Integrated software platform for automotive systems

Chiharu Takei, Hiroaki Takada, Masaki Yamamoto, Shinya Honda
Center for Embedded Computing Systems
Nagoya University
Nagoya, Japan
takei@nces.is.nagoya-u.ac.jp

***Abstract*—Advances in microprocessor technology provide sophisticated automotive control systems, in both realtime systems and information control systems. However, this trend brought the electronic control unit (ECU) complexity and large-scale software. One important solution will be 'Integration' on both software and hardware side. Center for Embedded Computing Systems, Nagoya University (NCES) has performed several studies regarding this integration for software platform, especially from the point of view of realtime operating systems**

I. Introduction

Major recent topics in automotive systems are complexity and increasing scale of controlling software. According to the requirements from safety, comfort and ecology, scale of automotive software is going to become one Mega lines, including engine control, body control, chassis control and navigation systems. It is almost same scale as banking systems of mega-bank. Most recent automotive system adopts more than 80 ECUs in one vehicle.

To keep the development term in a reasonable range, following considerations are to be required.

- Prepare common software platform for ECU to reduce the code newly developed.
- Introduce software packages those available in the market.
- Prepare the integrated automotive data from many sensors to make it easy to design new application systems. Also, standardization of data format and API method should be considered.

From the reliability point of view, following consideration is also required.

- Fine grain monitoring and recovering systems.

This paper introduces these studies established in NCES.

II. Safe-G: System protection for dual OS systems

A. overview

Previously, realtime control systems were commonly used in automotive systems. However, information processing systems also be adopted in the systems, such as navigation systems, audio and visual systems and Internet connected applications. Since these systems are large; the reliability level of the software is not so enough as required level of automotive environment, even though these systems are running on a ECU together with realtime systems. .

Therefore, architecture that protects the realtime systems from the hazard in information processing systems should be considered.

B. Approach.

There are some proposals to secure the protection between two operating systems.

1) - Virtual Machine (VM) method

VM monitor will manage all resources corresponding to the hardware systems, such as memory, interrupt controller and special registers. VM monitor will control several guest OSs as if those guest OS can manage resources directly. [1] This system, however, has relatively large overhead, especially, if the processor does not have special functions supporting VM. Therefore, this system is not considered to fit the realtime systems.

2) -Hybrid OS method

In this system, all resources are distributed into each OS prior to execution. As each OS controls the resources directly, system overhead is smaller than VM method. Therefore, it will be adoptable in realtime systems. But, on the other hand, if one OS has some hazard and access to another OS's resources the protection will not work enough, because this system assumes that each OS act as "gentlemanly behavior".

C. Result

We have adopted new hardware protection mechanism, provided by ARM ltd. onto ARM11 processor, called "TrustZone" enabling both small overhead and full protection. In this mechanism, execution mode will be separated into two modes: "trusted mode" and "non-trusted mode". Trusted mode is similar as Hypervisor mode, all resources including system critical resources can be accessed in this mode. In non-trusted mode, however, resources those assigned to trusted area can not be accessed physically.

Also, small monitor software called SafeG was developed to control TrustZone. Isolating two OSs by SafeG, interruption also is managed by SafeG. Therefore, a realtime system is able to keep its control even if another system's interruption handler occupies CPU resource by its problem. [2]

978-1-4244-2599-0/08/$25.00 ©2009 IEEE

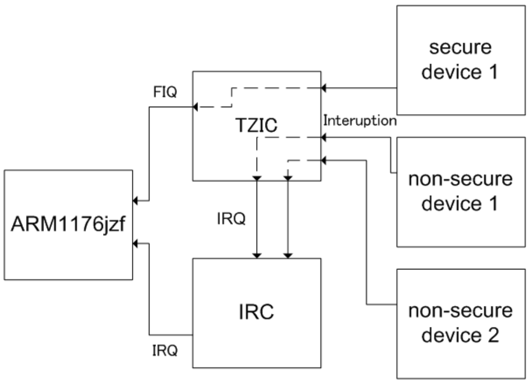


Figure1. Hardware configuration

Overhead of interruption response time was measured and it was smaller than 5 micro seconds when SafeG is activated in the environment of microITRON+SafeG+Linux on ARM1176@210MHz. It is reasonably small for realtime systems.

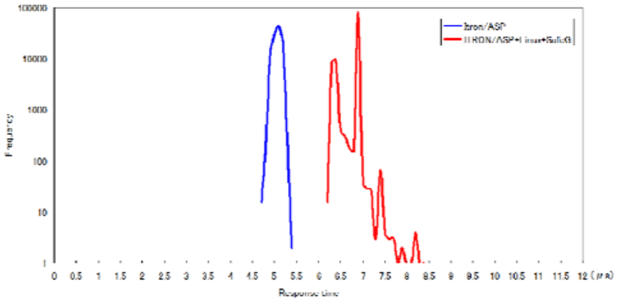


Figure2. Overhead of SafeG

D. *Summary*

New SafeG architecture, which provides hardware protection on to hybrid OS, enabled small overhead and secure protection for the dual operating systems. This system is flexible for the number of cores, single core to multiple cores. Using SafeG, new secure automotive platform, in which realtime systems and large scale information systems run concurrently, will be realized.

III. Data Integration for Automotive Systems

A. *Overview*

For the safety of the vehicle, many sensors are adopted in automotive systems, therefore, management and development of new application becomes more difficult.

To improve the efficiency of utilizing the sensor data, those data should be shared among applications. New data handling systems, those provide data to applications with low latency in the required format, should be studied.

B. *Approach*

Expressing the running environment of automobile into specific data space, then unified access method for this data space was studied.

The merits of this methodology are:

- Application program does not need to control sensor system. Or rather, it easily gets the required data form the data space.
- Individual design is available for both sensor systems and application.
- Flexibility of the system.
- Sensor cost will be reduced by cutting redundant sensor system.

C. *Result.*

Two methods were studied.

1) *Based on data streaming management systems.*

Hierarchical management of the sensor data provides required data to the application in each abstraction levels. This method has a merit that effective data access will be done by continuous query language (CQL). But, on the other hand, it is difficult to manage an indexed data. [3]

2) *Based on relational data base systems.*

Adopting RDB, Occupation Grid and Scene graph, the unified database was constructed. Requested data will be passed through the API defined by this system. The merit of this method is easy to manage by Occupation Grid or Scene Graph. But demerit is overhead on getting the data. Also, relational data base does not fit to streaming data.

Both systems were installed and evaluated. Some application program was developed to evaluate these data management systems, such as “Computer Assisted Parking System” and “Automatic Crouse Control System”. As the result, above mentioned characteristics was observed.

D. *Summary and future plan*

Two integrated data management systems were designed and evaluated. More detailed evaluation will be needed depending on the requirements from automotive applications. Standardized management format and access API should also be defined and be opened. Unification with inter-vehicles communication and vehicle-infrastructure communication will also be defined and be opened.

IV. Fine Grain Monitoring and Recovering Systems.

A. *Overview*

As ECU becomes complex and scale of software becomes large, once system trouble has been occurred the influence to the systems becomes more serious.

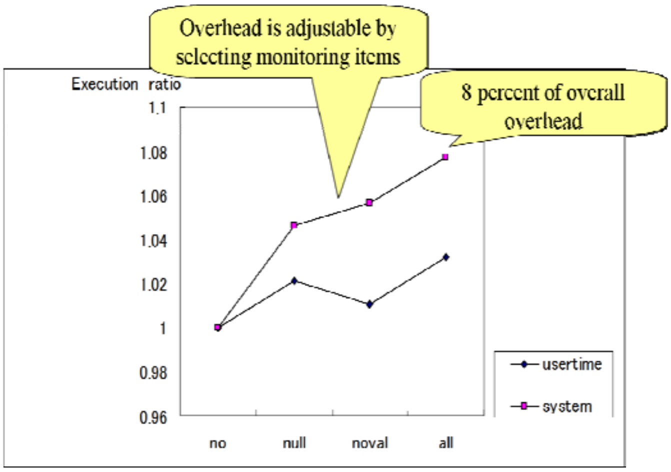
Hierarchical monitoring systems, therefore, is considered that covers whole ECU, whole OS systems, whole application and subsystems. When the monitoring systems detects problem, system recovery will be activated depending on the level of the problem. Therefore, total systems influence will be reduced.

B. Evaluation and Result

Hazard and operability study (HAZOP) was adopted to analyze the monitoring point and confirm the sufficiency of the condition.

Dtrace, available in open SOLARIS, was used as a probing monitor. [4]

As the result, trouble detection was certainly performed with the overhead of less than 8 % of execution time.



Overhead is adjustable by selecting monitoring items

8 percent of overall overhead

C. Summary

Dtrace is useful for the fine grain monitoring of process. More detailed evaluation for Dtrace should be planned. Other monitoring system than Dtrace will also be researched. Also, more fine grain monitoring and recovering system on sub process will be planned. [5]

V. Summary of Paper

The key words of the integration of automotive software platform are:

- Large scale software.
- Complex software/hardware systems.
- High reliability.
- Separation and coexistence of realtime systems and information systems.
- Integrated data management for automotive systems.

Along to these keywords, we NCES has been studied following themes:

- SafeG which realize safety separation of realtime OS and informational OS.
- Integration of automotive data.
- Fine grain monitoring and recovering.

Each systems were evaluated and proved its usefulness.

For the future activity, expansion on the multi-core system and standardization of automotive data format will be planned.

Acknowledgment

. The authors would like to thank Prof. Hiroaki Takada, Prof. Hiroyuki Tomiyama, Prof. Masaki Yamamoto, and Assistant Prof. Shinya Honda for their encouragement and suggestions of this work.

References

[1] T. Horie, H, Eiraku, T. Shinagawa, K. Kato“Virtual Machine Monitor "BitVisor" and I/O Virtualization Technology” IPSJ SIG Notes 2009(6) pp.35-41 20090121

[2] K. Nakajima, S.Honda,S,Teshima, H.Takada, “Enhancing reliability in Hybrid OS system with security hardware,” IPSJ SIG Technical report 2008(116) pp.1-7 20081120

[3] M, Yamada, H, Kamada, K. sato, S. Teshima, H. Takada, ”Distributed Sensor Data Procesing System with Streaming Processing Model for Vehicle Integrated Control System” IPSJ SIG Technical report 2009(24) pp.79-85 20090226

[4] McDongall, R. Manto,J. and Gregg,B.:”Solaris Performance and Tools:Dtrace and Mdb Technique for Solaris 10 and Open Solaris”, Reutice Hall(2006)

[5] K. Akiyama, Y. Suzuki, N. Tsumura, Y. Nishimura, ”Autonomic Computing Architecture for Ebmedded Devices” PROVISION 58, IBM(2008)