DARA: A Low-Cost Reliable Architecture Based on Unhardened Devices and Its Case Study of Radiation Stress Test

Jun Yao, *Member, IEEE*, Shogo Okada, Masaki Masuda, Kazutoshi Kobayashi, *Member, IEEE*, and Yasuhiko Nakashima, *Member, IEEE*

***Abstract*—A microprocessor with an architectural redundancy to achieve high dependability is designed and manufactured to explore the effectiveness of tolerating soft errors without circuit hardening. The processor architecture is based on a modularized pipeline which contains several functionalities to facilitate a real-time error detection and a fast roll-back recovery. As a further extension for a possible increase of hard errors in the future technology, an energy-effective coverage of hard errors by dynamically adapting the redundancy between a dual and a triple module is also included in the processor. A radiation stress test result indicates that the designed redundant but unhardened processor can successfully achieve the same dependability as a hardened processor. Our synthesis and layout results show that radiation hardened circuits increase processor hardware area by 71% and power by 28%, respectively. It is thus possible to use the architectural redundancy instead of circuit hardening to achieve a cost-effective reliability, as suggested by these factors.**

***Index Terms*—Fault tolerance, radiation hardening, redundancy, .**

I. Introduction

NOWADAYS, with the continuous decreasing of switching voltages and feature sizes of the semiconductor transistors, the susceptibility to radiation Single Event Effects (SEEs) caused by the interaction with the natural space environment, including Cosmic Rays, Solar Energetic Particles and trapped protons in the Van Allen Belts will increase and become an increasingly severe problem [1]–[4]. Schemes specially designed for mitigating SEE and tolerating permanent faults, including radiation hardened circuits, either by process or design, radiation hardening approaches such as Triple Modular Redundancy (TMR), and other design/architecture approaches such as watch-dog timers with roll back and recovery, are thereby required to keep processors advancing along with the continuous scaling of process technology.

This paper provides a detailed discussion of a modularized system design with multiple redundant sub-system approaches to address SEE affects with only using unhardened cells. We then try a series of radiation stress tests on the manufactured chip to study its performance compared to rad-hard circuits in surviving an environment with extremely accelerated SEE injections. Our proposed functionalities in the chip include a stage-level data-bypassing interface to facilitate the comparison of sensitive data between pipelines, a well-tuned instruction decomposition to ensure the atomic update in commercial instruction set architectures (ISAs), and a fast roll-back recovery scheme. A DMR/TMR (dual/triple modular redundancy) adaptive processor architecture is then implemented by simply scaling the designed pipeline module. The adaptive redundancy supports a fast recovery DMR approach to address SEE. To tolerate possible permanent defects whose rate also increases with the shrinking of the transistor feature size, TMR is dynamically activated by an in-chip hardware controller to locate the permanently defective unit in case of a frequent error detection. After isolating the pipeline module with the permanent defect, the TMR structure can fall back to the DMR one for the purpose of an energy-effective soft/hard fault coverage. Compared to an always TMR execution, the adaptive redundancy has better power consumption. In addition, the approach to replace one defective DMR pipeline each time can be expected to have a longer lifespan under a given resource pool, as compared to an always TMR approach.

Usually, rad-hard circuits are commonly considered to guarantee a solid coverage for the relatively high fault rate under nuclear and space environments. With the manufactured chip, the radiation stress test result has shown that the designed redundant but unhardened processor can successfully achieve the same dependability as a hardened processor. However, the rad-hard circuits usually provides the high dependability at a cost of area and power consumption. As an example, the BCDMR Flip-Flops [4] that we used in the hardened processor embedded a redundant circuit and is thus $3.34\times$ in area and $3.17\times$ in power consumption as compared to a normal FF. A system-level comparison indicates that the rad-hard circuit design has a larger processor hardware area by 71% and a higher power by 28%

Manuscript received July 13, 2012; revised September 19, 2012, September 30, 2012; accepted October 01, 2012. Date of current version December 11, 2012. This work was supported in part by the VLSI Design and Education Center (VDEC), University of Tokyo with the collaboration of Synopsys and Cadence Corporations, by JST CREST, JST ALCA, JST A-STEP (FS), and in part by a Grant-in-Aid for Young Scientists (B) No. 23700060.

J. Yao and Y. Nakashima are with the Graduate School of Information Science, Nara Institute of Science and Technology, Nara 630-0192 Japan (e-mail: yaojun@is.naist.jp; nakashim@is.naist.jp).

S. Okada and M. Masuda are with the Graduate School of Science and Technology, Kyoto Institute of Technology, Kyoto 606-8585, Japan (e-mail: sokada@vlsi.es.kit.ac.jp; mmasuda@vlsi.es.kit.ac.jp).

K. Kobayashi is with the Graduate School of Science and Technology, Kyoto Institute of Technology, Kyoto, Japan and also with CREST, Japan Science and Technology Agency (JST), Tokyo 102-0076 Japan (e-mail: kazutoshi.kobayashi@kit.ac.jp).

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

Digital Object Identifier 10.1109/TNS.2012.2223715

than the unhardened one. These findings suggest that it is possible to use the existing redundancy instead of circuit hardening to achieve a cost-effective reliability. In summary, this paper will present the following contributions:

1) A DMR/TMR adaptive redundancy architecture with a negligible recovery cost is proposed for an effective soft and hard fault toleration.
2) A chip is manufactured for the purpose of a thorough radiation stress test by including both unhardened and hardened circuits. Tests under different voltages and temperatures are first conducted on a processor level. The results show that an unhardened circuit with architectural redundancy can be used as an effective transient error toleration.

II. Dynamic Adaptive Redundant Architecture

The approach in this paper addresses the mitigation of SEE in satellite on-board processing systems for the natural space radiation environment. Spatial redundancy [5]–[8] is used because of its better coverage for both transient and permanent faults than the time redundancy [9]–[11].

Generally, a Triple Modular Redundancy (TMR) [12] architecture is preferred for its seamless coverage for all transient and permanent faults. However, a traditional fixed connection between the three identical modules in TMR presents little flexibility. Its seamless recovery capability will be unsustainable after a permanent fault and will thus require a new set of TMR to continue the recovery function, despite that 2/3 of the original logic may still work properly. In addition, the triple redundancy is usually an over-design under the assumption that transient faults are still more common cases than permanent ones. A possible solution to these problems can be a flexible connection and an adaptive spatial redundancy based on proper reconfiguration. For these reasons, we give a scalable pipeline architecture with features specially designed for data comparison and state restore. Our dependable processor with a dynamic adaptive redundancy architecture (DARA) is then constructed by scaling this pipeline architecture into a dual or triple modular redundancy form to perform dependable executions.

A. Scalable Pipeline Module Design With Dependability

The scalable pipeline design to construct dependable processors is given in Fig. 1, which contains six traditional pipeline stages as instruction fetch (IF), instruction decode (ID), register read (RR), execution (EX), memory access (MA), and write-back (WB).

Different from the implementation in papers [5]–[7] which are also fault-tolerant designs based on spatial redundancy, the error detection unit is embedded to perform data checks at a stage boundary. The distributed comparators inside every stage can help achieve an early and thorough error detection. With a fast recovery scheme, it is possible to introduce a minimal performance impact even under a relatively high error rate.

As shown in Fig. 1, input/output unidirectional links are included in the pipeline module, which are used for dependable data bypassing. A spatially redundant processor can be constructed by connecting multiple pipeline modules where copies of a single thread are simultaneously executed and compared.

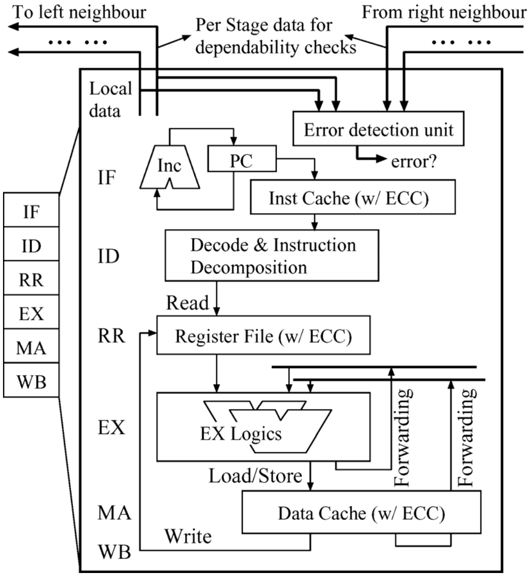

Fig. 1. A scalable pipeline module with dependable check.

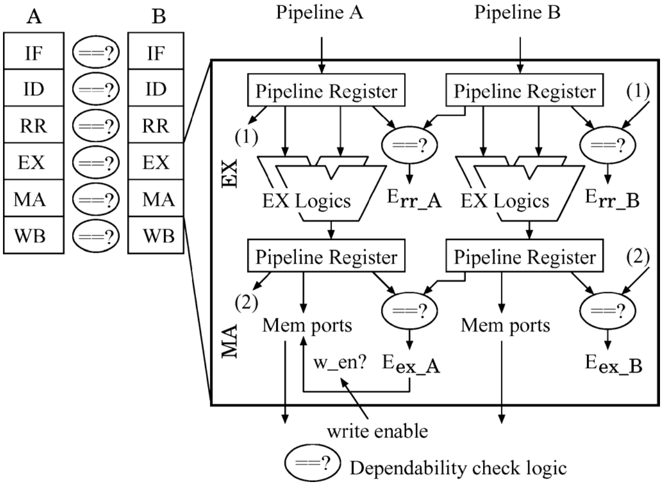

Fig. 2. DMR execution scheme in EX and MA stages.

The storage units including register files, memory units (instruction and data caches) are designed to be covered by Error Correcting Codes (ECC) [13] logic to guarantee reliable data storage. Data stored into the memory structures are regarded to be safe if they were checked before the store operation.

However, ECC is not applied to pipeline registers though their regular forms are suitable for using ECC to tolerate faults. This is because the delays of pipeline registers are usually the key factor to determine the critical path and hence adding ECC logic will increase latency. Besides the frequency issue, adding ECC encoder/decoder logic for each pipeline register will also lead to a large hardware increase which cannot be negligible. For all these reasons, we use the architectural redundancy to tolerate faults for the pipeline registers, as well as the combinational logic which do not have regular forms for ECC or parity.

B. DMR Based Fault-Tolerance

The spatial redundancy in DARA is based on a pipeline granularity. From the system view, a reliable computation is guaran-

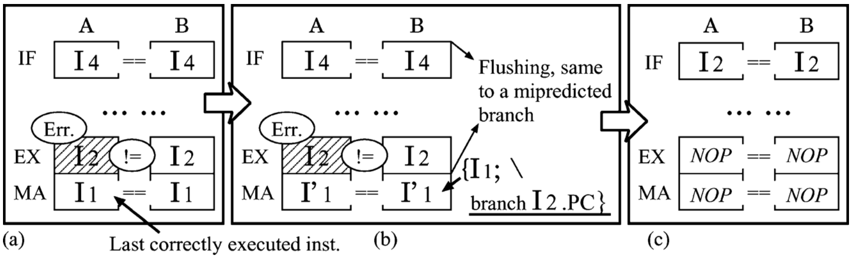

Fig. 3. Restarting execution by inserting a branch style instruction. (a) Error detected, (b) Preparation of recovery, (c) Re-execution starts, in next cycle.

teed by issuing two identical threads of a program in the framework, given in Fig. 2. By using the two identical pipelines with connected I/O ports, it is possible to achieve a synchronized multi-thread execution which facilitates data checks. The instruction replication starts at the fetch by using the same starting program counter (PC) in each thread. Simultaneously, the two threads will request the same blocks in both instruction and data spaces from the memory, i.e., the I cache (I$) and D cache (D$) misses and accordingly the cache contents in each pipeline are also identical. By assuming that I/O ports are driven by a higher voltage (3.3 V) than the in-core gates (1.8 V or lower), we can regard requests, acknowledgements and data between caches and off-chip memories are far less vulnerable to soft errors than the in-chip units. Specifically, our experiments show that flip-flops supplied by 1.8 V will have a 3% sensitivity to SEE as compared to 1.25 V, while our alpha source is unable to inject SEE under the supply voltage of 3.3 V. In addition, these I/O buses can easily apply ECC logic or it is possible to use TMR approaches to improve the dependability. For these reasons, we mainly focus on the in-chip lock-step-like duplicated execution for dependability in this research.

Although this DMR architecture is very similar to some traditional lock-step based fault-tolerant microprocessors such as IBM's S/390 G5 [14], the lock-step mechanism of DARA is designed not to increase the processor working latency. As Fig. 2 illustrates, the dependability check logic is employed after each pipeline register, which contains the result of the last stage, generated in the previous cycle. For example, in EX stage, the correctness checks, which give $E_{\rm rr\_A}$ and $E_{\rm rr\_B}$, are done in parallel to the normal processing in EX logic. As the dependability check logic does not prolong the delay of the EX stage, the clock frequency can remain uninfluenced after adding reliability features.

1) Roll-Back Based Recovery Procedure: Our DMR architecture uses an error-detecting and roll-back scheme instead of error correction to tolerate faults. The error reported by the detection logic indicates one of the two pipelines has experienced an erroneous state. When the fault is caused by SEE, a roll-back can fix the error without the necessity to identify which pipeline is actually giving the error. Many current dependable architectures [5], [6], [14] rely on a coarse-grain checkpoint to recover. They require additional hardened storage units to cache the processor running status including contents of register file, system control registers, and memory updates. Other than the hardware extension, the penalty from the long recovery procedure will not be negligible when the recoveries are frequently triggered under a high error rate environment. Differently, DARA is embedded with a fine-grained fast recovery scheme by making full use of the existing redundant information inside the dual-pipeline architecture. It is expected to take a far shorter time to restore to the previous correct state than coarse-grained checkpoint-based recoveries in [5], [6], [14]. The basic idea of recovery is given in Fig. 3.

An example of two consecutive instructions as $\rm I_1$ and $\rm I_2$ is shown in Fig. 3. Assume that at cycle n, the comparators report that one of the executions of $\rm I_2$ is problematic while the executions of $\rm I_1$ are verified to be correct. The information of $\rm I_1$ can thus be used to instruct the recovery. As depicted in Fig. 3(b), $\rm I_1$ will be extended to compound with a dummy branch instruction, as branch $\rm I_2 \cdot PC$. $\rm I_2 \cdot PC$ is basically the successive one of $\rm I_1 \cdot PC$. Only when $\rm I_1$ is a taken branch, $\rm I_2 \cdot PC$ becomes the branch target in $\rm I_1$. Thus, we can successfully assemble this branch $\rm I_2 \cdot PC$ by making a full use of correctly executed checkpoint $\rm I_1$. The dummy branch here is not a real instruction but a notation of overwriting the PC register in IF stage with the PC of $\rm I_2$.

IF stage then rolls back to the PC of $\rm I_2$ in the next cycle, re-fetching the unfinished $\rm I_2$ from the cache and the re-execution will thereby start afterward, as shown in Fig. 3(c). Here, we assume that caches are covered by ECC so that a re-fetch can obtain the correct data and instructions.

This recovery scheme follows the same mechanism as a branch misprediction in a normal processor. From this view, it can be seamlessly extended to support an out-of-order processor, which may be more complicated than the processor architecture in Fig. 3. As the out-of-order processor still commits in the program order, there is always a final correctly executed instruction in the sequence. The recovery scheme in Fig. 3 can be similarly applied by using this instruction as the checkpoint.

2) Instruction Decomposition for Atomic Update: Generally, the roll-back based error recovery requires updating atomicity inside one instruction. However, current commercial ISAs usually can not guarantee this feature. As an example, in SH-2 [15], which is a RISC ISA, the instruction `LD Rn, @(Rm+)` performs two operations: a memory load as `Rn` ← `@(Rm)`, and an address update as `Rm++`. However, if the address update to `Rm` is successful but an error occurs during loading or updating loaded data into `Rn`, a correct recovery can not be achieved. Even when the processor can re-execute the memory load part, it will not return to the correct execution since the address `Rm` has already been modified. To solve this, we give an instruction decomposition method to help maintain one atomic operation per instruction. The decomposition is achieved in decode stage in our de-

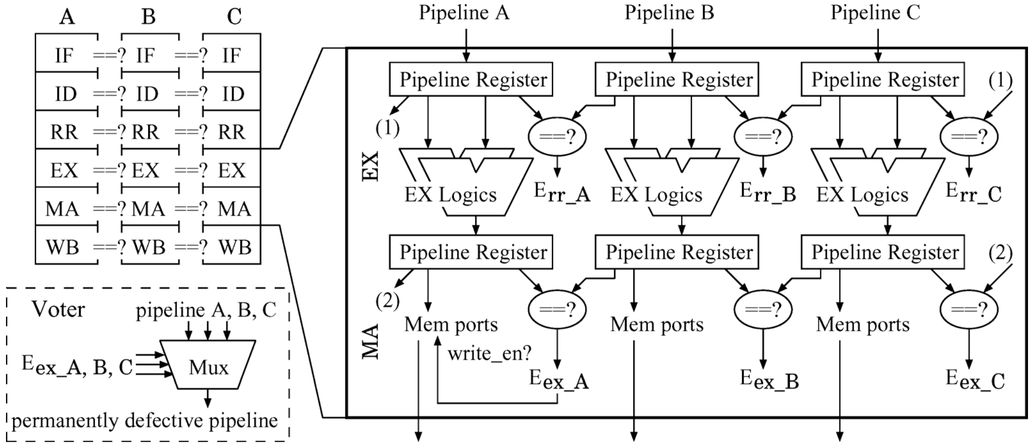

Fig. 4. TMR structure.

signed microprocessor without any special compiler aids. Rules for decomposition are as follows:

1) Keep address update after memory access to avoid the race competition between the two operations.
   a) `Rn` ← `@(Rm)`, //4-byte memory load;
   b) `Rm` ← `Rm +#4`, //#4: immediate value.
2) Use shadow registers for intermediate values.
   Shadow registers are necessary to pass intermediate values between decomposed instructions. Using the SH-2 instruction `RTE` as an example, it performs the following operations: (a) `LD PC, @(R15+)`; (b) `LD SR, @(R15+)`. It will be decomposed in the following way to make sure `R15` updates after the two loads, as:
   a) `TMP1` ← `R15`, //R15: stack pointer
   b) `TMP2` ← `R15 +#4`
   c) `SR` ← `@(TMP2)`, //SR: status register
   d) `R15` ← `TMP2 +#4`
   e) `PC` ← `@(TMP1)`, //see the third rule
3) Program Counter (PC) should be updated in the final sub-instruction, as shown in the above example.

After this, possible errors are isolated in each sub-instruction. By adding an additional `sub-PC` together with the PC in our recovery scheme, we are able to roll-back to the sub-instruction that previously experienced error. Note that it is not necessary to keep the sub-instruction in the instruction cache, as it can always be obtained by decomposing the original instruction. This ISA independent solution is expected to extend the applicability of DARA into architectures where ISA itself is not originally considered with dependability requirement.

C. TMR Mode

To additionally tolerate increasing threats from permanent faults, as discussed in paper [16], DARA will include a third pipeline module, which is originally prepared but disabled by power gating inside the processor into the DMR processor as shown in Fig. 4. With the enabling of the third pipeline under a very frequent error occurrence, a TMR core is formed to diagnose the defective units. After that, the formed TMR processor will be changed back to a DMR one by removing the erroneous pipeline. As the major working mode is still the DMR structure, the adaptive architecture can perform a power-effective execution while achieving a similar dependability as a fixed TMR structure. As the operating system (OS) may hang on an already defective DMR hardware, all the steps above are automatically performed by a hardened hardware controller to avoid the OS interference.

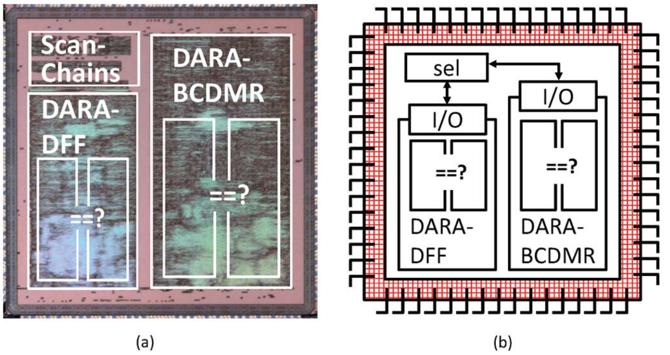

Fig. 5. A 0.18 μm chip with two DARA cores of radiation unhardened/hardened FFs. (a) Chip micrograph, (b) Outlined layout.

III. Radiation Stress Test

To verify the dependability, we have designed two different DARA cores with normal unhardened FFs and radiation hardened BCDMR FFs [4], respectively. Each DARA core follows the SH-2 instruction set architecture (ISA) [15]. These two cores are placed in the same chip and then synthesized by Synopsys Design Compiler with a 0.18 μm library. The chip size is 5 mm × 5 mm. Fig. 5 shows the chip micrograph and the outlined chip layout. DARA-DFF represents the DARA core with radiation unhardened FFs while in DARA-BCDMR, every FF is replaced by the radiation hardened one. They are exclusively enabled by a selector, which also provides a proper I/O route to the corresponding core.

As there is currently no practical method to inject hard (permanent) faults, only DMR constructions with the recovery scheme are included for a soft error stress test by radiation. Note that with the limitation of the chip pin number—which is 208 in this chip—I/O and related logic in each core do not have

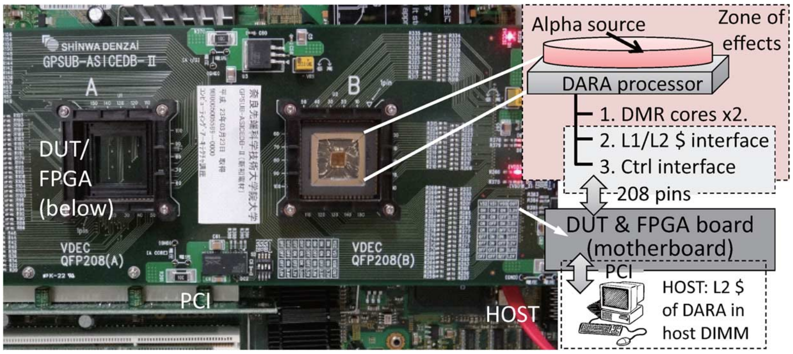

Fig. 6. DARA test platform.

TABLE I
AVERAGE ERRONEOUS FLIPS OF FFS UNDER 60-SECOND ALPHA PARTICLE IRRADIATION

Duration & voltage	Temper-ature (°C)	1,080 DFFs	1,080 BCDMRs	1,080 Delayed BCDMRs
60s, 1.25V	24.5	14.5	0	0
	71.3	13.5	0	0
60s, 1.8V	24.5	0.5	0	0
	71.3	1.2	0	0

any duplication. Delayed BCDMRs [4], which can mitigate SET pulses from combinational logic by a delay element in addition to an SEU, are used to cover these parts, including the selector as well.

A. Estimation of Fault Injection by Alpha Source

We use an alpha particle source, 3 MBq 241Am, to provide the fault injection in our radiation tests. Firstly, we give an approximate estimation of the fault injection rate by using an engineering tester to supply test vectors into the scan-chain part (the upper-left block in Fig. 5(a)) of the DARA chip. We have irradiated the scan-chains of DFFs, BCDMRs, and delayed BCDMRs, respectively. Two voltages 1.8 V and 1.25 V, together with the room temperature 24.5°C and a higher temperate 71.3°C, are applied during the test to investigate the possible impacts from both the supply voltage and the ambient temperature. All of the chains are triggered by a 1 MHz clock.

Table I shows the results of average erroneous flips of FFs during irradiations of 60 seconds. Both the BCDMRs and the delayed BCDMRs demonstrate no erroneous flips during the 60-second irradiations. The main impact on the errors in the DFF chain is the supply voltage while the temperature does not make a difference. From the data, the fault injection rate is about 2.24×10^{-4} per second per bit under 1.25 V. As DARA includes 2 614 unprotected DFFs, which are not covered by ECC, it is expected that these unprotected DFFs will be flipped at a rate of 0.58 FF/second by the irradiation. It is then used to perform the accelerated radiation stress test on the DARA chip.

B. Test Platform of DARA

Fig. 6 shows the platform of the DARA chip radiation test. The chip is connected to a host server with a DUT and an FPGA board, where the two boards can be regarded as a simplified motherboard in normal computer systems. Specially in this research, as shown in Fig. 6, to limit the fault injection inside the chip itself, the alpha source is directly mounted on the bare chip. The L2 cache contents, which are not covered by DARA, are physically stored in the DIMMs of the host server. The host server is used to handle both start/stop signals and L1 cache misses from the DARA chip. Both the DUT/FPGA and the host server are carefully calibrated to be outside of the penetration range of the alpha source. Therefore, only units within the DMR protection are exposed under the zone of fault effects.

C. DARA Irradiation Results

To give a stress test, the chip core supply voltage is lowered to the 1.25 V level, which has shown a higher sensitivity to SEE. The processor core is working under a 25 MHz frequency, which is at the boundary that all critical paths under the lowered supply voltage still meet the setup deadline.

We conducted two sets of experiments by using programs *Bubble, FFT, Intmm, Perm, Puzzle, Queens, Quick, Towers*, and *Trees* from the Stanford benchmark suite [17] as the real applications. We wrapped each benchmark with a loop of `for (i=0;i < N;i++) benchmark();` to have an execution time of 30–40 seconds, in order to accumulate a visible number of error injections. Each wrapped benchmark is run for 30 times to cover the deviations of alpha source soft fault injections.

We collected number of average recoveries in these executions, to serve as a measure of the errors detected by DARA. Similar to Table I, our results show that benchmarks on DARA-BCDMR have demonstrated 0 recoveries, which indicates that DARA-BCDMR can successfully provide an SEE-free execution. The tests on DARA-DFF show several recoveries during soft fault injection, as shown in Table II. Overall, across the 30-second executions, there will be around 10–15 errors detected by the comparison logic and recovered by the roll-back scheme in DARA-DFF.

Meanwhile, a thorough comparison between the execution results of radiation stress tests and the non-radiation tests shows that programs on both DARA-DFF and BCDMR give exactly same memory data access sequences and identical final memory results as the zero-fault injection runs. This specifically verifies that the unhardened DARA-DFF has mitigated SEEs and provided the same dependability as the rad-hard circuits.

The execution time differences between DARA-DFF and DARA-BCDMR in Table II represent the time required for recoveries. As the recovery in DARA will be started in a cycle

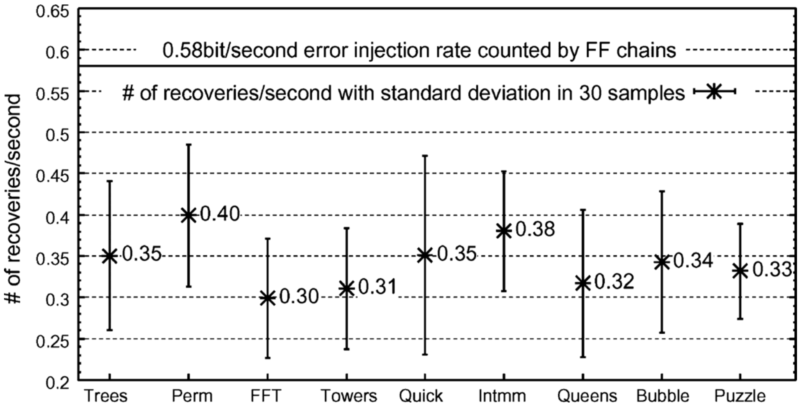

Fig. 7. Numbers of recoveries per second in DARA-DFF.

TABLE II
AVERAGED RESULTS OF RADIATION STRESS TEST ON DARA-DFF & DARA-BCDMR

Program	DARA-DFF		DARA-BCDMR	
	# of Recoveries	Time (second)	# of Recoveries	Time (second)
Trees	13.2	37.6	0	37.6
Perm	14.3	35.8	0	35.8
FFT	10.7	35.6	0	35.4
Towers	11.0	35.5	0	35.4
Quick	12.3	35.1	0	35.0
Intmm	13.0	34.3	0	34.3
Queens	10.7	33.7	0	33.4
Bubble	11.3	32.5	0	32.4
Puzzle	10.6	31.9	0	31.9

level, introduced in Section II.B.1, it is expected to have a negligible time cost. The data in Table II shows that the time differences are within 1%, which may be caused by errors of different runs.

In addition, the program characteristics, which is specifically known as architectural vulnerability factor (AVF) [18], also affects the number of recoveries in DARA-DFF. AVF measures the sensitivity to faults in different instructions and units. A program containing more low AVF instructions can be less vulnerable to SEEs. Our results in DARA-DFF give a good support for this statement that not every fault injection will come to require a recovery. The average errors per second, together with the standard deviations in the mean square error from the 30 execution samples of each benchmark, are shown in Fig. 7. According to the standard deviations, the uncertainties of the error detection rates are large among the 30 executions. This can be caused by the instability of the fault injection rate by our alpha source in the short irradiations, and the different AVFs of the different parts even inside each single benchmark. Overall, the average number of errors/time of each benchmark varies about 25% from the smallest one in FFT to the largest one in Perm, indicating a visible difference in the average AVFs among these benchmarks. Meanwhile, the largest error recovery rate with the deviation in benchmark Perm is still smaller than 0.58/sec from the FF chains where every upset is counted.

D. Exploration

From the view of mitigating SEE, a single pipeline based on BCDMR may be sufficient to address soft errors even under our accelerated fault injection. However, with the adaptive DMR/TMR switch, we have extended the architecture with the ability to tolerate permanent faults. By breaking the fixedly connected TMR structures, our method uses 2/3 of its working power to address SEE affects and provide for the spare and healthy module when a permanent fault occurs. Working under a large resource pool, this adaptivity can dynamically extend the processor lifespan.

TABLE III
BREAKDOWNS OF CHIP AREA

DARA Core	Synthesis [mm^2]		Layout (A_l) [mm^2]	A_s/A_l
	FF	Total (A_s)		
w/ DFF	1.41 (1.00)	3.28 (1.00)	5.47 (1.00)	60%
w/ BCDMR	3.60 (2.55)	5.61 (1.71)	9.38 (1.71)	60%

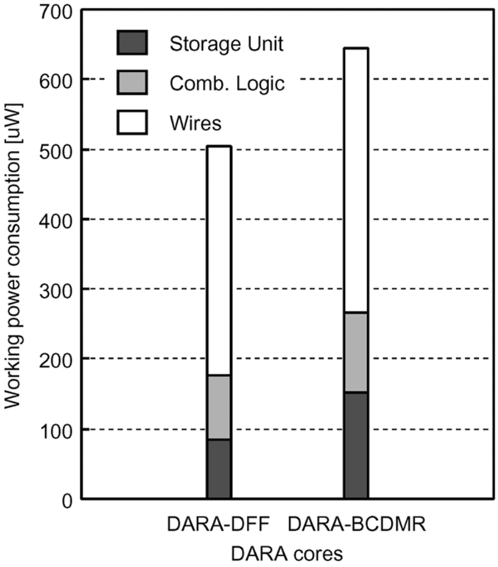

Fig. 8. Power consumption of DARA cores, estimated by logic-level simulation.

IV. AREA AND ENERGY RESULTS

Based on the observation that a similar dependability with unhardened devices is possible by simply using architectural duplication, this section gives the hardware cost and energy consumption results to study the efficiency of architectural duplication.

The difference between DARA-DFF and DARA-BCDMR is mainly at the DFF parts, either radiation unhardened or hardened. As BCDMR [4] contains more logic to provide a soft fault invulnerable circuit, it will take a larger hardware area than the normal DFF. As shown in Table III, the area of storage units in DARA-BCDMR is 255% of the one in DARA-DFF. Accordingly, the two cores will have different total areas, which has already been clearly shown in the chip photo. The area increase from DARA-DFF to DARA-BCDMR is 71%, as given in Table III.

Fig. 8 demonstrates the working power consumption results of the two DARA cores from a logic level simulation, which also indicates a similar changing tendency between the selection of DARA-DFF or DARA-BCDMR. The major difference is still in the storage unit parts. The hardened storage units in DARA-BCDMR consume 81% more power than DARA-DFF. However, as shown in Fig. 8, about 60% of power goes into wires. The total power consumption difference from DARA-DFF and DARA-BCDMR becomes smaller, which is 28%.

V. Conclusion

This paper introduced DARA, a dynamic adaptive redundancy architecture to tolerate both soft and hard errors by architectural redundancy. We specifically explored its ability of soft error tolerance by performing a radiation stress test on a chip in a 0.18 μm technology. From the scan-chain test result, the alpha particle source is able to flip unhardened DFFs at the rate of 0.58 bit/sec. Our radiation stress test results on the DARA chip have indicated that even under this high error injection rate, DARA based on unhardened FFs can have the same dependability as a processor with hardened circuits. Under the need for redundancy such as DMR and TMR to tolerate both soft and hard errors, the study of our radiation test shows that DARA can avoid the increases of 71% hardware cost and 28% power consumption introduced by radiation hardened FFs. By this means, DARA meets our purpose of an area-and-energy-efficient dependable architecture.

References

[1] X. Li, S. V. Adve, P. Bose, and J. A. Rivers, "Softarch: An architecture level tool for modeling and analyzing soft errors," in *Proc. Int. Conf. on Dependable Systems and Networks*, 2005, pp. 496–505.

[2] P. Shivakumar, M. Kistler, S. W. Keckler, D. Burger, and L. Alvisi, "Modeling the effect of technology trends on the soft error rate of combinational logic," in *Proc. Int. Conf. on Dependable Systems and Networks*, 2002, pp. 389–398.

[3] S. Mitra, N. Seifert, M. Zhang, Q. Shi, and K. S. Kim, "Robust system design with built-in soft-error resilience," *Computer*, vol. 38, no. 2, pp. 43–52, 2005.

[4] R. Yamamoto, C. Hamanaka, J. Furuta, K. Kobayashi, and H. Onodera, "An area-efficient 65 nm radiation-hard dual-modular flip-flop to avoid multiple cell upsets," *IEEE Trans. Nucl. Sci.*, vol. 58, no. 6, pp. 3053–3059, Dec. 2011.

[5] P. Meaney, S. Swaney, P. Sanda, and L. Spainhower, "IBM z990 soft error detection and recovery," *IEEE Trans. Device Mater. Rel.*, vol. 5, no. 3, pp. 419–427, Sep. 2005.

[6] S. K. Reinhardt and S. S. Mukherjee, "Transient fault detection via simultaneous multithreading," in *Proc. 27th Annu. Int. Symp. on Computer Architecture*, 2000, pp. 25–36.

[7] M. Gomaa, C. Scarbrough, T. N. Vijaykumar, and I. Pomeranz, "Transient-fault recovery for chip multiprocessors," in *Proc. 30th Annu. Int. Symp. on Computer Architecture*, 2003, pp. 98–109.

[8] J. C. Smolens, J. Kim, J. C. Hoe, and B. Falsafi, "Efficient resource sharing in concurrent error detecting superscalar micro-architectures," in *Proc. 37th Annu. Int. Symp. on Microarchitecture*, 2004, pp. 257–268.

[9] M. K. Qureshi, O. Mutlu, and Y. N. Patt, "Microarchitecture based introspection: A technique for transient-fault tolerance in microprocessors," in *Proc. Int. Conf. on Dependable Systems and Networks*, 2005, pp. 434–443.

[10] E. Rotenberg, "AR-SMT: A microarchitectural approach to fault tolerance in microprocessors," in *Proc. 29th Annu. Int. Symp. on Fault-Tolerant Computing*, 1999, pp. 84–91.

[11] N. Oh, P. Shirvani, and E. McCluskey, "Error detection by duplicated instructions in super-scalar processors," *IEEE Trans. Rel.*, vol. 51, no. 1, pp. 63–75, 2002.

[12] D. P. Siewiorek and R. S. Swarz, *Reliable Computer Systems (3rd ed.): Design and Evaluation*. London, U.K.: A. K. Peters, Ltd., 1998.

[13] C. L. Chen and M. Y. Hsiao, "Error-correcting codes for semiconductor memory applications: A state of the art review," *Reliable Computer Systems—Design and Evaluation*, pp. 771–786, 1992.

[14] T. J. Slegel, I. Averill, R. M. , M. A. Check, B. C. Giamei, B. W. Krumm, C. A. Krygowski, W. H. Li, J. S. Liptay, J. D. MacDougall, T. J. McPherson, J. A. Navarro, E. M. Schwarz, K. Shum, and C. F. Webb, "IBM's S/390 G5 microprocessor design," *IEEE Micro*, vol. 19, no. 2, pp. 12–23, Mar./Apr. 1999.

[15] Renesas Technology, SH-1/SH-2/SH-DSP Software Manual Rev. 5.00 2004.

[16] J. Srinivasan, S. V. Adve, P. Bose, and J. A. Rivers, "The impact of technology scaling on lifetime reliability," in *Proc. Int. Conf. on Dependable Systems and Networks (DSN'04)*, 2004, pp. 177–186.

[17] J. Hennessy, Stanford Benchmark Suite Personal communication.

[18] S. S. Mukherjee, C. Weaver, J. Emer, S. K. Reinhardt, and T. Austin, "A systematic methodology to compute the architectural vulnerability factors for a high-performance microprocessor," in *Proc. 36th Annu. Int. Symp. on Microarchitecture*, 2003, pp. 29–40.