J Inf Process Syst, Vol.13, No.2, pp.348~359, April 2017
https://doi.org/10.3745/JIPS.03.0071

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Design of Real-Time CAN Framework Based on Plug and Play Functionality

Sungheo Kim* and Kwang-il Hwang*

Abstract

Nowadays most vehicles are equipped with a variety of electronic devices to improve user convenience as well as its performance itself. In order to efficiently interconnect these devices with each other, Controller Area Network (CAN) is commonly used. However, the CAN requires reconfiguration of the entire network when a new device, which is capable of supporting both of transmission and reception of data, is added to the existing network. In addition, since CAN is based on the collision avoidance using address priority, it is difficult that a new node is assigned high priority and eventually it results in transmission delay of the entire network. Therefore, in this paper we propose a new system component, called CAN coordinator, and design a new CAN framework capable of supporting plug and play functionality. Through experiments, we also prove that the proposed framework can improve real-time ability based on plug and play functionality.

Keywords

Controller Area Networks, Plug and Play, Real-Time

1. Introduction

Nowadays vehicles are being evolved as a means of various entertainment space as well as higher safety and convenience by integrating state of the art electric, electronics, and information technologies, and thus the number of electronics control devices is rapidly increased. Particularly, the portion of electronic devices in automobile industry is increasing more and more, and many experts [1] forecast that these electronic technologies will occupy more than 80% of innovative technologies in vehicles. As the electronics devices are increased, to control these devices a number of wires are required, and thus it might make it difficult to design vehicle internals, increase its entire weight, and degrade performance and fuel efficiency as well as the increase of manufacturing costs [2]. Therefore, to address these problems Controller Area Network (CAN) [3] is introduced by Bosch, Germany in 1980s. Since the CAN has a number of advantages of supporting more accurate controls, and improving performance and fuel efficiency of a vehicle by drastically reducing the number of wires in a vehicle, it is widely used in automobile areas. In particular, in vehicle industry high reliability is one of the most important issues, so that it is important that the network should support reliable transmission and reception at appropriate timing without any errors between devices. However, in CAN timing errors can occur

Manuscript received January 13, 2016; accepted January 30, 2016.
Corresponding Author: Kwang-il Hwang (hkwangil@inu.ac.kr)
* Dept. of Embedded Systems Engineering, Incheon National University, Incheon, Korea (shplussh@naver.com, hkwangil@inu.ac.kr)

when messages collide with each other on the Bus. More specifically, the collision avoidance of CAN depends on the identifier priority, so that if more messages with higher priority are generated, some nodes can lose a chance to send data or experience unpredictable delay [4].

Therefore, in this paper we propose a new CAN component, which is capable of managing integrated information of each node on the same Bus, and introduce a new CAN framework which improves real-time ability of conventional CAN as well as Plug and Plug functionality by exploiting dual identifier deployment.

The remainder of this paper is organized as follows: in Section 2, CAN protocol is overviewed briefly and existing researches on CAN delay are investigated in Section 3. The proposed scheme is presented in Section 4. Experimental result is presented in Section 5, and Section 6 provides some concluding remarks.

2. Controller Area Network

CAN protocol is a global standard defined specifying ISO11519-2 (low speed) and ISO11898 (high speed), and initially designed to be applied to vehicles but in recent widely used in various industrial areas due to advantages in higher performance and cost efficiency over other network protocols. According to the data rate applied, CAN specification is divided into ISO 11898-9-3 (low-speed, 50 kbps to 125 kbps) and ISO11898-2 (high-speed, 50 kbps to 1 Mbps). It also provides non-destructive bitwise arbitration (NDBA) and broadcasting. In addition, CAN protocol follows OSI 7 layer, and in particular used for two lower layers, data link and physical layer.

Fig. 1 shows basic CAN frame architecture of CAN 2.0A and CAN 2.0B. A CAN frame consists of 6 fields: arbitration field, control field, data filed, CRC filed, ACK filed, and end of frame field, and in particular arbitration filed shows one of the biggest difference between CAN 2.0A and CAN 2.0B. CAN 2.0A arbitration field consists of 11 bits, and CAN 2.0B has total 29 bits including basic 11 bits and extended 18 bits. The basic 11 ID bit is necessary to support backward compatibility of CAN 2.0B with CAN 2.0A. In addition, through data filed which has a variable length of data up to 64 bits, each device can load data on the Bus [5].

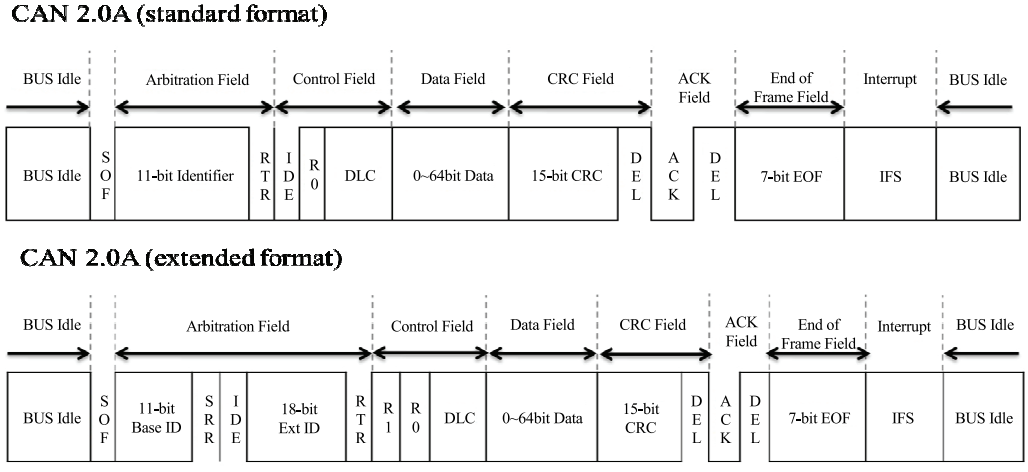


Fig. 1. Controller Area Network (CAN) frame architecture.

3. Related Work

3.1 CAN Transmission Delay

Even though CAN has a number of advantages, it also has some critical limitations. In particular, if more than two devices send data to CAN bus at the same time, the data will collide with each other. Of course, to solve this collision problem, CAN standard uses identifier priority based arbitration. However, during the arbitration, some data which should be transmitted in a specific cannot complete transmission if higher priority nodes send data at that time. To address this transmission delay problem, a number of researches [6-9] are conducted and most of them focused on scheduling messages. In particular, these are also categorized [6] into static scheduling, dynamic scheduling, and distributed message allocation.

Static scheduling, which is an approach widely used in conventional CAN systems, assigns priority according to the message criticality, and then allocates ID based on the priority. Therefore, the transmission sequence follows priority initially assigned. On the other hand, in dynamic scheduling, each device operates normally, but the ID of each message can be changed according to the needs or situation. That is, the priority can be dynamically varied according to events.

3.2 Real-Time Scheduling

Deadline-monotonic scheduling (DMS), which is a representative of static scheduling methods, is a variation of rate-monotonic scheduling (RMS) for CAN system. RMS determines priority according to the length of a period. More specifically, the priority of a message having shorter period precedes messages having longer period. The messages assigned priorities using RMS transmitted sequentially. In general RMS assumes that all the transmission period is the same as a deadline, but not always. Therefore, DMS assigns priority based on not a period but deadline time of each message. Even though this static scheduling methods are simple but it also requires global synchronization of all the nodes in a network. Furthermore, in case of high traffic situation, messages having lower priority might experience unpredictable transmission delay or transmission failures.

Earliest Deadline First (EDF), which is one of the representatives of dynamic scheduling methods, gives higher priority to the message that is the closest to deadline. Unlike static scheduling method, on the completion of a transmission of a message, all the nodes check deadline of its own messages, and then the message that has a shortest deadline has the highest priority at the next round. Even though dynamic EDF method can enhance some problems of DMF, a research [7] revealed that EDF requires a huge overhead to reschedule all messages every time communication is performed. Furthermore, since all the nodes in the network have to check how close it is to the deadline and difference of the remaining time from other messages, it is not feasible to be implemented in small embedded systems.

3.3 Distributed Message Allocation

Distributed message allocation is a method enabling to guarantee real-time QoS performance by increasing the amount of message transmission and its operation is based on Microcontroller with more than two CAN controllers and dual channels using dual transceivers. However, in case of message transmission using dual channels, without fair transmission distribution on dual channels, network traffic can be forced to only one channel. In order to address the problem, Kim et al. [8] proposed a

message allocation method about how to determine an appropriate channel out of two channels. However, the scheme has also a critical limitation of which message will be unfairly allocated when errors in traffic prediction mode, which is used to continuously monitor network traffics, occur. Kim et al. [9] also introduced a distributed message allocation method in which all the nodes in a network predict traffic and based on the prediction messages are fairly allocated.

These distributed message allocation methods basically assumes to utilize dual channels based on dual transceivers, but it also requires additional cost to construct a CAN network. In worst case, utilizing the dual channels (transceivers) will require more than twice cost as much as a single transceiver system. The authors [10,11] also introduced a real-time Plug-and-Plug CAN system, but our work presents more elaborate, enhanced framework based on a CAN coordinator.

4. Real-Time CAN Framework

As a matter of fact, CAN was officially published at the Society of Automotive Engineers (SAE) in Detroit, USA, 1986, and after that it was published by the International Standard Organization (ISO) in 1993. Since then, a number of enhanced versions have been revised, and in recent the CAN system is settled as a safe network system to be applied to most vehicles. Therefore, it is not easy to add or modify some functions of CAN, which is in mass production, and to do so it is required to verify and prove the overall reliability with respect to the modification.

In that sense, our design goal is to develop a real-time CAN framework to be naturally added on the CAN standard. It means that the new system should be able to solve some problems supporting a compatibility with general CAN systems. Therefore, in order to address some problems of CAN system depicted in the previous Section, we propose a real-time CAN framework based on a new component, called CAN coordinator, and while it does not require any changes in conventional CAN system, it works well with other standard CAN systems.

In our framework, we first define a CAN coordinator and new mode, a registration mode for CAN coordinator to arbitrate message periods. The registration mode is used for the CAN coordinator to manage messages, when a new node joins the CAN network that is already running, and in particular, it enables Plug-and-Play functionality which can operates dynamically without any upgrade, reconfiguration, or modification. After completion of registration mode, the framework operates as a normal CAN network. In the following subsections, we present the design and implementation details of Plug-and-Play functionality.

4.1 System Architecture

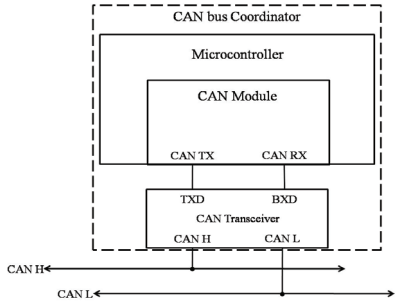


Fig. 2. Coordinator architecture.

4.2 CAN Coordinator Architecture

As shown in Fig. 2, the coordinator in composed of microcontroller and CAN transceiver. Actually, since the coordinator is a logical unit, H/W components as the same as general CAN devices. For CAN ID, we employed CAN 2.0A 11bit ID, which is widely used in most vehicles, as well as compatible with both of CAN 2.0A 11 bits and CAN 2.0B 29 bits. In addition, the data rates are based on 100 kbps which is dominantly used in body electric system in a vehicle.

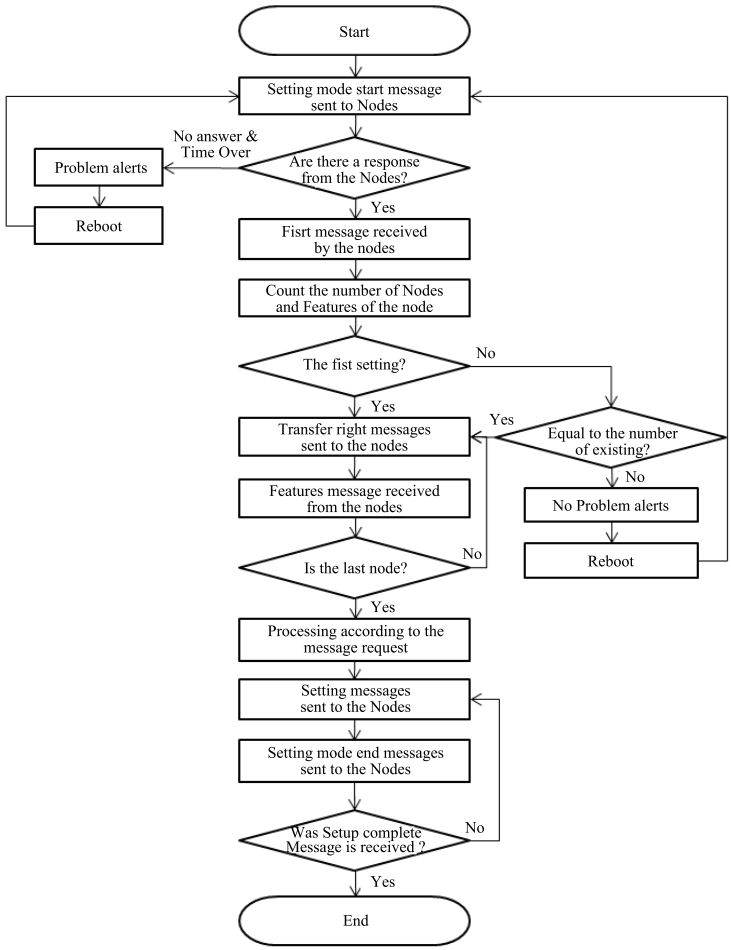


Fig. 3. Controller Area Network coordinator operation flow.

4.3 Device Type Definition

4.3.1 Coordinator

Fig. 3 illustrates a CAN coordinator operation flow. After power-on, the coordinator goes to the registration mode, and sends initiation message on the CAN Bus. During the time, it waits for responses from devices connected on the Bus for a given duration, and monitors data messages having ID between 701 and 7 FF. If there is no response during the time, CAN coordinator terminates the registration mode, and resends no response messages on the Bus for each device to check the system. If

each node replies with the initiation message of coordinator, the CAN coordinator counts the number of nodes and the number of messages of each node. If there is no error in the messages, the coordinator sends a second message, which is to notify nodes transmission sequence assigned, and then waits for reply. The CAN coordinator, which receives the responses from devices stores several information about device role, operation, location, urgent message, and ID deployment request. Based on the ID deployment request, CAN coordinator assigns ID according to either automatic deployment or manual deployment. In the case that conventional ID deployment is used, it also check redeployment request for performance enhancement or revisions, and if so, the coordinator reassigns ID according the requests. On the completion of ID deployment, the coordinator checks urgent messages. It is important to note that the urgent message can have dual IDs with respect to a single task.

After finishing ID deployment for role and operation of messages, the coordinator sends a set-up message in which operation information, and data ID in normal mode are defined. Once receipt of set-up message, each device sends back registration termination message on the Bus. It is important to notice that the CAN coordinator does not mediate normal mode CAN operation after registration mode until next power on or reset. Each node is now ready to communicate with each other according to data ID, role, and operation method which are configured by the coordinator.

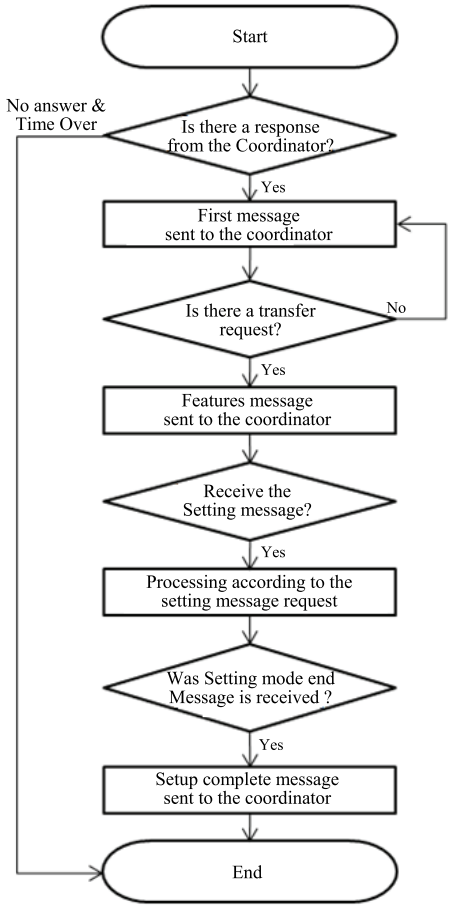


Fig. 4. Device operation flow.

4.3.2 CAN device

Fig. 4 illustrates the operation flow of a CAN device. After power-on, each node goes to registration mode, and waits for registration initiation message from coordinator. If a device cannot receive the initiation message from a coordinator, in order to avoid system faults, it changes its mode to normal mode. Otherwise, after initiation response transmission, each node waits for message sequence having several functional descriptors. The nodes that receive the sequence message from coordinator, transmits second message according to the given sequence. After transmitting configuration and registration completion message, each node changes its mode to normal mode, and operates as in conventional CAN system.

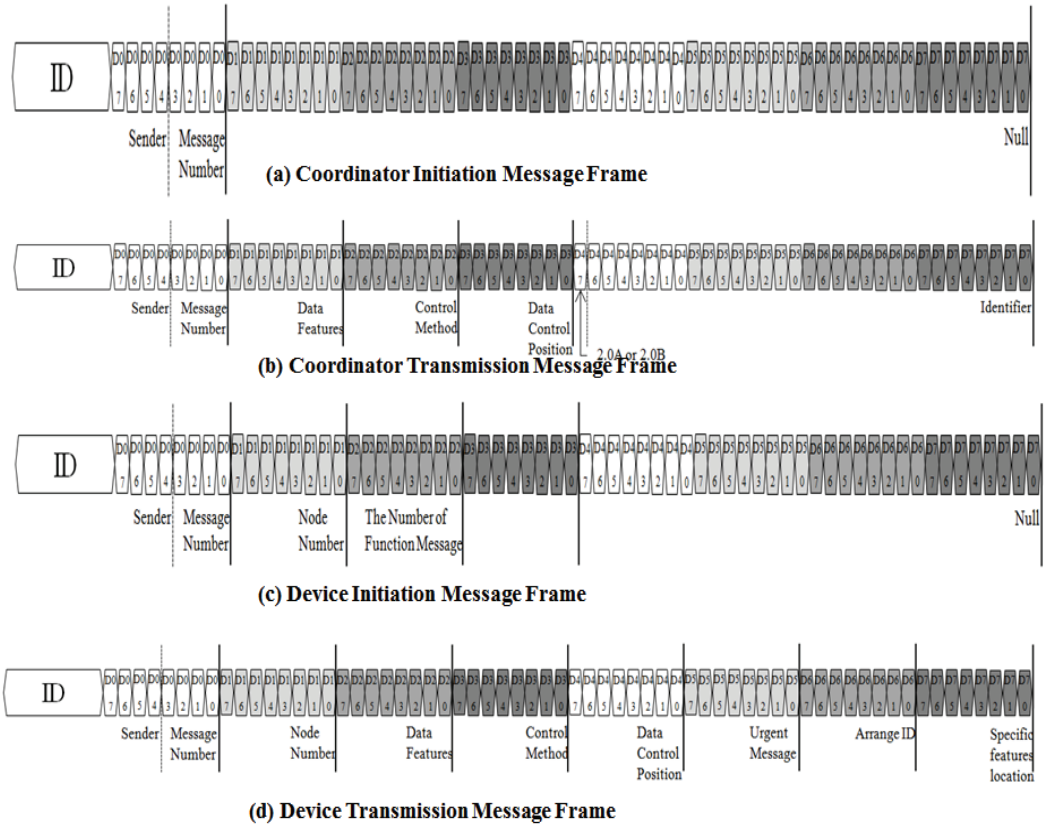


Fig. 5. Examples of frame structures according to different functions.

4.4 Frame Structure

In order to accommodate the functionalities newly defined in this paper, we redefine the frame structure based on the new ID assignment criteria as shown in Fig. 5. As already mentioned in the previous subsection, since we utilize CAN 2.0A 11 bits, the range of ID is 000 to 7 FF as shown in Table 1. Therefore, the section is used for urgent message deployment, normal message deployment, registration messages during registration process.

Table 1. ID range according to the message functions

ID	Message description
000–0 FF	Urgent message section
100–6 FF	Normal message section
700–7 FF	Registration message section

* Urgent message: The messages have the highest priority over other messages. Furthermore, it also has least transmission delay due to high priority. For example, AIR BAG system.
* Normal message: The messages are used in normal CAN operation ranging from 100 to 6 FF.
* Registration message: In the proposed framework, after power-on, all the nodes enter registration mode, the messages for the registration mode are assigned ranging from 701 to 7 FF.

5. Experimental Results

5.1 Experimental Environments

In order to test the feasibility and performance of the proposed CAN framework, we developed new CAN systems including CAN coordinator and devices. As shown in Fig. 6, hardware architecture of CAN device and coordinator is the same but they operate differently according to the software. Hardware prototype is composed of Microcontroller [12] integrated with CAN transceiver, and simple IO (Buzzer and push switch).

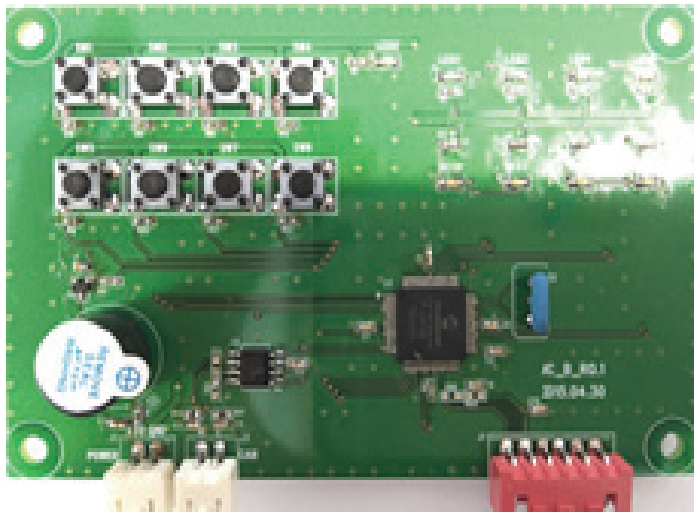
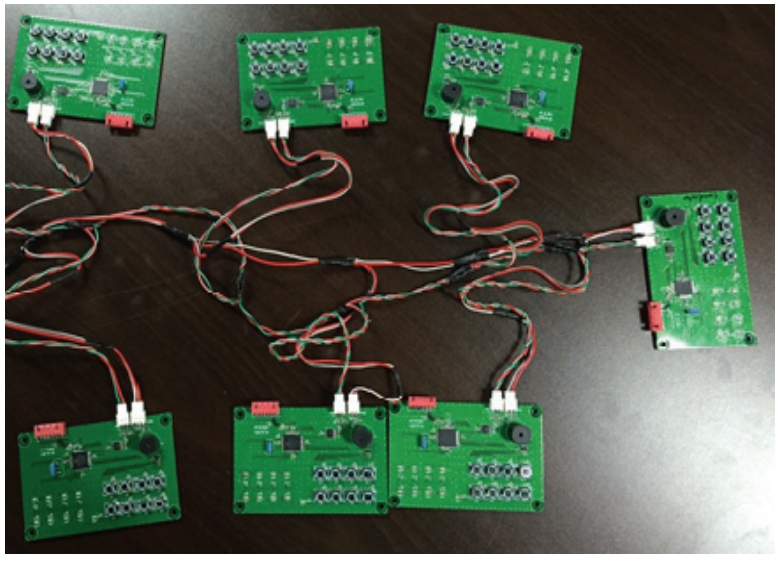
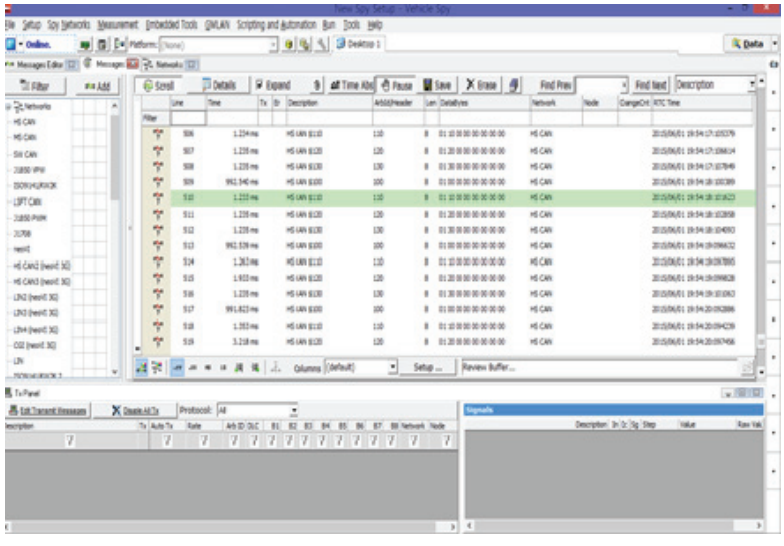


Fig. 6. Controller Area Network test node.

To construct a test-bed, we used 1 coordinator and 6 devices. In addition, each device regardless of device type, each device can generates messages based on on-off signal, and to verify and monitor CAN messages on the Bus in real-time, we used Vehicle Spy [13] with professional instrument, neoVI (Intrepid Control System Inc., Madison Heights, MI, USA) as shown in Fig. 7.



(a)



(b)

Fig. 7. Test-bed (a) and monitoring software (b).

5.2 Performance Evaluation

First we observed transmission delay of the proposed system. As mentioned previously, we used CAN 2.0A ID system, and allocate ID ranging 100–6 FF by separating ID region according to data functions. Each device transmits a message at the data rates of 100 kbps every 1000 ms at the same time. Therefore, several messages are collided with each other, and due to collision arbitration mechanism, some messages that have lower priorities are delayed by 1.2–1.7 ms. As shown in Fig. 8, when 6 collisions occur in the network, the delay is about 6 ms. However, when our algorithm is applied, the delay is negligible compared to normal CAN system. That is because to cope well with urgent messages, we allocated urgent message region ranging from 000 to 0 FF. Therefore, an urgent message can get

dynamically high priority, so that urgent messages will experience less transmission delay. On the other hand, transmission delay in the normal CAN system is increasing proportional to the number of TX data.

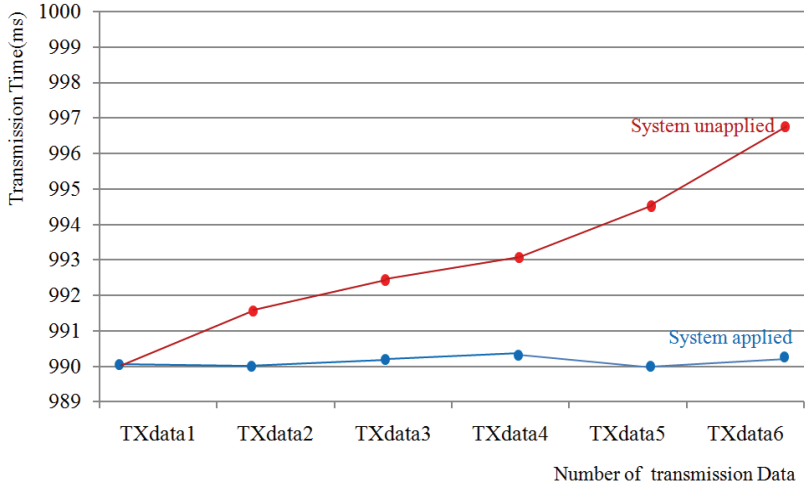


Fig. 8. Transmission delay (proposed CAN framework vs. conventional system).

In addition, we also tested the feasibility and compatibility of the proposed framework with conventional CAN systems. To verify our framework, we experimented with Plug and Play functions, in which 1 coordinator, 4 nodes. They are already installed and operate, and during runtime, a new node is attached to the network. Fig. 9 shows the summary of plug and play operation verified by the CAN monitor instrument.

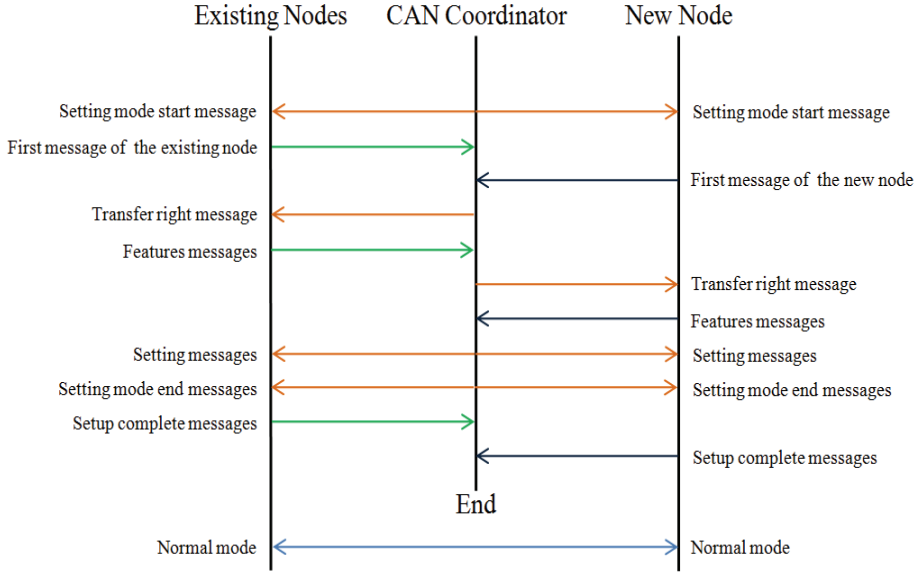


Fig. 9. Plug and Play function verification.

6. Concluding Remarks

In this paper we proposed a new system component, called CAN coordinator, and design a new CAN framework capable of supporting plug and play functionality. In the proposed framework, we tried to solve two inherent problems of CAN systems. First, our framework enhanced unpredictable transmission delay, in particular, with respect to real-time messages that should be transmitted within deadline, and which enables to get higher priorities by newly defining urgent messages using the new ID region. The experiments showed that in the case that a number of nodes transmit messages at the same time, while urgent messages can send earlier, transmission delay of normal messages are relatively increased. However, in real system environment, since the time gap between CAN messages exists, the possibility that normal messages are seriously delayed is very low.

Next goal of our work is to support Plug and Play function. To achieve this, we defined a new device, a CAN coordinator, which can manage the information, such as message function, operation and location, and reallocate ID. In addition, we also redefine a new mode, registration mode for a coordinator to dynamically recognize and manage a new node. Our experimental results also proved that a new node can enter the existing network regardless of operation of the other nodes on the Bus.

Therefore, we expect that the proposed framework will be used as a concrete guideline to enhance CAN system and network, and we are also trying to design flexible CAN system to be used in more various application fields beyond vehicle areas.

Acknowledgement

This work was supported by the Incheon National University Research Grant in 2016.

References

[1] D. Marsh, "Network protocols compete for highway supremacy," 2003 [Online]. Available: http://www.edn.com/design/communications-networking/4331809/Network-protocols-compete-for-highway-supremacy.

[2] S. Lee, M. H. Kim, and K. C. Lee, "Survey on in-vehicle network system researches," *Journal of the Korean Society of Precision Engineering*, vol. 23, no. 9, pp. 7-14, 2006.

[3] Bosch, *CAN Specification Version 2.0*. Stuttgart: Bosch GmbH, 1991.

[4] K. N. Ha, M. H. Kim, K. C. Lee, and S. Lee, "performance evaluation of network protocol for automated transfer crane system," *Journal of Control, Automation, and Systems Engineering*, vol. 11, no. 8, pp. 709-716, 2005.

[5] W. E. Seitz, "Controller area network in embedded machine control," 2004 [Online]. Available: http://www.techonline.com/electrical-engineers/education-training/tech-papers/4125494/Controller-Area-Network-in-mbedded-Machine-Control.

[6] J. P. Lehozky and L. Sha, "Performance of real-time bus scheduling algorithm," *ACM SIGMETRICS Performance Evaluation Review*, vol. 14, no. 1, pp. 44-53, 1986.

[7] G. C. Buttazzo, "Rate monotonic vs. EDF: judgment day," *Real-Time Systems*, vol. 29, no. 1, pp. 5-26, 2005.

[8] M. H. Kim, K. N. Ha, K. C. Lee, and S. Lee, "Traffic prediction of CAN network system with dual communication channels," in *Proceedings of International Conference on Control, Automation and Systems*, Seoul, Korea, 2008, pp. 397-400.

[9] M. H. Kim, J. G. Lee, S. Lee, and K. C. Lee, "A study on distributed message allocation method of CAN system with dual communication channels," *Journal of Institute of Control, Robotics and Systems*, vol. 16, no. 10, pp. 1018-1023, 2010.

[10] S. H. Kim, "Design and implementation of CAN communication system capable of supporting real-time plug and play," M.S. thesis, Incheon National University, Incheon, 2015.

[11] S. H. Kim and K. Hwang, "Design and implementation of CAN communication system capable of supporting real-time plug and play," in *Proceedings of the Korea Information Processing Society (KIPS) Fall Conference*, Jeju, Korea, 2015.

[12] Microchip PIC18F66K80 datasheet [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/39977f.pdf.

[13] Vehicle Spy [Online]. Available: http://www.intrepidcs.com/products/software/vehicle-spy-professional/.

Sungheo Kim

He received B.S. degree in the National Institute for Lifelong Education (NILE), Korea, 2012, and M.S. degree in Incheon National University, Korea, 2015. He is currently working in Garin System Corp. Korea.

Kwang-il Hwang http://orcid.org/0000-0002-4196-4725

He is an associate professor, in the Department of Embedded Systems Engineering, Incheon National University, Korea. He has received M.S. and Ph.D. in Electronics and Computer Engineering from Korea University, Seoul, Korea, 2004 and 2007, respectively. He is a number of international conferences hosted by KIPS CSWRG. He is also a member of IEEE, KICS, KMMS, KIPS, and KISS.