOI: 10.1587/transele.E98.C.333

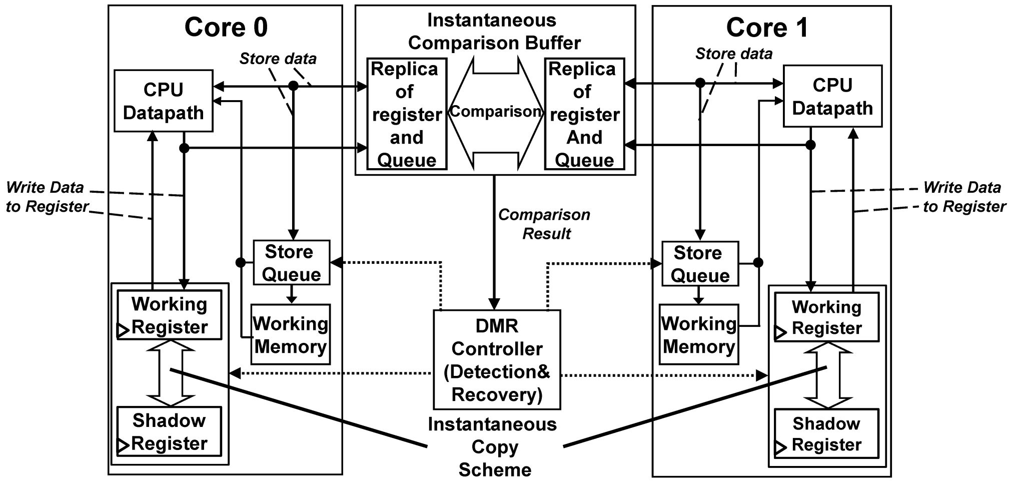

Core 0
Instantaneous Comparison Buffer
Core 1
Store data
CPU Datapath
Replica of register and Queue
Comparison
Replica of register And Queue
Write Data to Register
Comparison Result
Store Queue
Working Memory
Working Register
DMR Controller (Detection& Recovery)
Shadow Register
Instantaneous Copy Scheme

Fig. 1 Proposed DMR Architecture with recovery scheme.

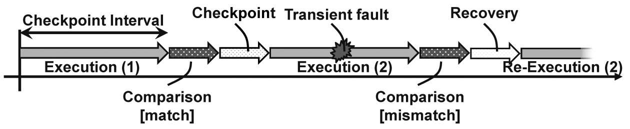


Fig. 2 Execution with checkpoint and recovery.

an instantaneous comparison buffer, and a DMR controller. The proposed architecture executes the normal operation using data stored in working memory, the store queue, and the working register. Then, store data and write data to the register are written to the store queue or the working register, and are written simultaneously in the instantaneous comparison buffer. As presented in Fig. 2, normal operations begin during checkpoint intervals. In the instantaneous comparison buffer, data of replicas of the register and queue are compared during a comparison period. Figure 3 depicts data transfer between the store queue and working memory. The store queue has a dirty bit for each block of the store queue. The dirty bit indicates whether the corresponding block of the store queue has been overwritten or not. If a datum required for a processing exists in the store queue and the corresponding dirty bit is “1,” then the datum in the store queue is used for the processing. If a datum required for a processing exists in the store queue and the corresponding dirty bit is “0,” then the datum in the work memory are used because new data should be used for processing. The comparison result is transferred to the DMR controller. If the result of the comparison indicates a match, then all the data in the store queue are written in working memory in the next normal operation (Fig. 3(a)). Data of the working register are copied in the shadow register using simultaneous data copy. The dirty bits remain unchanged. If the result of the comparison indicates a mismatch, then the store queue data are erased and the data of the shadow register are transferred to the working register (Fig. 3(b)). Dirty bits are set to “0” because the possibility exists that faults occur in the store queue. Then the process restarts from the latest checkpoint.

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

The proposed architecture is effective for a transient fault such as soft error and a read/write failure in a SRAM cell. However, a permanent fault cannot be covered because this architecture will repeat fault detection and recovery during execution if a permanent fault occurs. In some cases, a transient fault causes an infinite loop. For example, when a comparison result indicates a mismatch and a transient fault occurs at the flip-flop in the DMR controller, this architecture will perform an infinite loop. However, the probability that a transient fault occurs in DMR controller and the comparison result indicates that a mismatch is extremely low.

3. Instantaneous Comparison and Simultaneous Copy

The proposed DMR architecture exploits SRAM with simultaneous copy and compare functions. Herein, we explain the instantaneous comparison and the simultaneous copy scheme.

A. SRAM with Simultaneous Copy and Compare Function

Figure 4 presents a schematic of a pair of 6T bitcells, which realize a simultaneous copy scheme and the instantaneous comparison function [11]. The pair of 6T bitcells mutually connects their internal nodes using pMOS transistors. The area overhead of the bitcell is greater than that of the conventional 6T bitcell by 11%.

B. Instantaneous Comparison

The instantaneous comparison can be explained using

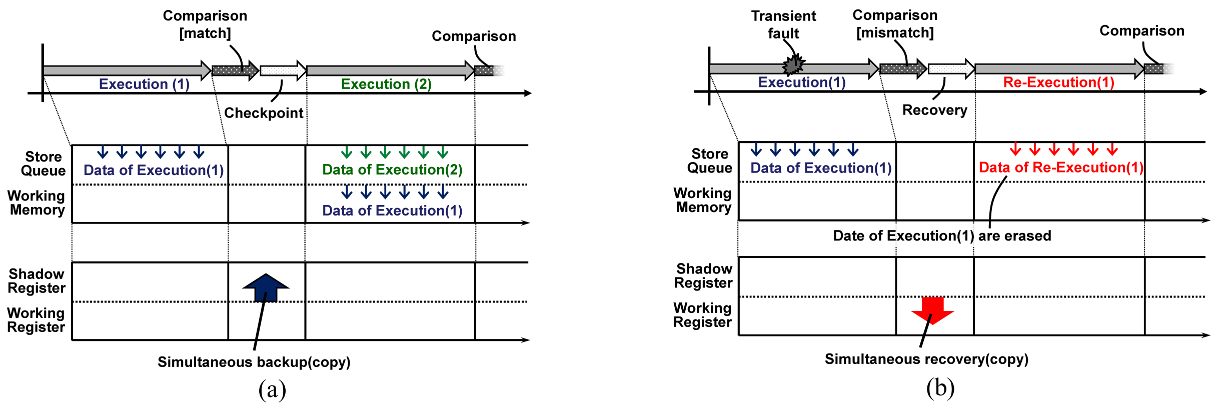


Fig. 3 Date between Store Queue and Working Memory: (a) comparison match and (b) comparison mismatch.

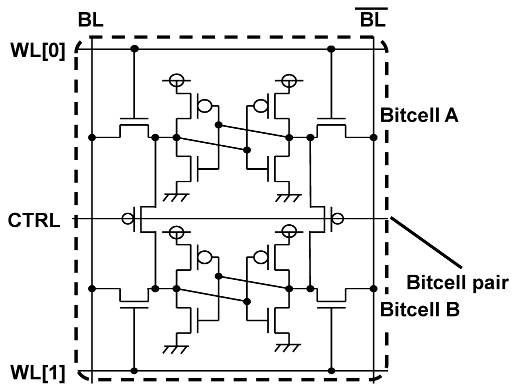


Fig. 4 Schematic of SRAM.

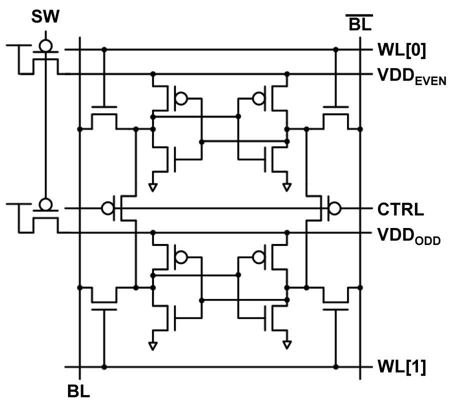


Fig. 5 Instantaneous comparison feasible 6T bitcell pair.

Fig. 5. In comparing data, the connecting pMOSes are turned on by lowering the CTRL signal [9]. If a 6T bitcell pair retains different data, then the 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. A comparison is made instantly among all 6T bitcells pairs. The instant comparison is presented in [9]. It takes four cycles to realize an instantaneous comparison independent of the size of the comparison buffer. The cycle overhead by normal comparison via the shared bus lengthens as the number of registers increases. Although comparison with the CRC can accomplish comparison in two cycles, the instantaneous comparison scheme consumes 99.79% less energy than comparison with a CRC comparison circuit because no complex calculation is required for the instantaneous comparison [10]. However, the shortcoming of this comparison scheme is that area overhead increases according to the size of the instantaneous comparison buffer.

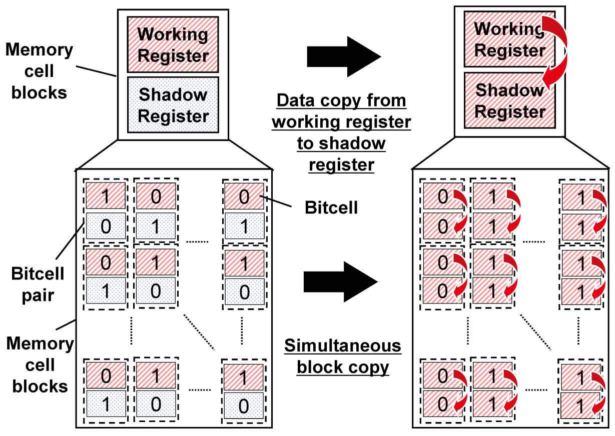


Fig. 6 Simultaneous copy scheme.

C. 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 simultaneous copy scheme between the working register and a shadow register. The process of copy between 6T bitcell pair is explained in [10] in detail. The checkpoint data can be backed up or restored simultaneously using no shared bus, but all 6T bitcell pairs. Consequently, the proposed SRAM structure requires only four cycles irrespective of the size of registers for the simultaneous block copy.

4. Simulation Results

4.1 Cycle Penalty Attributable to the Checkpoint Recovery Approach

The evaluated cycle overhead in the DMR phase is presented in Fig. 7. The DMR phase is defined as the period of comparison and copy between registers for the checkpoint recovery approach. Automotive embedded processors, used for a real-time system such as a engine control and a airbag system, are designed to integrate register banks. Assuming that the processor have 15 banks [14], in each of which the number of registers is 24, the number of registers is set to 360 here. 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 an SRAM with simultaneous copy function: the conventional DMR architecture requires 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.

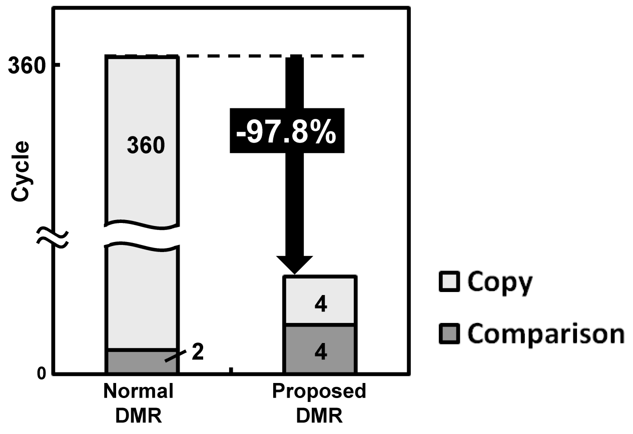
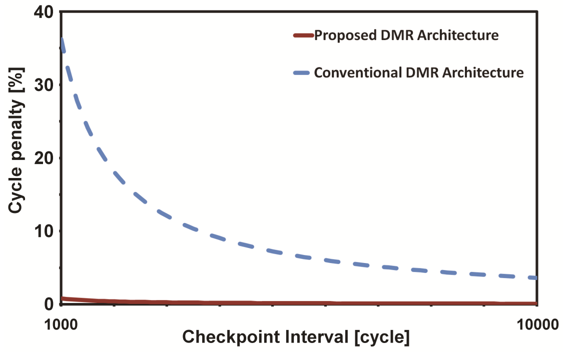
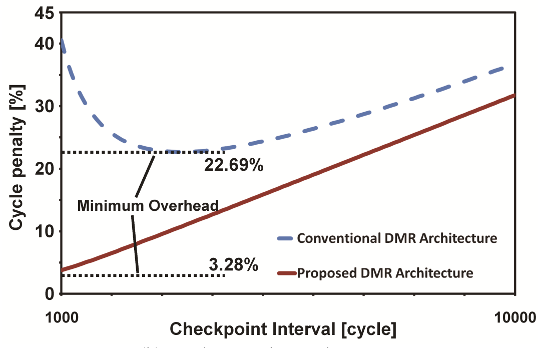


Fig. 7 Measured copy cycle, where the number of working registers is 360.



(a) when no mismatch occurs.



(b) when a mismatch occurs.

Fig. 8 Cycle penalty with respect to the checkpoint interval.

In Fig. 8(a) and 8(b), the y-axis shows the cycle penalty with the checkpoint recovery approach; the x-axis shows the checkpoint interval, which is the number of cycles between two consecutive checkpoints. A collision detection for airbag system is performed every 1 ms [15], and pressure sensor system for the detection of side crashes transmits the pressure data every 500 μs [16]. Assuming that the execution time of the task is 500 μs, and that one cycle time is 16 ns because the operation frequency of ordinary automotive embedded processor is set to 62.5 MHz [13], then Fig. 8(a) presents the cycle penalty in the execution of a task that is fault-free. Figure 8(b) shows the case in which a fault occurs in executing the task once. In Fig. 8 (a), as the checkpoint interval decreases, the cycle penalty increases rapidly because comparison and copy are performed frequently. In the proposed architecture, the cycle penalty increases slightly when the checkpoint period is short because the proposed architecture dramatically reduces the cycle overhead in the copy operation. When the checkpoint interval increases, both the cycle penalties of conventional and proposed architectures are low. However, if a fault occurs in the working register and the comparison result indicates a mismatch, then the DMR controller must control cores so that a rollback is performed. Figure 8(b) shows that the cycle penalty increases as the checkpoint interval lengthens because the reexecution time is proportional to the checkpoint interval. The minimum cycle penalties of the conventional DMR and the proposed DMR are 22.6% and 3.28%, respectively, in this case.

Figure 9 shows the cycle penalty when multiple mismatches occur. The number of mismatches stands for the number of times that a comparison result indicates a mis-

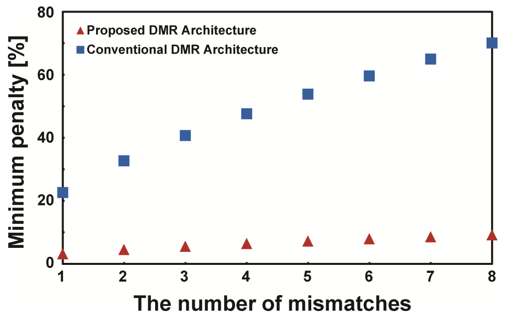


Fig. 9 Cycle penalty when multiple mismatches occur.

match while executing a task, which is equal to the number of times that a rollback is done. As the figure shows, the conventional DMR requires a larger cycle penalty under more frequent soft errors or stronger power supply noise. However, the proposed DMR can keep the minimum cycle penalty low, even if the rollback is performed numerous times. We assume that the checkpoint interval is set to the optimal checkpoint interval for each number of mismatches. Table 1 presents the optimal checkpoint intervals when the number of mismatches is varied. In the proposed DMR, the optimal checkpoint interval is much smaller than that of the conventional DMR. The cycle penalty is given by Eq. (1) as shown below.

$$\text{cycle penalty} = \left(\frac{T_{DMR}}{T_{checkpoint}} + N \times \frac{T_{DMR} + T_{checkpoint}}{T_{task}} \right) \times 100 \tag{1}$$

Therein, T_{DMR} stands for a cycle count overhead in the DMR phase, $T_{checkpoint}$ signifies a cycle count of the checkpoint interval, T_{task} denotes an execution cycle count of the task with fault free, and N represents the number of mismatches. If the checkpoint interval of the proposed DMR is set as 500 cycles, then the cycle penalty is 14.6% when a mismatch occurs eight times. However, in the conventional DMR, the respective optimal checkpoint intervals differ remarkably from one another, so it is difficult to get closer to a minimum cycle penalty. For example, if the checkpoint interval is set to be 3363 cycles, which is the optimal interval when a mismatch occurs, then the cycle penalty is 106% when a mismatch occurs eight times. Consequently, the proposed architecture is suitable for real-time systems because it can keep the cycle penalty lower.

4.2 Success Probability

We evaluate the success probability versus the fault rate, which is the probability with which a transient fault occurs. The success probability indicates the probability of task completion within its deadline time. We used the simulation method with a Markov model and Poisson process presented in an earlier report of the literature [12]. This method requires some parameters. Letting E be the execution time, and letting N be the number of checkpoints, then the relation among N, E and $T_{checkpoint}$ is written as follows.

$$\mathrm{N} = \frac{E}{T_{Checkpoint}} \tag{2}$$

Letting D be the deadline time of the task and letting t_{DMR} be the checkpoint overhead time required for the DMR phase, then, in this simulation, the execution time of the task and deadline time of the task are $E = 1$ and $D = 2$, respectively, for the number of checkpoints $N = 10$. The checkpoint overhead in the conventional architecture is $t_{\mathrm{DMR}} = 0.0116$; that in the proposed architecture is $t_{\mathrm{DMR}} = 0.000128$. Figure 10 shows the success probability versus the fault rate. In the conventional architecture, the fault rate increase lowers the probability of success. However, in the proposed architecture, the success probability decreases only slightly. Consequently, this simulation shows that the success probability is improved considerably using the proposed architecture.

Table 1 Optimal checkpoint interval that minimizes the cycle penalty.

♯ of mismatches	Optimal Checkpoint interval [cycle]	
	Proposed DMR	Conventional DMR
1	500	3363
2	354	2378
3	289	1941
4	250	1682
5	224	1504
6	204	1373
7	189	1271
8	176	1189

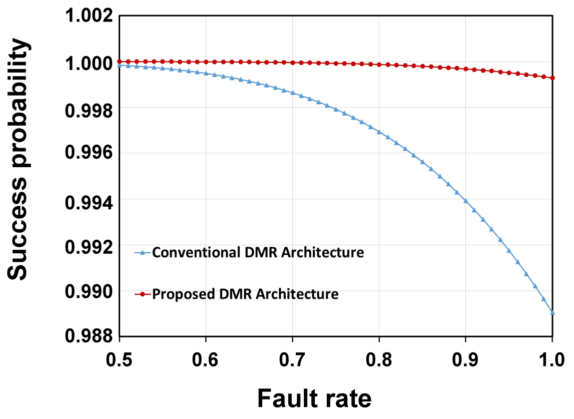


Fig. 10 Success probability with respect to fault rate.

5. Conclusion

We proposed a DMR architecture that can conduct an instantaneous comparison and a simultaneous copy scheme using the novel SRAM cell configuration. The proposed DMR architecture performs high-speed copying using a simultaneous copy scheme and low power comparison using an 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.

Acknowledgement

The authors thank to Mr. Takashi Yasumasu with Renesas Corporation for his valuable discussion.

References

[1] J. Teifel, “Self-voting dual-modular-redundancy circuits for single-event-transient mitigation,” IEEE Trans. Nucl. Sci., vol.55, no.6, pp.3435–3439, 2008.

[2] A. Ziv and J. Bruck, “Performance optimization of check-pointing schemes with task duplication,” IEEE Trans. Comput., vol.46, no.12, pp.1381–1386, 1997.

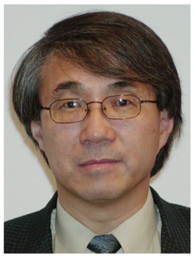
[3] Y. Zhang and K. Chakrabarty, “Energy-aware adaptive check pointing in embedded real-time systems,” Proc. Design Automation and

Test in Europe (DATE), pp.918–923, 2003.
[4] J. C. Smolens, B. T. Gold, B. Falsafi, and J. C. Hoe, “Reunion: Complexity-effective multicore redundancy,” Proc. IEEE/ACM Int. Symp. Microarchitecture (MICRO), pp. 223–234, 2006.
[5] W. J. Townsend, J. A. Abraham, and E. E. Swartzlander, “Quadruple time redundancy adders,” Proc. IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems (DFT), pp.250–256, 2003.
[6] J. Chen and Y. Yang, “A minimum proportional time redundancy based checkpoint selection strategy for dynamic verification of fixed-time constraints in grid workflow systems,” Proc. Asia-Pasific Software Engineering Conference (APSEC), pp.299–306, 2005.
[7] T. Sakata, T. Hirotsu, H. Yamada, and T. Kataoka, “A cost-effective dependable microcontroller architecture with instruction-level rollback for soft error recovery,” Proc. IEEE/IFIP Int. Conf. Dependable Systems and Networks (DSN), pp. 256–265, 2007.
[8] G. Matsukawa, Y. Nakata, Y. Kimi, Y. Sugure, M. Shimozawa, S. Oho, H. Kawaguchi, and M. Yoshimoto, “A low-latency DMR architecture with efficient recovering scheme exploiting simultaneously copiable SRAM,” Architecture of Computing Systems (ARCS), 2014.
[9] H. Fujiwara, S. Okumura, Y. Iguchi, H. Noguchi, H. Kawaguchi, and M. Yoshimoto, “A 7T/14T dependable SRAM and its array structure to avoid half selection,” Proc. Int. Conf. VLSI Design, pp. 295–300, 2009.
[10] S. Okumura, Y. Nakata, K. Yanagida, Y. Kagiyama, S. Yoshimoto, H. Kawaguchi, and M. Yoshimoto, “Low-power block-level instantaneous comparison 7T SRAM for dual modular redundancy,” Proc. IEEE Custom Integrated Circuits Conference (CICC), 2011.
[11] S. Okumura, S. Yoshimoto, K. Yamaguchi, Y. Nakata, H. Kawaguchi, and M. Yoshimoto, “7T SRAM enabling low-energy simultaneous block copy,” Proc. IEEE Custom Integrated Circuits Conference (CICC), 2010.
[12] J.-M. Yang and S. W. Kwak, “A checkpoint scheme with task duplication considering transient and permanent faults,” Proc. IEEE Int. Conf. on Industrial Engineering and Engineering Management (IEEM), pp.606–610, 2010.
[13] Renesas, http://am.renesas.com/applications/automotive/chassis/air_bag
[14] PLS, http://www.pls-mc.com/content/view/200/368/
[15] CITA, http://www.cita-vehicleinspection.org/Portals/cita/Documents/08%20Publications/01%20CITA%20studies/02%20Electronic%20Controlled%20systems%20-%202002/ECS-RSP%20Study%202%20-%20TP%20airbags.pdf
[16] Infineon, https://www.infineon.com/cms/en/about-infineon/press/press-releases/2012/INFATV201206–048.html

Go Matsukawa received a B.E degree in Computer and Systems Engineering from Kobe University, Hyogo, Japan in 2013. He is currently in the master course at Kobe University. His research interests include fault tolerant processor designs and multi-core processor architecture. He is a student member of IEICE.

Yohei Nakata received B.E. and M.E. degrees in Computer and Systems Engineering from Kobe University, Hyogo, Japan in 2008 and 2010, respectively, and received a Ph.D. degree in Electrical Engineering from Kobe University in 2013. His current research interests include dependable and variation-aware processor designs and multi-core processor architecture. Dr. Nakata is a recipient of the 2011 IPSJ Yamashita SIG Research Award. He is a member of IEEE.

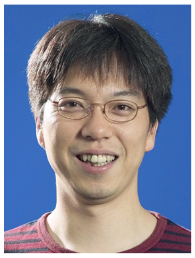
Yasuo Sugure received B.E. degree in electrical and electronic engineering and M.E. degree in electronics and information science from Chiba University, Japan, in 1997 and 1999, and Ph.D. degree from Tokyo Institute of Technology in 2012 respectively. He joined Central Research Laboratory, Hitachi, Ltd., in 1999. He was stationed in the Detroit area in the U.S.A. for Automotive Product Research Laboratory, Hitachi America, Ltd., from 2007 to 2008. He is currently a senior researcher at Central Research Laboratory, Hitachi, Ltd., where he has been engaged in the research and development of virtual prototyping system methods using a microcontroller model for embedded control systems. He is a member of IEICE, SAE.


Shigeru Oho graduated from Sasebo National College of Technology in 1976, received B.E. from University of Electro-Communications in 1978, and M.E. and PhD from Tokyo Institute of Technology in 1980 and 1997, respectively. He joined Hitachi, Ltd. in 1980 and then worked at Hitachi’s research laboratories both in Japan and U.S.A. He became a professor at Nippon Institute of Technology in 2012. He has been majoring in automotive electronics that encompasses electronic control, sensors and actuators, and model-based development technologies. He is a member of IEICE, SICE, JSAE, SAE and IEEE.

Yuta Kimi received a B.E degree in Computer and Systems Engineering from Kobe University, Hyogo, Japan in 2013. He is currently in the master course at Kobe University. His research interests include variation-tolerant processor designs and multi-core processor architecture.

Masafumi Shimozawa received M.E. degrees in Mathematics and Physics from Osaka City University, Osaka, Japan in 2004, and received a Ph.D. degree in Mathematics from Tokyo Woman's Christian University in 2009. He joined Hitachi Solutions, Ltd., in 2004. He is a member of MSJ.

Shuhei Yoshida received a B.E degree in Computer and Systems Engineering from Kobe University, Hyogo, Japan in 2014. He is currently in the master course at Kobe University. His research interests include variation-tolerant processor designs and multi-core processor architecture.



Hiroshi Kawaguchi received B.Eng. and M.Eng. degrees in electronic engineering from Chiba University, Chiba, Japan, in 1991 and 1993, respectively, and earned a Ph.D. degree in electronic engineering from The University of Tokyo, Tokyo, Japan, in 2006. He joined Konami Corporation, Kobe, Japan, in 1993, where he developed arcade entertainment systems. He moved to The Institute of Industrial Science, The University of Tokyo, as a Technical Associate in 1996, and was appointed as a Research Associate in 2003. In 2005, he moved to Kobe University, Kobe, Japan. Since 2007, he has been an Associate Professor with The Department of Information Science at that university. He is also a Collaborative Researcher with The Institute of Industrial Science, The University of Tokyo. His current research interests include low-voltage SRAM, RF circuits, and ubiquitous sensor networks. Dr. Kawaguchi was a recipient of the IEEE ISSCC 2004 Takuo Sugano Outstanding Paper Award and the IEEE Kansai Section 2006 Gold Award. He has served as a Design and Implementation of Signal Processing Systems (DISPS) Technical Committee Member for IEEE Signal Processing Society, as a Program Committee Member for IEEE Custom Integrated Circuits Conference (CICC) and IEEE Symposium on Low-Power and High-Speed Chips (COOL Chips), and as an Associate Editor of IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences and IPSJ Transactions on System LSI Design Methodology (TSLDM). He is a member of the IEEE, ACM, IEICE, and IPSJ.

Masahiko Yoshimoto joined the LSI Laboratory, Mitsubishi Electric Corporation, Itami, Japan, in 1977. From 1978 to 1983 he had been engaged in the design of NMOS and CMOS static RAM. Since 1984 he had been involved in the research and development of multimedia ULSI systems. He earned a Ph.D. degree in Electrical Engineering from Nagoya University, Nagoya, Japan in 1998. Since 2000, he had been a professor of Dept. of Electrical & Electronic System Engineering in Kanazawa University, Japan. Since 2004, he has been a professor of Dept. of Computer and Systems Engineering in Kobe University, Japan. His current activity is focused on the research and development of an ultra low power multimedia and ubiquitous media VLSI systems and a dependable SRAM circuit. He holds on 70 registered patents. He has served on the program committee of the IEEE International Solid State Circuit Conference from 1991 to 1993. Also he served as Guest Editor for special issues on Low-Power System LSI, IP and Related Technologies of IEICE Transactions in 2004. He was a chair of IEEE SSCS (Solid State Circuits Society) Kansai Chapter from 2009 to 2010. He is also a chair of The IEICE Electronics Society Technical Committee on Integrated Circuits and Devices from 2011–2012. He received the R&D100 awards from the R&D magazine for the development of the DISP and the development of the realtime MPEG2 video encoder chipset in 1990 and 1996, respectively. He also received 21th TELECOM System Technology Award in 2006.