

Design and Performance Analysis of 10 Mbit/s CAN

Ryo Kurachi, Hiroaki Takada and Shigeharu Teshima

Center for Embedded Computing Systems, Nagoya University

Yukihiro Miyashita

AutoNetworks Technologies, Ltd.

Copyright © 2009 SAE International

ABSTRACT

In this paper, we propose a new automotive network protocol which can transmit data at a speed of 10 Mbit/s. The new automotive protocol "10 Mbit/s CAN" is based on two changes to CAN; a star topology, in which nodes are physically connected using a multi-port gateway, and a new collision resolution algorithm which guarantees delivery of a message within a finite period of time. According to our analysis using a simulator, the "10 Mbit/s CAN" protocol demonstrated several advantages over throughput performance of a conventional CAN, including a maximum possible data transmission speed, high scalability, and reduction of priority inversion. The implementation considerations of "10 Mbit/s CAN" are also reviewed briefly.

INTRODUCTION

Recently, there has been a trend in the automotive industry to increase the number of ECUs per vehicle to provide many applications. For example, in a modern car, there are approximately 100 ECUs connected together throughout the vehicle to perform functions such as engine management, assisted braking and entertainment. In future cars, fault-tolerant applications (e.g., X-by-wire [1]) using a next-generation protocol will likely be standard, hence improving the network capacity and guaranteeing the transmission of all safety related messages are expected.

Many automotive manufacturers have shown interest in next-generation protocols such as FlexRay [2]. For reliable communication, a few next-generation protocols

are based essentially on a time-triggered approach, which requires reconfigurations of the whole system for the addition of new nodes. Furthermore, as the development period of automotive systems shortens, there is a need to make changes to the original design quickly and safely. For example, it should be possible to add new functions to ECU without affecting sub-systems that require a high level of safety.

In this paper, we propose the protocol "10 Mbit/s CAN", which is based on an event-triggered approach and whose throughput is ten times as much as a conventional CAN [3]. Because "10 Mbit/s CAN" enables messages to be sent whenever required, a schedulability analysis is needed for reliable communication. To guarantee that real-time requirements are fulfilled, this paper proposes a worst-case analysis for "10 Mbit/s CAN".

It should be noted that a conventional CAN is called CAN and our proposed protocol is called 10 Mbit/s CAN in following sections.

PROBLEM

THEORETICAL UPPER BOUND OF CAN

CAN has to ensure that every node reads the same bit value from the bus. If two nodes send messages at the same time, message arbitration on the bus is handled by a bitwise arbitration mechanism. Thus, the propagation delay between two nodes must not exceed a certain value. One of the prerequisite for this arbitration mechanism is the bit rate on the bus must be below a

certain limit. In short, arbitration only works when the bit rate is sufficiently low so that propagation delays between any two nodes of the network are much smaller than one bit time. A similar issue was found for the acknowledgement sequence.

In addition, a limitation of bus topology is the number of nodes connected. As the data transfer rate increase, the number of the nodes decreases accordingly in the network, because of ringing problems caused by reflections and noise. This undesirable effect also occurs in high-speed protocol such as FlexRay.

To permit a much higher bus bandwidth of CAN, we have to change the arbitration process and acknowledge sequence to overcome propagation delays.

PROPOSED PROTOCOL

This section describes elements of the 10Mbit/s CAN protocol and characteristics of a system model that are needed to formulate the worst-case response analysis.

The 10Mbit/s CAN physical layer supports two states termed dominant ("0") and recessive ("1") as similar to CAN.

The message transmission of 10 Mbit/s CAN is divided into two cases according to channel conditions. When an idle bus is detected before attempting to transmit, a node starts to transmit; when the bus is busy (the controller state is transmission/reception), message transmission is determined by alternative transmission scheduling (sender→receiver→sender...). However, in the idle condition, if two or more nodes start to transmit at the same time, it is possible for each frame to collide on the bus. Therefore, a method (i.e., a collision resolution algorithm) is needed to prevent the collision. 10Mbit/s CAN has three major changes from CAN. First, 10Mbit/s CAN uses a star topology to reduce the effects of ringing problem. Second, Acknowledge sequence is changed to overcome the effects of propagation delays. Finally, the protocol uses a new collision resolution algorithm with asymmetrical waiting time. These are described in more detail below.

A) Changing topology: a star topology

To overcome the effects of propagation delays, each ECU is physically connected via a multi-port gateway. This arrangement, which generally called a point-to-point link, is a way to reduce the ringing problem.

B) Acknowledgement sequence

In order to obtain reliable communication, an acknowledgement sequence uses a straightforward

algorithm, such as the stop-and-wait algorithm. (After transmitting a message, the sender waits for an acknowledgement before transmitting the next frame. If the acknowledgement does not arrive within a certain period of time, the sender times out and retransmits the original message.) To use the stop-and-wait algorithm, we change Data frame to include acknowledgement as information. (see APPENDIX A)

By applying this protocol on the receiving side, the time interval between receiving a message and transmitting an acknowledgement is a one-time $EOF + IFS$ for each node, where EOF is the period of seven bit times and IFS is the period of three bit times on the bus.

On the sending side, for every transmission, the sender waits for an acknowledgement. This waiting time is determined statically by implementing different prioritized scheduling options. In the first transmission attempt, the sender waits for receiving the SOF of response message. If the response of the first transmission is lost, the retransmission starts after waiting for a finite time. The determination of priority is based on the assumption that prioritizes retransmission of the gateway over that of the ECU. For this algorithm, no failures occur in the retransmission.

C) Collision resolution algorithm

To accomplish a fast recovery after collision, proposed algorithm uses a configuration parameter of propagation delays, which is estimated from actual delays between ECU and gateway. As an illustration, consider the timing sequence shown in Figure 1. In order for the response frame to be received after transmission, the sender must monitor the bus for the "dominant" bit to detect the SOF of the response frame. Since the sender completes transmission of the CRC Delimiter at time $t = 0$, the monitoring result is divided into three periods:

- $t < 7$: Detection of the dominant bit after the CRC Delimiter's transmission until sampling seven bits means a collision.
- $7 \leq t < rtout$: Detection of the dominant bit after the CRC Delimiter's transmission to the wait response time $rtout$ means successful transmission. The wait response time $rtout$ is determined by retransmission priority.
- $t \geq rtout$: No response means a transmission failure, such as a fatal error.

The wait response time $rtout$ is determined by the type of node: a gateway has an $EOF + IFS + 2\tau$ response, and an ECU has $EOF + IFS + 4\tau$, where τ is a estimating propagation time between nodes (including

the actual propagation delay time, $1 < \tau$ bit times). Therefore, transmission must result in a finite time. In this way the transmissions of different nodes can be scheduled back-to-back, with little dead-time between transmissions.

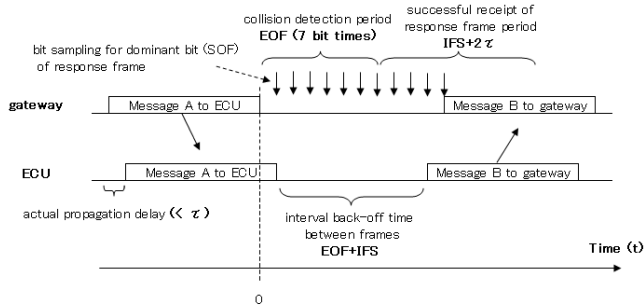


Figure 1 Method to overcome the effects of propagation delays

As a result of collision detection and retransmission rules, the sender can send a message to a receiver according to the determined retransmission period. Figure 2 illustrates the flow of the collision resolution algorithm.

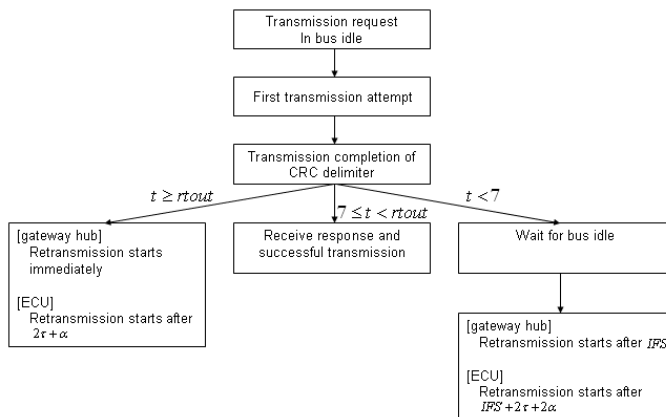


Figure 2 Proposed collision resolution algorithm

In this algorithm, α is the time of clock drift compensator, which is determined by specific implementations.

LAN ARCHITECTURE

Figure 3 illustrates a minimal 10 Mbit/s CAN network with a star topology, which supports concurrent message transmission of all ECUs. The network consists of N ECUs, a wired CAN cable, and a gateway which broadcasts messages from the sender.

An ECU consists of a protocol processor, a number of buffers, and at least one transmit mailbox. The protocol processor controls transmission and reception. Buffers

are designated according to the type of traffic: one is for transmitting data, and the other is targeted for receiving data. For example, Figure 3 shows an ECU that has only one controller which configures only one transmit mailbox and several transmit and receive queues. In transmitting, a message is queued by an ECU application task. If a message is queued by a given task and the mailbox is empty, then the highest priority message in the transmit queue is placed in the mailbox. Then, according to the bus timing, the message can start transmission process on the bus.

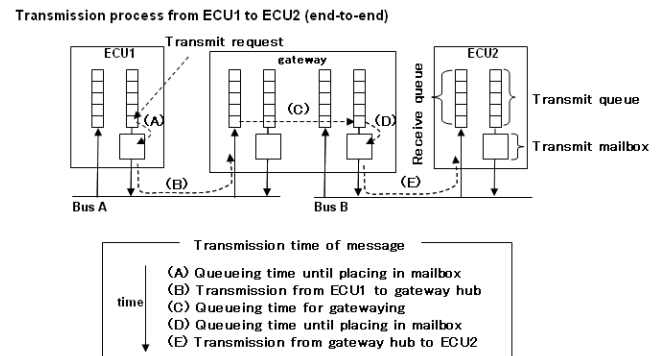


Figure 3 Transmission between ECUs via a gateway

A multi-port gateway consists of a processor and two or more ports configured for the individual device. A gateway port has a number of buffers and at least one transmit mailbox. Figure 3 shows that the gateway has two ports which connected with each bus (Bus A and B). The trigger on the streaming message in a multi-port gateway is a message arriving from an ECU. If the transmit buffer in the gateway is not full, then the message is transferred to this buffer and transmitted in a similar manner to the destination ECU.

Figure 3 also illustrates individual steps during end-to-end message transfer. If ECU1 transmits a message to ECU2, the message transmission time end-to-end is divided into two major steps: one is messages from ECU1 arriving at the gateway (A+B), the other is messages from the gateway arriving at ECU2 (C+D+E). In the former case (A+B transmission), it is possible that scheduling by each ECU can prevent collisions of messages, hence transmission of messages can complete faster than expected worst-case transmission time which is all messages of a ECU starts to transmit at the same time. In the latter case (C+D+E transmission), it is impossible to control the transmission timing between each ECU without a scheduling method such as a synchronized clock; hence, messages concentrated at the gateway may become a serious bottleneck.

The remaining sections of this paper present an evaluation of the proposed 10 Mbit/s CAN protocol, which is evaluated in two approaches. First, we consider the worst-case response analysis for 10 Mbit/s CAN and compare the result of this analysis with the simulation