SRAM Failure Injection to a Vehicle ECU and Its Behavior Evaluation

Yusuke Takeuchi[1], Yohei Nakata[1], Yasuhiro Ito[2], Yasuo Sugure[2],
Shigeru Oho[3], Hiroshi Kawaguchi[1], and Masahiko Yoshimoto[1, 4]

[1]Graduate School of System Informatics, Kobe University, Kobe, 657-8501 Japan
[2]Central Research Laboratory, Hitachi, Ltd., Kokubunji, 185-8601 Japan
[3]Department of Electrical and Electronics Engineering, Nippon Institute of Technology, Minamisaitama, 345-8501 Japan
[4] Japan Science and Technology Agency (JST) CREST, Tokyo, 102-0076 Japan
E-mail: takeuchi_u@cs28.cs.kobe-u.ac.jp

***Abstract*—Recently, technology of electronic control units (ECUs) mounted on vehicles has progressed. ECUs are now used in various applications such as throttle control, brake control, and pedestrian recognition. Accordingly, testing to ensure ECU dependability has become complicated. We have developed a model-based evaluation environment that can readily analyze the effects of the device-level SRAM failure on an entire vehicle system. Injecting failures that occur in this environment yielded data related to fuel injection quantity and vehicle speed. These data clearly reflect the effects of the SRAM failures. We have also constructed a demonstration system to report the effects of SRAM failure intelligibly.**

Keywords—ECU; model-based evaluation environment; SRAM; demonstration system

I. INTRODUCTION

With the advent of automobile electronics, electronics control units (ECUs) mounted on vehicles have been increasing year by year. The ECUs mounted on a typical car now number 30–60. A luxury vehicle might have 100 or more ECUs mounted. The use of ECUs has extended to higher functionality, such as intelligent transport systems (ITS) and pursuit of safety and comfort. For an automotive LSI, the most important consideration is dependability. The operating voltage and temperature must be managed carefully. Nevertheless, tests to clarify their function are difficult to conduct. They are costly and time consuming. To make matters worse, managing the behavior of a whole vehicle system is virtually impossible. An efficient evaluation environment to assess vehicle dependability is therefore earnestly sought by engineers.

We propose a model-based environment to evaluate an automotive LSI. Particularly in this paper, we specifically examine SRAM. On CPUs used today, SRAM occupies a large number of the transistors. Therefore, the dependability of the LSI operation can be regarded as reliant on the SRAM. We injected various failures to the SRAM in an engine control ECU and assessed the impacts of the failures on a vehicle. We also created a demonstration system for vehicle behavior evaluation.

II. FAILURE INJECTION SYSTEM

A. Evaluation System

Fig. 1 presents an illustration of the concept of the evaluation environment. It comprises three models: a vehicle model, engine model, and ECU model. It is called a cyber physical system (CPS), which is a collaborative system of electronic and mechanical subsystems. The kinetic behavior of the vehicle (e.g. transmission and tire friction calculation) is simulated by CarSim®[1]. The engine (torque production and engine speed calculation) is simulated by Matlab®/Simulink®[2], based on a 3.8 L V6 gasoline engine. The ECU behavior is simulated by Virtualizer™[3].

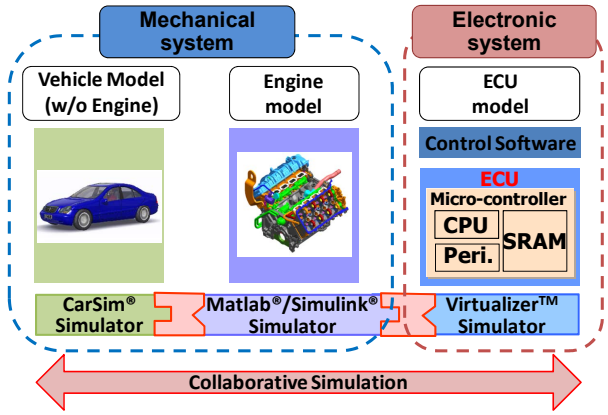

Fig. 1. Proposed evaluation environment.

Failure injection to the ECU is presented briefly in Fig. 2. The fault case generator (FCG) generates time-series failure data patterns according to evaluation conditions: supply voltage, temperature, NBTI aging (e.g. 10 years), and SRAM capacity. The fault-injectable bus bridge (FIB) is inserted between CPU and internal SRAM. By injecting a failure data pattern to the internal SRAM through the FIB, data destruction can be modeled.

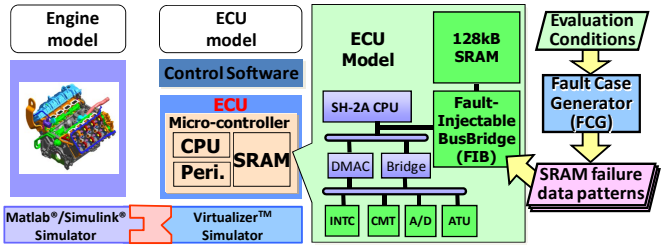

Fig. 2. Failure injection to ECU.

[1] CarSim® is a registered trademark of Mechanical Simulation Corporation.
[2] Matlab®/Simulink® is a registered trademark of The Math Works, Inc.
[3] Virtualizer™ is a trademark of Synopsys, Inc.

B. *SRAM fault case generator*

Fig. 3 shows a block diagram of the FCG. As inputs, the FCG uses device conditions of the SRAM (supply voltage, temperature, process variation, NBTI aging, soft error rate, capacity), SRAM bit error rate (BER) libraries obtained from a transistor-level Monte Carlo simulation, and information of a virtual chip (described below). The SRAM failure data pattern generator refers to the SRAM BER libraries and obtains one datum corresponding to the SRAM device conditions. Using the BER and virtual chip information, the FCG generates time-series SRAM failure data patterns as output. The supply voltage and temperature can change with time.

A virtual chip represents one chip considering SRAM variability. On an actual LSI, SRAM failures are attributable to poor read and write margins derived from transistor random variation, or because of soft errors that occur randomly: the failure might occur randomly in time and space dimensions. Information of the virtual chip is expressed as failure occurrence locations and times. Using the virtual chips, we can execute large-scale simulations assuming chip variation.

TABLE I presents a summary of the parameters for our simulation. The design rule is 65 nm, and the process is at the typical corner. For the NBTI aging in a pMOS, we assume that the threshold degradation is –24 mV in 10-year aging.

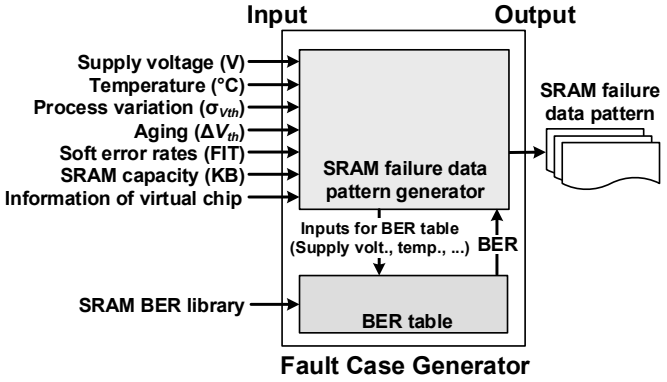

Fig. 3. Fault case generator.

TABLE I. CO-SIMULATION PARAMETERS

Engine spec.	3.8 l V6 gasoline engine
Vehicle spec.	Full-size sedan
Initial velocity of vehicle	80 km/h
SRAM capacity	128 kB
#of virtual chips	1000
Execution time	3 s
Range of supply voltage	0.4 V – 0.8 V
Range of temperature	-50 °C – 150 °C
σ*Vth* of PMOS, NMOS	40 mV, 30 mV
Delta *Vth* of PMOS	-24 mV

III. EVALUATION

A. *Evaluation Methodology*

Using the proposed evaluation environment, we inject the SRAM failures and evaluate the vehicle dependability. As the internal SRAM in the ECU, we use the conventional 6T SRAM. By injecting the SRAM failures to the co-simulation environment with the vehicle, engine, and EUC models, we can obtain run-time behavior of the software and vehicle. The software in the co-simulation is sometimes terminated abnormally because of the failure injection. In abnormal termination, the simulation presents two cases: a watchdog timer interruption (runaway of the software) or an access violation to an illegal address (bus error).

First, to obtain baseline data, we once run the co-simulation without failure injection, which produces normally ended trace data. The runtime of the co-simulation is 3 s on each virtual chip. The accelerator throttle input is presented in Fig. 5(a). Without injecting failures, the fuel injection quantity corresponding to the accelerator throttle input should be Fig. 5(b). The vehicle velocity results in Fig. 5(c).

We inject failures to 1000 virtual chips. Even in a failure injection case, sometimes neither the runaway nor bus error occurs. It appears to be a normal termination, but the vehicle speed might not be the value we expect. We define such an outcome as abnormal vehicle behavior. If a vehicle velocity at the end of the co-simulation (Time = 3 s) is in error by 0.5 km/h or more in comparison with Fig. 5. It is handled as an abnormality. TABLE II shows a summary of the abnormalities.

TABLE II. SUMMARY OF ABNORMALITIES

Name of abnormality	Description
Runaway of software	Watchdog timer interrupts the software and stops the simulation.
Bus error	Error occurs when accessing illegal address of SRAM and stops the simulation.
Abnormal vehicle behavior	Vehicle velocity at 3 s is in error by more than 0.5 km/h in comparison with baseline waveform.

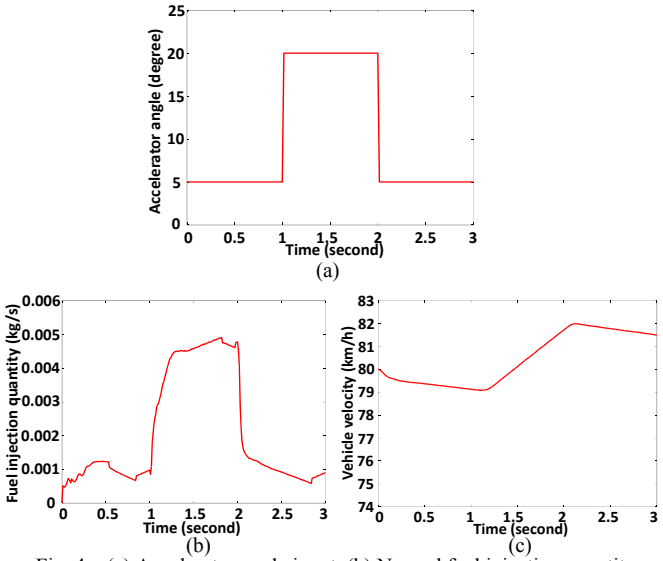

Fig. 4. (a) Accelerator angle input. (b) Normal fuel injection quantity. (c) Vehicle velocity without failure.

B. *Evaluation Result*

Fig. 5 shows the failure-injected waveforms of the fuel injection quantities and the corresponding velocity data obtained from the co-simulation when an operating voltage in the SRAM is varied. The temperature is set to 0 °C. Among these waveforms, 11 waveforms (i.e. 11 virtual chips) are judged as having abnormal behavior at 0.6 V, 387 at 0.5 V, and all at 0.4 V because the operation voltage drop worsens the SRAM BER, which in turn negatively affects vehicle reliability.

As Fig. 5 shows, the fuel injection quantity and vehicle velocity are closely related. At the operation voltage of 0.5 V, three patterns of abnormality are apparent in the waveforms:

- Pattern 1: Fuel is somehow injected and the vehicle speed increases between during 1–2 s; their waveforms are out of alignment for the normal waveforms. This phenomenon is explicable when failures are injected to some of the 32 bits in the fuel injection quantity variable. The difference from the normal waveform becomes greater as the failure is injected to a more-significant bit.
- Pattern 2: The fuel injection quantity is fixed to a certain small value. The vehicle speed decreases slightly. In this case, the engine control software runs away because of SRAM failures, although the watchdog timer is not triggered. The fuel injection quantity variable cannot be updated and is therefore fixed. In this pattern, the value of the injection quantity is constant, whereas the vehicle speed changes slope at the time of 1 s and 2 s. This is related to the air–fuel ratio. The inflow of air varies along with the throttle angle variation. Therefore, the air–fuel ratio changes, thereby changing the engine speed and the produced torque.
- Pattern 3: Fuel is not injected to the engine at all. An output enable variable in SRAM domain is tampered.

Fig. 6 presents the number of abnormal virtual chips when the temperature is varied. The range in red shows the number of abnormal terminations (runaway of software and bus error). Another in blue represents the number of abnormal vehicle behaviors. At high temperatures, the number of abnormal virtual chips becomes greater because the read margin of the SRAM degrades. At 0.7 V or higher voltage, abnormal termination is not observed.

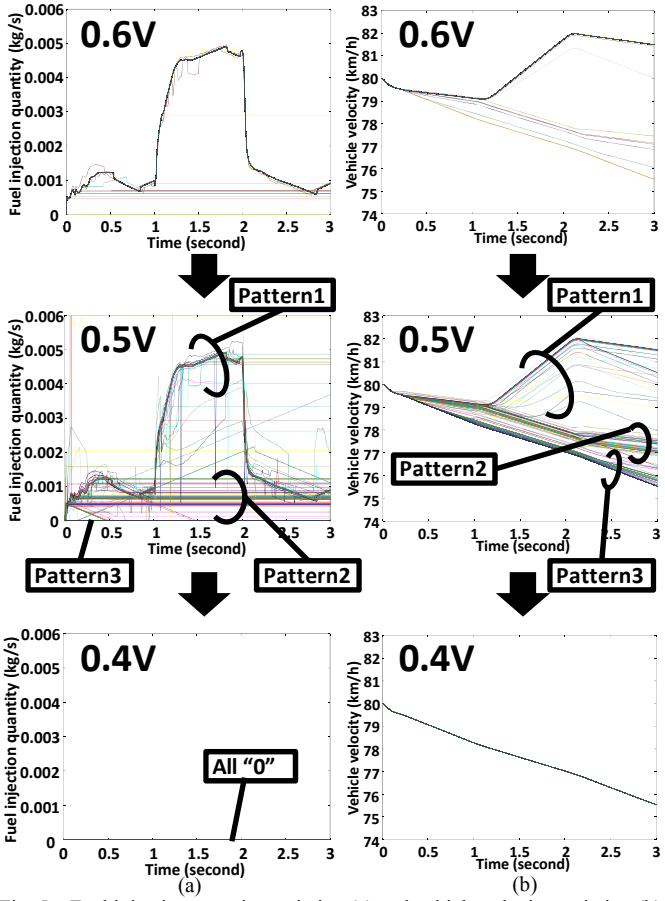

Fig. 5. Fuel injection quantity variation (a) and vehicle velocity variation (b) with the voltage drop.

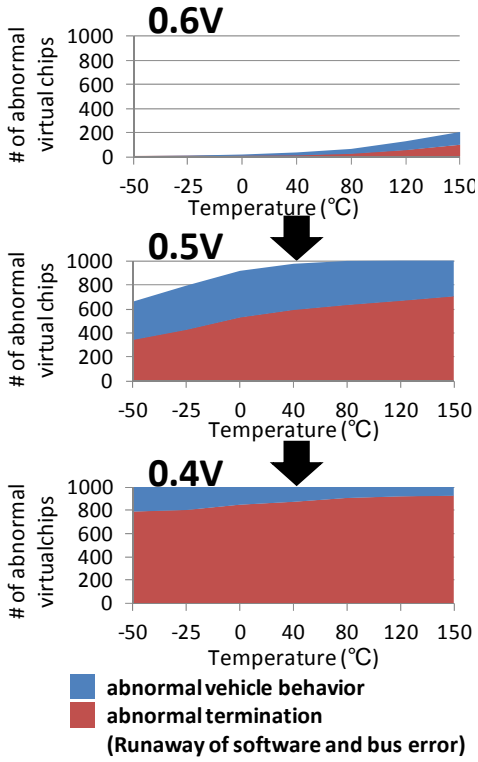

Fig. 6. Variation of the number of abnormal chips by the voltage drop.

IV. DEMONSTRATION SYSTEM

To elucidate the effect of failure injection on vehicle behavior, we created a demonstration system. This system consists of the engine model (Matlab®/Simulink®) and vehicle model (CarSim®); moreover, we added software that reflects vehicle graphics (VehicleSim Visualizer). At this time, we did not use Virtualizer™ to simulate the ECU because it is unsuitable for real-time simulation. Therefore, we prepared another ECU model and failure model, which modified the output signals from the ECU to Matlab®/Simulink®. The failure model can not simulate strictly in the same way as Virtualizer™. We selected and modified important variables only. Figure 7 shows the demonstration system data flow:

1. A wheel and an accelerator are connected to a PC in the demonstration system. They obtain a wheel angle and an accelerator angle and communicate them to the engine model.
2. The engine model transfers its states to the ECU.
3. The ECU model dispatches the subsequent command signals. For example, if the engine is at low speed and the accelerator angle is large, the ECU model injects more fuel to the engine.
4. The failure model modifies important variables. The modification is made based on the failure data pattern that has been prepared in advance. The important engine control variables, comprising 32 bits, are merely modified.
5. Using the fuel injection variables, the engine model calculates an engine speed and torque, which are forwarded to the vehicle model (CarSim®). It simulates the vehicle behavior.
6. The vehicle behavior appears on the display with VehicleSim Visualizer, and the reaction force in steering is propagated to the wheel.

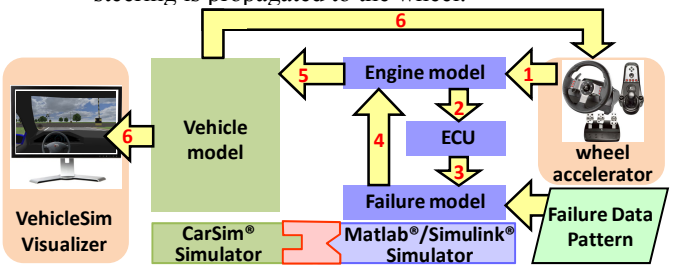

Fig. 7. Data flow of the demonstration system.

In the demonstration system, we prepared a scenario related to a railroad crossing. We prepared a driving course that has a railroad across it. The railroad crossing is well known to emit an electromagnetic noise [4]. When the vehicle approaches the railroad crossing, the ECU gets large noise and its supply voltage decreases. The BER increases because of the operation voltage drop, which causes SRAM failures.

Fig. 9 shows an actual image displayed on the monitors in the demonstration system. The left side in Monitor 1 shows parameters: a supply voltage, accelerator angle, fuel injection quantity, and engine speed. The right side in Monitor 1 shows an SRAM failure map on which red dots represent failure bits. Monitor 2 shows actual vehicle graphics displayed on the VehicleSim Visualizer. A person can drive it by grabbing the wheel. This monitor indicates that the vehicle does not move even if stepping on the accelerator, and it collides with the train. The accelerator failure occurs when the crossing was approached, the noise goes into Engine control ECU, causing SRAM failure.

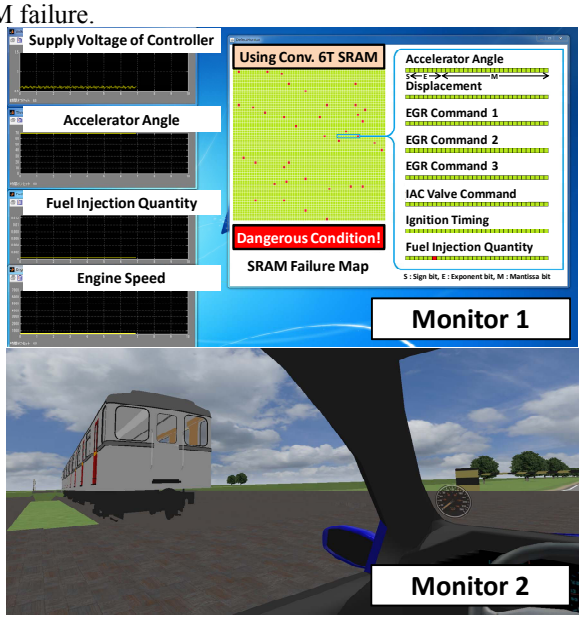

Fig. 8. Image displayed on the monitors.

V. CONCLUSION

We have developed a model-based vehicle dependability evaluation environment that can inject device-level SRAM failure. After using this environment and injecting SRAM failure, we observed the vehicle velocity and confirmed that the failure effects of important variables appeared clearly. Therefore, vehicle ECU evaluation can be performed easily. We have also created a demonstration system that readily portrays the effects of SRAM failure on the vehicle behavior.

ACKNOWLEDGMENT

This work was supported by Synopsys Inc. for providing the licenses of Virtualizer™ in this paper.

REFERENCES

[1] Y. Nakata, Y. Ito, Y. Sugure, S. Oho, Y. Takeuchi, S. Okumura, H. Kawaguchi and M. Yoshimoto, "Model-Based Fault Injection for Failure Effect Analysis – Evaluation of Dependable SRAM for Vehicle Control Units –," *Workshop on Dependable and Secure Nanocomputing (WDSN), in conjunction with International Conference on Dependable Systems and Networks (DSN)*, pp. 91-96, June, 2011.

[2] E. Seevinck, F.J. List, and J. Lohstroh, "Static-noise margin analysis of MOS SRAM cells," *IEEE Journal of Solid-State Circuits,* vol. 22, no. 5, pp. 748-754, 1987.

[3] R. Heald, and P. Wang, "Variability in sub-100 nm SRAM designs," *IEEE/ACM International Conference on Computer Aided Design (ICCAD),* pp. 347-352, 2004.

[4] S. Niska, "Electromagnetic Interface: A major Source of Faults in Swedish Railway," *International Journal of Performability Engineering (IJPE),*vol. 5, no. 2, pp. 187-196, 2009.