Development of CRAMAS_MCC-CAN

Junji Takahashi
Takeshi Yamasaki
Jun Nishii
Kaoru Noumi
Kazushi Mahana

Abstract

Due to the increasing sophistication and complexity of vehicle control, recent years have seen a rapid rise in the number of electrical equipment items - especially ECU (Electric Control Unit) - that are installed in automobiles. This embededd control such as vehicle stability and chassis, to realize which the utilization of in-vehicle networks has now begun in earnest, and a rapid expansion is taking place in the communication data capacity of CAN (Controller Area Network) - the communication protocols widely employed in the automobile industry. How to perform efficiently the annually multiplying communication design needed for vehicle systems development, and to assure its quality, are important issues for automobile manufacturers and ECU suppliers. Our company likewise faces a problem in the large number of man-hours consumed by CAN communication checks. In order to resolve such problem, we have now developed a CAN automatic communication function checking system called "MCC (Multi Communication Checker) - CAN", which uses our own "CRAMAS" (ComputeR Aided Multi-Analysis System) HIL simulator and incorporates an ECU supplier's know-how. The present paper describes the new checker.

1 Background of the development

Due to the increasing sophistication and complexity of vehicle control, recent years have seen a rapid rise in the number of electrical equipment items - ECU (Electric Control Unit), sensors, actuators and so on - that are installed in automobiles. This embededd control such as vehicle stability and chassis, to realize which the utilization of in-vehicle networks has now begun in earnest. Against this background, a rapid expansion is taking place in the communication data capacity of CAN (Controller Area Network), the communication protocols widely employed in the automobile industry. How to perform efficiently the annually multiplying communication design needed for vehicle systems development and to assure its quality are important issues for automobile manufacturers and ECU suppliers. In order to resolve such problems we have now developed a CAN communication function checking system which uses our own "CRAMAS" (ComputeR Aided Multi-Analysis System) HIL (Hardware In the Loop) simulator.

2 Current situation of communication protocol design in ECU development

2.1 Problems with current methods

With the enlargement of in-vehicle networks, communication protocol design has come to account for a large proportion of the work of vehicle system development. As a result, ECU suppliers are now faced with the issue of enhancing the efficiency of such design work in order to cope with its increasing volume; in our company's ECU development, the number of man-hours taken up by communication function checking has reached high levels. Analysis of the factors involved yields the following reasons for this.

1) Large number of check items

Conducting the standard CAN communication function checking, plus checking to meet the checking standards stipulated by automobile manufacturers, entails running checks on over 100 items. Around half of these checks must be repeated at each prototype stage, and each time a specification change occurs, on the way to mass production. (Refer to Fig. 1.)

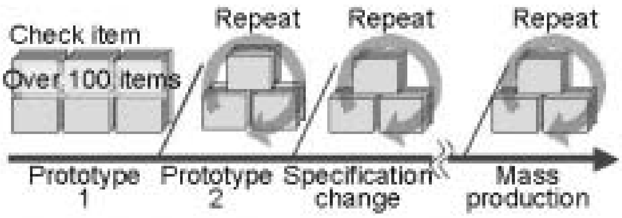

Fig.1 Process of repeated communication function checking

2) Special checking equipment is required

In many cases, checking of CAN communication requires the use of specialized apparatus including bus analyzers and error generators. But since specialized equipment of this kind is generally costly, it is no simple matter to prepare the quantity of such equipment that will be adequate for speedy checking of more than 100 items. Moreover the designer must perform by manual operations all of the work from setting of the equipment through to implementation of the checks. Thus the checking requires both expert knowledge of the specialized equipment and time for the performance of such operations.

3) Expert and high-level skills are required

In order to conduct the tests, one must have understanding of numerous specifications including the CAN communication protocol, the ECU specifications, the automobile manufacturer's own particular communication protocol specifications, and also the checking specifications. The check methods too have to be standardized so as to prevent inconsistency in the checking quality levels. Development of ECU incorporating CAN is set to continue expanding in the future, and securing designers possessing all of the above skills needed for such work is anticipated to be problematic.

The above 3 items constitute the difficulties for Fujitsu Ten communication protocol design, and we believe that most of them are probably also applicable to other ECU suppliers.

2.2 Countermeasures to deal with the problems

The following 3 countermeasures may be enumerated for curbing the number of man-hours needed for communication function checking:

1) **Create methods that will permit large numbers of checks with a small number of man-hours**
   → Automate the checking
2) **Simplify checking apparatus preparation and the checking procedure**
3) **Eliminate the need for special skills in order to conduct the checking**
   → Standardize the checking content and methods
   (so that the checks can be conducted without knowledge about the communication)

In order to put the above 3 countermeasures into practice, we developed an automatic communication function checking system dubbed "MCC-CAN", which uses the "CRAMAS" HIL simulator produced by Fujitsu Ten.

3 Overview of MCC-CAN

The concept behind the development of the MCC-CAN system was to have the CRAMAS simulate a vehicle communication environment that would permit a targeted ECU to be checked automatically and with ease when connected to such environment. Specifically, MCC-CAN automatically checks the behavior of the targeted ECU with regard to the in-vehicle network and with regard to the communication data from such network. The pillars of the system are its 3 basic checking functions employing simulations of mounted-on-vehicle environments. These functions are the following:

1) Checking of the target ECU's degree of satisfaction with the communication specifications, via reproduction of communication loads comparable to those in a mounted-on-vehicle environment.
2) Checking of the target ECU's fail-safe functions, via reproduction of communication line status changes (shorts between power supply and ground, broken/dis-

connected wires) in a mounted-on-vehicle environment.

3) Checking of the target ECU's susceptibility to power supply voltage fluctuations and measurement of its performance, via reproduction of power supply voltage fluctuations comparable to those in a mounted-on-vehicle environment.

The configuration of the MCC-CAN system is shown in Fig. 2.

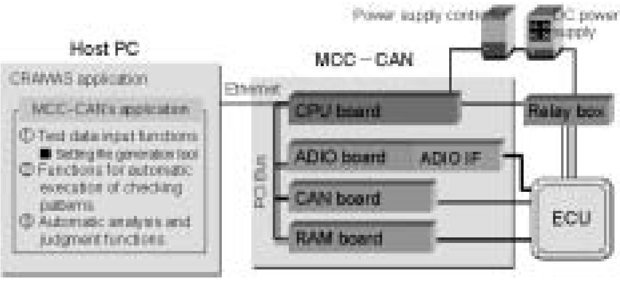

Fig.2 Configuration of MCC-CAN

3.1 Hardware configuration

MCC-CAN is configured from CRAMAS standard hardware. The roles of the various pieces of hardware are described below.

1) CAN board

This has CAN communication emulation functions, and creates CAN transmission/reception logs assigning the check execution time stamps that are necessary for the check judgments.

2) RAM board

This has functions for writing in and reading out values to and from the ECU microcomputer's RAM, and creates RAM measurement logs assigning the check execution time stamps that are necessary for the check judgments.

3) ADIO and ADIO-IF boards

These analog/digital IO boards make the voltage measurements necessary for deciding the result of the ECU power supply voltage fluctuation test.

4) External relay box

This performs individual ON/OFF switching of each of the following power supplies to the ECU: BAT (battery power supply), ACC (accessory power supply) and IG (ignition power supply).

5) Power supply controller (commercial product)

This is able to vary the DC power supply employed as the ECU's power supply, and is used for the ECU power supply voltage fluctuation test during the communication function checking.

3.2 Descriptions of the functions

3.2.1 Test data input functions

These functions are realized by the "setting file generation tool" (Fig. 3) which is one of the items among MCC-CAN's application software.

The ECU receives and transmits, in the form of communication data, large amounts of information relating to vehicle control. It handles information of a multiplicity of types ranging from information for control of the vehicle's basic operations of "forward motion, turning bends and stopping" to information indicating the control status. Manually performing all of the checking apparatus settings needed to check such a huge mass of communication data would require a considerable number of man-hours. The "setting file generation tool" has functions for entering the data that the checking requires into a communication specification (electronic data) sheet drawn up in a standard format, by means of which it is able to reflect automatically in the checking patterns the different sets of communication specifications for each different ECU. It further possesses functions for reflecting checking specifications entered manually by the user, and various different parameters for the different ECU software specifications, in the checking patterns. This entire series of inputs can be made via GUI*1 on MCC-CAN's host PC, and moreover can be made in wizard style, thus enabling even first-time users of the tool to make the checking settings with ease.

Fig.3 Screen of setting file generation tool

3.2.2 Functions for automatic execution of checking patterns

All of the operations required in order to execute the checks after the checking settings have been completed can be made via a dedicated GUI referred to as the "MCC manager" (Fig. 4) which provides the requisite interfacing between MCC-CAN and the user.

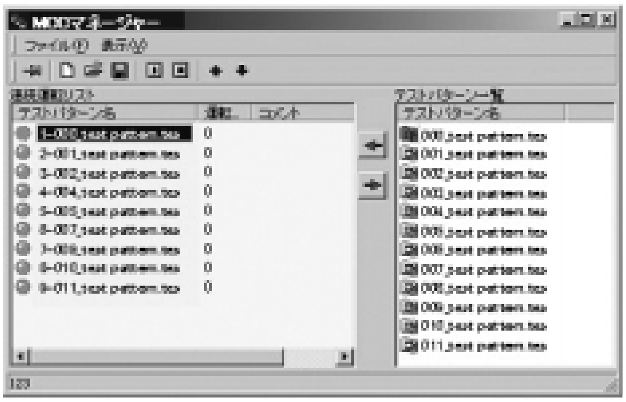

Fig.4 MCC manager

---

*1 GUI (Graphical User Interface):
generic term for user interfaces that provide ease of user operation by permitting file operations and running of application software, etc., to be performed simply by clicking on windows, icons and the like.

Merely by selecting the checking pattern(s) to be executed on the MCC manager and clicking on the execute button, users can have any desired checking executed automatically. And they can use the GUI shown in Fig. 5 to ascertain the checking pattern execution status while the checks are being run.

Fig.5 GUI of check pattern execution

3.2.3 Automatic analysis and judgment functions

These are functions that are run in tandem with automatic execution of the checking patterns selected on the MCC manager by the user. These analysis and judgment functions compare with the expected values the measurement log values (for CAN bus, ECU microcomputer's RAM value and power supply operations) that are output as measurement results following automatic execution of the checking patterns, and output "OK" or "NG" judgments on the basis of such comparison.

4 Features of MCC-CAN

There follow descriptions of the 5 major features of MCC-CAN.

1) Highly effective at reducing man-hours

The greatest advantage of the MCC-CAN automatic checking system is that once the checking equipment settings are made it automatically executes the checking and automatically makes the "OK" or "NG" judgments from the measurement results. This is equivalent to 80% of the checking (as a fraction of the total check items) when performed manually in the usual manner, and the man-hour reduction effect will be greater in proportion to the number of check items implemented. As an example for reference, benchmark results obtained at Fujitsu Ten are provided. It can be seen that introduction of MCC-CAN yielded a 55% reduction in man-hours compared to the situation before introduction in Fig. 6.

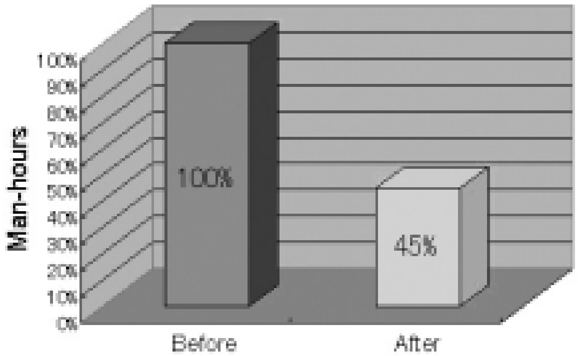

Fig.6 Man-hour reduction result by MCC-CAN at Fujitsu TEN

2) Provides easy creation of checking patterns

Checking patterns conforming to ECU communication checking specifications are registered in the MCC-CAN application software. As shown in Fig. 7, basic operation patterns executable by the MCC-CAN hardware are combined to create checking patterns conforming to the communication checking specifications.

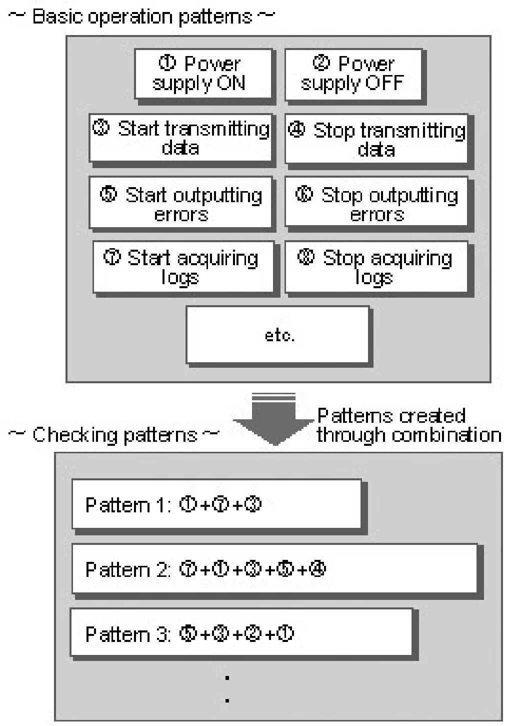

Fig.7 Procedure of checking patterns

Thanks to such architecture whereby checking patterns are created via formulation or combination of basic operation patterns - in other words, thanks to such platformization - the system makes it possible to add or alter checking patterns, or to accommodate CAN communication protocol and other communication protocols besides, using just a minimum of man-hours.

3)Easy to introduce

Because MCC-CAN is configured from CRAMAS standard hardware, the only additional purchases that our company outfits and other users who have already introduced CRAMAS need to make in order to introduce the system are of the boards and external equipment that the MCC-CAN system requires, plus the MCC-CAN application software. Thus the system can be introduced at low cost. And by replacing the CAN board with other communication protocol boards, requirements for new communication protocols can be met with a minimum of additional hardware (Fig. 8). Moreover, operation of the MCC-CAN application software mainly uses Windows application GUI environments, just as CRAMAS does, so that automatic checking can be executed simply by entering data via the host PC, without the need for any special operations.

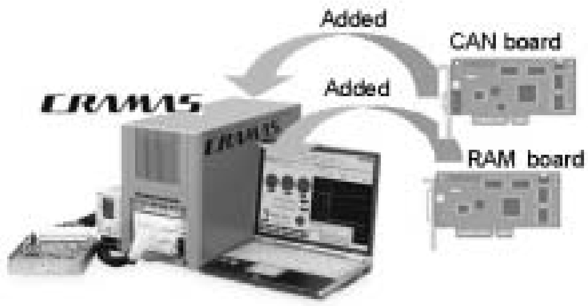

Fig.8 Example of addition to existing CRAMAS

4) Quality improvement

The present MCC-CAN has the purpose of executing checks stipulated by automobile manufacturers. The latter can expect this system to assure constant quality, since it allows checks that they themselves stipulate to be performed free of inconsistency, using unified methods. We have also collaborated with automobile manufacturers to introduce checks for past defect patterns, enabling formulation of countermeasures to prevent defect recurrence. For ECU suppliers, the fact that the system is able to perform checks at a constant level of accuracy without dependence on the skills and ability of the persons implementing the checking means that it can eliminate overlooking of design errors revealed by checking, incorrect checks due to operational errors, and the like. Additionally, the man-hours saved by the automation of checking can be spent on conducting more detailed checks. Thus the system is capable of yielding overall improvements in ECU quality.

5) Simple analysis

Besides outputting the check results as "OK" or "NG" judgments represented by "○" and "×" MCC-CAN also outputs auxiliary results for problem analysis, that is, for assisting in using the check results to determine where the mistake(s) in the target ECU's design is/are located and what is the nature of the mistake(s). Such auxiliary results are generated at various stages of the checking process, from measurement log output through to the final judgments, and are of a total of 4 files, namely: CAN bus logs shown in Fig. 9; logs of the target ECU microcomputer's RAM values and similar shown in Fig. 10; files of judgments for individual check items compiled from analysis of the two foregoing types shown in Fig. 11; and files aggregating all of the check results shown in Fig. 12. By back-searching through these auxiliary result files, it is a simple matter to identify the problem location(s).

INDEX	STS	TIME	ID	DLC	DATA[0]	DATA[1]
0		1.4337	0x 037	4	0	0
0		1.4358	0x 120	8	0	0
0		1.4378	0x 206	5	0	0
0		1.4397	0x 348	7	0	0
0		1.4417	0x 037	4	0	0
0		1.4449	0x 340	7	0	0
0		1.4467	0x 30A	5	0	0
0		1.449	0x 30B	7	0	0
0		1.4498	0x 037	4	0	0
0		1.4508	0x 528	3	24	0
0		1.4519	0x 120	8	0	0
0		1.4531	0x 5E0	7	28	0
0		1.4539	0x 527	4	25	0
0		1.458	0x 037	4	0	0
0		1.46	0x 5C8	4	25	0

Fig.9 CAN bus measurement log

# SAMPSTEP=0.005000												
TIME[sec]	00_Time	01_BATT	02_ACC	03_IG1	04_IG2	05_B	06_TX	07_INFO_1	08_INFO_2	RAM	RAM2	RAM3
1.65	1.784	128	128	128	128	128	1	0	0	0	0	0
1.655	1.789	128	128	128	128	128	1	0	0	0	0	0
1.66	1.794	128	128	128	128	128	1	0	0	0	0	0
1.665	1.799	128	128	128	128	128	1	0	0	0	0	0
1.67	1.804	128	128	128	128	128	1	0	0	0	0	0
1.675	1.809	128	128	128	128	128	1	0	0	0	0	0
1.68	1.814	128	128	128	128	128	1	0	0	0	0	0
1.685	1.819	128	128	128	128	128	1	0	0	0	0	0
1.69	1.824	128	128	128	128	128	1	0	0	0	0	0
1.695	1.829	128	128	128	128	128	1	0	0	0	63	3
1.7	1.834	128	128	128	128	128	1	0	0	0	63	3
1.705	1.839	128	128	128	128	128	1	0	0	0	63	3
1.71	1.844	128	128	128	128	128	1	0	0	0	63	3
1.715	1.849	128	128	128	128	128	1	0	0	0	63	3
1.72	1.854	128	128	128	128	128	1	0	0	0	63	3
1.725	1.859	128	128	128	128	128	1	0	0	0	63	3
1.73	1.864	128	128	128	128	128	1	0	0	0	63	3
1.735	1.869	128	128	128	128	128	1	0	0	0	63	3
1.74	1.874	128	128	128	128	128	1	0	0	0	63	3
1.745	1.879	128	128	128	128	128	1	0	0	0	63	3

Fig.10 ECU microcomputer's RAM measurement log

	ECU transmission frame (FRAME1)				
	Transmission time stamp	Transmission ID	Transmission DLC	Transmitted data	
Time [ms]	FRAME1_Time	FRAME_ID	FRAME1_DLC	DAT_1	DAT_2
¥Result¥Tx_cyc	[1]	[2]	[3]	[4]	[5]
1433.7	–	×0x3T 0x037[2-1-7]	○0x4[2-1-6]	○0x0[2-2-4]	○0x0[2-2-4]
1435.8	–	–	–	–	–
1437.8	–	–	–	–	–
1439.7	–	–	–	–	–
1441.7	○ 8.00[2-1-5]	×0x3T 0x037[2-1-7]	○0x4[2-1-6]	○0x0[2-2-4]	○0x0[2-2-4]
1444.9	–	–	–	–	–
1446.7	–	–	–	–	–
1449	–	–	–	–	–
1449.8	○ 8.10[2-1-5]	×0x3T 0x037[2-1-7]	○0x4[2-1-6]	○0x0[2-2-4]	○0x0[2-2-4]
1450.8	–	–	–	–	–
1451.9	–	–	–	–	–
1453.1	–	–	–	–	–
1453.9	–	–	–	–	–
1458	○ 8.20[2-1-5]	×0x3T 0x037[2-1-7]	○0x4[2-1-6]	○0x0[2-2-4]	○0x0[2-2-4]

Fig.11 Individual check results file

Results	2-1-1	2-1-2	2-1-4	2-1-6	2-1-7	2-1-8	2-2-1	2-2-4	2-2-7
CHECK1	–	–	–	–	–	–	○	–	○
CHECK2	–	–	–	–	–	–	–	–	–
CHECK3	–	–	–	–	–	–	–	–	–
CHECK4	–	–	–	–	–	–	–	–	–
CHECK5	–	–	–	–	–	–	–	–	–
CHECK6	–	–	–	–	–	–	–	–	–
CHECK7	–	–	–	–	–	–	–	–	–
CHECK8	–	–	–	–	–	–	–	–	–
CHECK9	–	–	–	–	–	–	–	–	–
CHECK10	–	–	–	–	–	–	–	–	–
CHECK11	–	–	–	–	–	–	–	–	–
CHECK12	–	–	–	–	–	–	–	–	–
CHECK13	–	–	–	–	–	–	–	–	–
CHECK14	–	–	–	–	–	–	–	–	–
CHECK15	–	–	–	–	–	–	–	–	–
CHECK16	–	○	×	–	×	○	–	○	–
CHECK17	○	–	–	×	–	–	–	–	–
CHECK18	○	–	–	○	–	–	–	–	–
CHECK19	○	–	–	○	–	–	–	–	–
CHECK20	○	–	–	○	–	–	–	–	–
CHECK21	○	–	–	○	–	–	–	–	–
CHECK22	○	–	–	○	–	–	–	–	–
CHECK23	○	–	–	○	–	–	–	–	–
CHECK24	○	–	–	○	–	–	–	–	–
CHECK25	○	–	–	○	–	–	–	–	–
CHECK26	○	–	–	○	–	–	–	–	–
Overall	○	○	×	×	×	○	○	○	○

Fig.12 All check results file

5 Conclusion

In the present development we achieved a CRAMAS system that permits automatic check data entry, automatic execution, and automatic judgments for CAN communication function checking. Further, we were able to develop application software that, thanks to software platformization, is able flexibly to accommodate other communication protocols widely used in the automobile industry (such as LIN and FlexRay), as well as new communication protocols. In the future, the proportion of vehicle system development that is taken up with communication design is certain to increase. Coupled with that, there are rising market demands for improved functions and ease of use in simulators supporting communication design. Thus it will be indispensable to enhance applications so as to enable the simulators to cope with various types of communication device and permit users to build checking patterns into the simulators easily and as desired. The CRAMAS simulator is being developed by us in our role as an ECU manufacturer, and we aim to keep this product constantly up to speed with market demands.

References

1) ISO/DIS 11898 Part 1, Part 2
2) CAN2.0B (Bosch CAN specification)
3) Fukazawa et al, "Development of PC-based HIL Simulator "CRAMAS 2001", FUJITSU TEN Technical Report Vol. 37 (2001)

<Trademarks>

Profiles of Writers

Junji Takahashi

Entered the company in 1984. Since then, has been involved in the development of electronic devices for automobiles. Currently the General Manager (Special Project) of the Software Component Engineering Department of Solution Business Division, Business Division Group.

Takeshi Yamasaki

Entered the company in 1983. Since then, has been involved in the development of ECU for engine and body control. Previously assigned to positions in administrative engineering, and currently the Manager of the CRAMAS Engineering Department of Solution Business Division, Business Division Group.

Jun Nishii

Entered the company in 1997. Since then, has been involved in the development of development assistance tools, through the development of electronic devices for automobiles and sales engineering. Currently in the CRAMAS Engineering Department of Solution Business Division, Business Division Group.

Kaoru Noumi

Entered the company in 2000. Since then, has been involved in the design of ECU communication software. Currently in the Software Component Engineering Department of Solution Business Division, Business Division Group.

Kazushi Mahana

Entered the company in 2001. Since then, has been involved in the development of engine control systems. Currently in the Software Engineering Department of Solution Business Division, Business Division Group.