

Message Packing Algorithm for CAN-Based Legacy Control Systems Mixed with CAN and FlexRay

2010-01-0685

Published
04/12/2010Tasuku Ishigooka and Fumio Narisawa
Hitachi Ltd.

Copyright © 2010 SAE International

ABSTRACT

Hard real-time systems such as automotive control systems have to guarantee that strict deadlines are met for applications. Recent automotive control systems have been network systems that have combined event-triggered with time-triggered networks, i.e., Controller Area Network (CAN) and FlexRay. A CAN-FlexRay gateway has to execute real-time message transfers from CAN to FlexRay and from FlexRay to CAN to guarantee that communication deadlines are met. Most gateways in the automotive control systems select messages according to the priority of the messages and pack them into frames. However, when many events of same kinds occur within the short period, the gateway cannot guarantee that communication deadlines for time-triggered and first event-triggered messages will be met because many event-triggered messages prevent time-triggered messages from being packed into frame. This paper describes a novel real-time message packing algorithm that prioritizes transmissions of time-triggered messages and first event-triggered messages to guarantee that communication deadlines for them are met. The message packing algorithm statically establishes various clusters of both communication messages and FlexRay frames and preferentially packs a message into a specified frame according to relationship between clusters of them. We evaluated the proposed message packing algorithm by developing a communication middleware with the algorithm for CAN-based legacy control systems and a CAN-FlexRay gateway including the middleware and found that the algorithm guarantees communication deadlines for time-triggered messages and first event-triggered messages are met in spite of packing into fewer FlexRay frames by establishing the cluster which is defined to pack their messages certainly.

INTRODUCTION

Typical automotive control systems have been structured by subsystems, which consisted of electronic control units (ECUs) and in-vehicle networks such as CAN [1], TTP [2], FlexRay [3], LIN [4] and MOST [5]. Most automotive control systems utilize CAN widely because of low-cost of a CAN controller and sufficient performance for automotive control applications such as powertrain control applications. ECUs in the systems transmit messages that are control data or events to one another directly or via CAN-CAN gateways.

Recently, the number of the messages in the automotive control systems has gradually been increasing. Thus, we focused on a high speed in-vehicle network, FlexRay. FlexRay provides the fastest speed of 10Mbit/s and is much faster than CAN that provides the fastest speed of 1Mbit/s. We assumed that automotive control systems adopt FlexRay buses as an alternative to high-traffic-volume CAN buses. Therefore, the in-vehicle networks of the automotive control systems combine CAN with FlexRay. The systems need CAN-FlexRay gateways that transfer messages from CAN to FlexRay and from FlexRay to CAN. The gateway has the three main requirements.

1. Reuse of CAN-based Legacy Applications: We need to develop CAN-FlexRay gateways, where the CAN-based legacy applications transmit messages to FlexRay by calling a CAN driver API without having to modify them, to decrease the cost of development.
2. Real-time Message Transfer: The CAN-FlexRay gateways have to guarantee that messages meet communication deadlines, i.e., the time limits for transmitting the messages.
3. Efficient Message Transfer: The CAN-FlexRay gateways have to transfer the messages by utilizing fewer FlexRay

frames because FlexRay has limited network bandwidth although FlexRay is much faster speed than CAN.

Most automotive control systems are hard real-time systems. There are two approaches to designing the real-time systems, i.e., by using event-triggered and time-triggered architectures [6]. The event-triggered architecture consists of program modules that immediately respond to events such as messages or external inputs. The program modules are invoked by event-triggered task that are activated aperiodically and they transmit event-triggered messages that are transmitted to networks aperiodically. For example, the event-triggered messages are event messages such as those on gearshifts and data for diagnosis. The time-triggered architecture consists of program modules that periodically respond to events or external inputs. The program modules are invoked by time-triggered tasks that are activated periodically and they transmit time-triggered messages that are sent to networks periodically. For example, the time-triggered messages are state messages such as calculated physical values and sensor values. The automotive control systems are combined event-triggered architecture with time-triggered architecture, i.e., transmitting event-triggered and time-triggered messages.

Several middleware that transmit event-triggered and time-triggered messages to time-triggered networks have been presented [7][8]. The middleware is suitable for the automotive control systems combined with these architectures. However, we have to modify CAN-based legacy applications to utilize the middleware.

There are approaches to reusing CAN-based legacy applications by utilizing software platforms such as automotive open system architecture (AUTOSAR) [9] and middleware [10]. Although AUTOSAR is effective to reuse the applications that are developed according to AUTOSAR, we have to modify applications, which are not developed according to AUTOSAR, to apply them to AUTOSAR [11]. The middleware enables CAN-based legacy applications to be reused by modifying itself without having to modify the application. However, it has to execute message packing processing to guarantee communication deadlines are met.

There are approaches to guaranteeing deadlines are met, i.e., by scheduling both frames and tasks to handle messages [12][13][14], by scheduling messages to be packed into frames according to priority of the messages [15][16], and by packing messages into frames according to statically defined relationship between the messages and frames [17]. The first approach is effective to guarantee that end-to-end deadlines are met and the second approach is used widely by real-time systems such as event-triggered architecture. However, these approaches cannot guarantee that the deadlines for all time-triggered messages are met because it is difficult to estimate the number of event-triggered messages in the automotive control systems, i.e., the number of messages in the

automotive control system varies due to variable transmission timing executed by event-triggered tasks, ECU breakdowns, or bugs in the software of ECUs. The third approach is suitable for time-triggered network and it can guarantee communication deadlines are met to pack messages according to the message packing algorithm. However, if the transmission period for most messages is longer than one of time-triggered networks, the ECUs with the approach transmit many empty frames. Consequently, these approaches do not fulfill the second or third requirements.

We propose a novel message packing algorithm for the automotive control systems and the middleware to fulfill the three requirements. We also evaluate the proposed algorithm by developing the middleware with the algorithm and a CAN-FlexRay gateway without having to modify the CAN-based legacy gateway application by utilizing the middleware.

This paper is structured as follows. The first part describes the architecture of our target automotive control systems, communication messages, the concepts behind the CAN-FlexRay gateways and the challenges involved in achieving the first requirement. The second part presents a novel message packing algorithm to achieve the second and third requirements. The third part describes message transfer processing of the CAN-FlexRay gateway and the fourth part discusses the experimental results. Fifth part concludes this paper and mentions future work.

NETWORK SYSTEMS INCLUDING GATEWAY

ARCHITECTURE

Figure 1 shows our target automotive control system. The systems consist of ECUs connected to a CAN bus and three CAN-FlexRay gateway (GW) ECUs connected to both a CAN bus and a FlexRay bus. The ECUs operate sensors and/or actuators and transmit messages to a CAN bus. The CAN-FlexRay gateway ECUs transfer messages from a CAN bus to a FlexRay bus, and from a CAN bus to a FlexRay bus.

CAN is based on event-triggered protocol where messages is transmitted immediately. FlexRay is a hybrid protocol that consists of static and dynamic segments. The FlexRay controller transmits frames periodically in the static segment according to a defined scheduling based on time division multiple access (TDMA) such as time-triggered protocol and it transmits frames aperiodically in the dynamic segment according to the transmission requests of applications.

The ECUs in this automotive control system transmit messages during the static segment to ensure reliable communication. The static segment can guarantee that communication deadlines are met to transmit frames

determinately. On the other hand, the dynamic segment does not transmit frames determinately. For example, when it takes over a defined time of the dynamic segment, frames of the dynamic segment is not transmitted[3]. Consequently, the dynamic segment is not suitable for achieving communication systems that are extremely reliable.

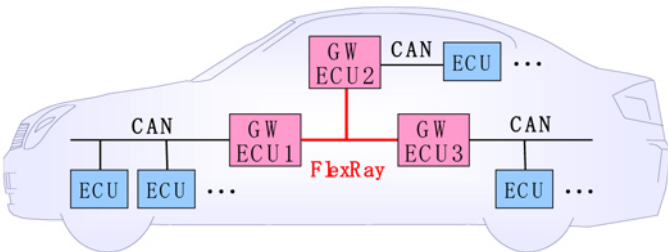
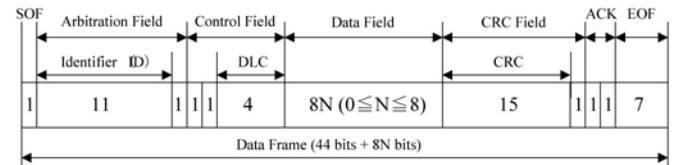


Figure 1. Target Automotive Control System Combined with Different In-vehicle Networks

COMMUNICATION MESSAGE

The ECUs transmit event-triggered and time-triggered messages to one another directly or via CAN-FlexRay gateways to execute collaborative processing such as active cruise control (ACC). [Figure 2\(a\)](#) shows a CAN data frame, which consists of a start of frame (SOF), an arbitration field including identifier (ID), a control field including a data length code (DLC), a data field, a cyclic redundancy check (CRC) field, an acknowledgement (ACK) and an end of frame (EOF). CAN ID is utilized by a CAN controller as priority of messages in a CAN bus.

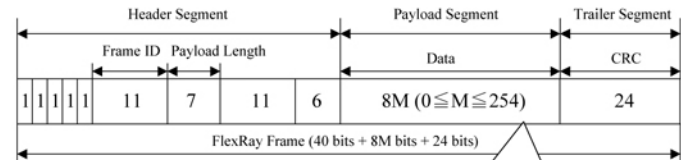
[Figure 2\(b\)](#) outlines a gateway message (GM) that the CAN-FlexRay gateway can transfer from a CAN bus to a FlexRay bus and from a FlexRay bus to a CAN bus. The gateway message consists of an identifier, the data target automotive control systems. However, we decided to fix the data field sizes of CAN messages to 8 bytes, i.e., all gateway messages were 10 bytes, to accomplish efficient message transfer processing in the gateway.



(a) CAN Data Frame



(b) Gateway Message (GM)



(c) FlexRay Data Frame

Figure 2. Communication Message

[Figure 2\(c\)](#) shows a FlexRay data frame, which consists of a header segment, a payload segment, and a trailer segment, in the target automotive control systems. The data of the payload segment that stores messages is longer than that in a CAN data frame. Therefore, we divided the payload segment into several small frames that are called sub-frames. The sub-frame stores a gateway message.

A CAN-FlexRay gateway has to extract the specified information from CAN messages and transmit FlexRay frames including gateway messages to the FlexRay bus.

CONCEPT OF GATEWAY

We develop a communication middleware that can handle FlexRay as CAN, i.e., an event-triggered (ET) FlexRay middleware, and CAN-FlexRay gateway ECU by utilizing the middleware.

[Figure 3](#) outlines the software architecture for a CAN-FlexRay gateway ECU. The gateway consists of a CAN-based legacy gateway application, an ET-FlexRay middleware, a real-time operating system (RTOS), a CAN driver, and a FlexRay driver. The ET-FlexRay middleware has an ET-FlexRay adapter that translates the CAN driver call to the ET-FlexRay interface call, an ET-FlexRay interface that stores transmission data requested by the application to buffers/queues of the middleware, a packing scheduler that executes message packing processing, and a packet translator transmits packed data as FlexRay frames by utilizing the FlexRay driver.