# Comparison of FieldBus Systems, CAN, TTCAN, FlexRay and LIN in Passenger Vehicles

Steve C. Talbot and Shangping Ren
Illinois Institute of Technology
Chicago, Illinois 60616, USA
{talbste, ren}@iit.edu

*Abstract*— **The Controller Area Network (CAN) architecture was developed for use in automobiles in the 1980's. It corresponds to the physical and datalink layers of the OSI network protocol stack. Manufacturers currently leverage and build upon this architecture to enable on-vehicle sensors, actuators and other commercial electronics to interoperate: communicate between different components, exchange data, and resolve operational dependencies. This paper examines the inner details of CAN and its three competitive networks, namely, TTCAN, FlexRay and LIN.**

***Keywords: automotive networks, autonomous, availability, bit stuffing, certification, CPS, distributed, error management, event-triggered, frames, fault tolerance, global reference time, master-slave, message arbitration, message retransmisison, mono polarity bit sequence, QoS, reconfiguring, real-time, resource management, scheduling, security, time-triggered, ubiquitous, validation, verification***

## I. INTRODUCTION

Since its inception, CAN has been extended to a myriad of embedded electronic application domains, including domestic appliances, military equipment and medical devices [6] to name a few. As an increasing number of embedded electronic components are added to vehicles in order to meet rising consumer demand for product features, the complexity of each embedded component as well as the complexity of groups or sub-systems of components (which intercommunicate) is rising exponentially along with the onset of time. CAN is an example of a complex Cyber Physical System (CPS), or a "deeply" embedded electronic system with locally autonomous or semi-autonomous sensors and actuators, having components which are coordinated across a network. CPS networks often have no server or central processing module, leaving local components to handle real-time-constrained sensor and actuator events and following strict Quality of Service (QoS) requirements [5]. Using the CPS approach in automotive networks reduces cost and helps meet QoS requirements.

CAN was the first automotive network approach to be developed, with LIN developed later for "lightweight" components and FlexRay developed later for high performance components. Very high-speed FlexRay networks (10 MBit/s) or high-speed CAN networks (1 MBit/s) are typically used to implement engine control, transmission control, braking, steering, suspension control, assistance systems, safety systems and diagnostics. Both low-speed (125 kBit/s) CAN networks or LIN networks (up to 10 kBit/s) are typically used to implement control of displays, lighting, alarm systems, air conditioner control, seat and mirror adjustment, power windows, windshield wipers and headlamps [9].

The remainder of this paper is organized into the following sections: Section 2 discusses the relevance of this paper in regards to the current existing literature. Section 3 is a brief discussion about automotive fieldbus systems. Section 4 discusses further about some of the basics for each system. Section 5 discusses "message arbitration" found in CAN, TTCAN and FlexRay. Section 6 discusses the format of message "frames". Section 7 discusses error management. Finally, we conclude in Section 8.

## II. BACKGROUND AND RELATED WORK

CAN was developed in the 1980's to account for the perceived deficiencies of the I2C and D2B networks as used in automobiles at the time. This development resulted in the production of a set of international standardized documents describing CAN in the form of ISO 11898-1 through ISO 11898-5 (physical, high-speed, low-speed fault-tolerant / LS FT CAN, time-triggered / TTCAN, miscellaneous, respectively) and in the form of ISO-16845 (CAN Conformance Testing) [1, 8]. In addition, a group of manufacturers independent of ISO formed CiA (CAN in Automation) for the purpose of marketing CAN and to develop documentation that exceeds and clarifies the standard CAN ISO documents [1, 8]. These documents include "CiA Draft Standards (DS)" (discussing the physical layer) and "CAN Application Layers" (discussing the software application layer). The company Freescale (Motorola) developed the LIN network, and the document "LIN specification v. 2.0" is its current authoritative document. A consortium of automobile manufacturers developed the FlexRay network, resulting in the "FlexRay Protocol Specification v. 2.1" [1]. These documents are solely concerned with describing their respective domains, with little comparison made to rival networks.

Reference [8] describes the CAN network using the ISO-11898-x documents as a basis, with the purpose of

clarifying these fairly terse CAN specifications for a technical audience that has no prior experience with CAN. Reference [1] is also concerned with educating the technical community about automotive networks, but takes a broader approach by including details about CAN, TTCAN, FlexRay, LIN and other networks in a single volume. However, although the intention of reference [1] is to describe these networks and to compare these networks by including them side-by-side, few direct contrasts are developed between the networks. In addition, none of these documents are much concerned with comparing the QoS attributes between these networks, nor with describing how these networks relate to CPS.

This survey paper attempts to describe an overview of the low-level details of the CAN, TTCAN, FlexRay and LIN automotive fieldbus networks. Direct comparisons are made at the end of each section between these networks, with attention paid specifically to the QoS attributes pertinent to each network. In addition, a table is included following Section 8 to describe how the details of each of these networks relate to the CPS paradigm.

## III. Automotive Fieldbus Systems

Electronic vehicle system networks such as CAN, TTCAN, FlexRay or LIN have many advantages. First, the cost of implementing the cabling for these networks is lower than traditional approaches. These are serial bus systems (all communicating components connected to the bus) which reduce the amount of cabling required over traditional point-to-point networks (all communicating components directly connected to all other communicating components) for the wiring harness of the vehicle and simplify vehicle assembly. In addition, the lower number of plug connections of a serial bus system corresponds to fewer wires and increases dependability and reliability. Second, control is typically distributed to the individual Electronic Control Units (ECUs) locally positioned at the specific control system. Service technicians can plug into the network and access any unit on the network using a common software package, and be capable of communicating with any component in the network (manufactured by any Original Equipment Manufacturer, OEM or vendor). Third, "hard real-time" requirements, parameters which must not deviate from strict limits (real-time braking control, real-time engine control) in order to avoid catastrophic events can be realized by meeting so-called "real-time" operation levels. This means that very fast response times to real-time physical events (less than 100 microseconds, which is unnoticeable to a human observer) are realized. This list of advantages supports the use of deeply embedded control systems in vehicles and is the rationale motivating the switch from mechanical systems to "brake-by-wire" and "drive-by-wire" electronic systems [7, 8].

Conventional fieldbus addressing schemes use both "source" and "destination" addresses. CAN avoids using addresses; instead, it requires that each node transmit its messages to all other nodes in the network via the "bus" linking all nodes in the network together. The receiving nodes "filter" incoming messages, only accepting messages which have "message types" that they have been registered to recognize before-hand. This approach increases "network elasticity" by allowing new nodes to be added to the network without requiring prior registration with the other nodes, so long as these new nodes have been programmed to recognize the existing set of message types [1].

Four commonly used electronic systems in passenger vehicles are CAN, TTCAN, FlexRay and LIN. In the following sections, we will discuss and compare each in detail.

## IV. Basics

CAN is a serial fieldbus communication network. CAN uses a "bus" to transmit messages between nodes in the network. The bus arrangement reduces the number of connections between nodes. Each node has a single 2-way connection to the bus. Point-to-point connections, by contrast, require an individual link between every node. The bus arrangement requires "n" connections for "n" nodes while a point-to-point arrangement requires "(n(n-1))/2" connections for "n" nodes. The bus arrangement provides a significant reduction in cost and reduces propagation delays. The fact that it is a serial bus indicates that only a single message can reside on the network at a time. CAN is a distributed "real-time" network used in the field ("Fieldbus") where "real-time" events occur [1].

Basic CAN is "event-oriented" (event-triggered), meaning that messages are generated in response to the generation of events on the network. Nodes send messages following an action (receives a data frame) or a request for information (receives a remote frame). However, due to the nature of CAN message arbitration where the outcome of and the time required to resolve each arbitration is completely dependent on the value of the message ids at the time of arbitration, the times required to send and receive messages cannot be characterized deterministically. This is considered to be insufficient for "real-time" [3] applications where "hard" real-time constraints impose strict requirements on the ability of network nodes to intercommunicate. In these applications, it is desirable to be capable of triggering deliberate messages at precise instants, avoiding and superseding arbitration. Transmission and reception of messages is temporarily performed during a deterministic time slot instant. To facilitate the adherence to a "time-oriented" (time-triggered) approach [1, 5], an upper layer called the TTCAN (Time-Triggered CAN) layer was developed [1].

As more functionality of the automobile is transferred from mechanical to electronic control, network topology complexity, number of gateways between networks and the need for portability between automobile platforms have

steadily increased. The standard high-speed CAN bit rate of 1 Mbit/second has been superseded by 5 Mbit/second rates on a single-channel link and 10 Mbit/second rates on two-channel links (with redundant traffic possible) using the "FlexRay" specification. "Hard" real-time requirements mean that "time-triggered" events are necessary, such that TTCAN-like messaging is employed. In addition, the bit rate (bandwidth) of the system is variable depending on the bus load required. Additional channels (links) are used to vary the bit rate (throttling) or when multiple nodes must send data at the same time. A master FlexRay node provides a "global reference time" from which all other nodes on the network derive their message timing. An optional "bus guardian" component manages Error Containment of messages, above and beyond the functionality provided by the CAN specification [1].

LIN was developed as a simpler, more cost-effective alternative fieldbus technology for bit rates (up to 20 kbits per second) on par with Low Speed CAN. Like FlexRay, LIN has a single master LIN node which coordinates timing across the network of slave nodes. However, in LIN there is no message arbitration and all messages are considered to be deterministic with static latencies. LIN provides sufficient functionality at low cost (with a finite number of nodes). It lacks redundant slave message re-sending (only a master node controls routing) and is for low performance (bit rate, bandwidth) requirements [1].

CAN is scalable and dependable due to its bus topology. However, CAN lacks deterministic scheduling for real-time events because message arbitration can in theory delay message routing indefinitely. LIN, TTCAN and FlexRay all provide deterministic scheduling with the aid of a master network node. FlexRay bandwidth is variable with multiple channels available. CAN, TTCAN and FlexRay support message re-transmissions following transmission of error messages.

## V. Message Arbitration

CAN "message arbitration" is the standard scheduling [4] algorithm by which nodes which are attempting to send messages at the same time must "arbitrate their dispute" by first transmitting the value of their message id to the bus. The message id with the lowest value has priority over all other messages, and therefore the node responsible for this message "wins the arbitration" to send its message over the bus to the other nodes. A transmitting node detects whether it has won or lost "message arbitration" via "bit monitoring" [1, 8].

"Message arbitration" is used in CAN, TTCAN and FlexRay systems and is more efficient than static scheduling. It is the basis for resolving message broadcasting, but it is generally non-deterministic because local nodes decide autonomously when to broadcast and the results of arbitration can only be determined probabilistically. Also, "worse case execution time" is favored over "best estimate execution time". WCET depends on deterministic behavior, which message arbitration lacks. For these reasons, "message arbitration" is less useful for "real-time" applications [5, 6].

## VI. Frames

CAN delineates the types of "frames" (message formats) that can be sent over the bus into 4 main groups. "Data Frames" transport data between nodes. "Remote Frames" are requests for transmission of "Data Frames". "Error Frames" are messages indicating that an error has occurred (transmission bit error, reception bit error, etc). "Overload Frames" are used to request a delay of bus signal between the transmission of data or remote frames when a node is "overloaded" with data. Modern processors often do not require the use of the "Overloaded Frame" [1, 8].

TTCAN resides primarily in the OSI "session" layer. Synchronization of messages on the network is accomplished by a single master TTCAN node which assigns the remaining nodes on the network "Time Slots". These time "windows" are the only times available for nodes to transmit. In general, each node has a different set of assigned windows in which it may transmit. Assignment is controlled by the master TTCAN node, and the number, duration and type of window assigned to each node will depend on the nature of the node and what the node is being used for in the network [1, 8].

FlexRay "static segments" are used for synchronous real-time time-triggered events, and FlexRay "dynamic segments" are used for asynchronous event-triggered events where one dynamic segment may demand more or less bus bandwidth than another dynamic segment. In this way, a single FlexRay frame can perform arbitration and non-arbitration via a method similar to a single TTCAN Base Cycle [1].

Master LIN nodes decide which messages may be transmitted on the bus and their timing. Slave LIN nodes are not allowed to send messages to the bus unless the master LIN node requests them to do so. Each message is called a "Frame" (similar to CAN) with a fixed format but a variable length. The main fields are the "Header", "Response Space", "Response" and "Interframe Space". The Header field is always provided by the master LIN node, and the Response field is provided by the transmitting node (either of the master or slave LIN nodes) [1].

CAN frames provide dependability and fault-tolerance via re-transmissions and error frames. Master LIN nodes control message frame re-transmission following a transmission error. Data correctness is ensured using CRC fields in LIN, CAN, TTCAN and FlexRay frames. "Global reference time" found in LIN, TTCAN and FlexRay provides better schedulability of time-triggered messages, while trading off some efficiency and local node autonomy. FlexRay has greater extensibility to future platforms because of its variable bit rate throttling, support for fiber optics and inclusion of an optional bus guardian.

TABLE I. ATTRIBUTES OF "AUTOMOTIVE NETWORKS"

| | (A) Messaging | (B) Network Synchronization | (C) Node Control | (D) Message Retransmission | (E) Error Management | (F) Bandwidth | (G) Physical | (H) Applications | (I) Cost |
|---|---|---|---|---|---|---|---|---|---|
| LIN | Deterministic, Static Message Scheduling | Global Reference Time | Master/Slave | Limited Message Retransmisison Supported | CRC Frame Field, ID Field Parity Bits, Diagnostic Frames | 10 kBit/s max (variable: 600, 1200, 2400, 4800, 9600) | 1-Wire | Displays, Lighting, Alarm Systems, A/C, Seat & Mirror Adjustments, Power Windows, Windshield Wipers, Headlamps | Low |
| CAN | Event-Triggered Messages | Priority-Based Arbitration | Autonomous | Message Retransmisison Supported | CRC Frame Field, Bit Monitoring Bit Stuffing, Error Frames, Overload Frames | 125 kBit/s max | 2-Wire (interference protection) | | Medium |
| TTCAN | Event- & Time-Triggered Messages | Global Reference Time | Autonomous, Master/Slave | | | 10 MBit/s max | 2-Wire (interference protection) | Engine, Transmission, Braking, Steering, Suspension, Assistance, Safety, Diagnostics | High |
| FlexRay | Event- & Time-Triggered Messages | Global Reference Time | Autonomous, Master/Slave | | 2 CRC Frame Fields, Bus Guardian (optional) | 2 Channels: 5 MBit/s, 10 MBit/s | 2 Channel, 2-Wire (interference protection), fiber optic (optional) | | High |

Note: "Event-Triggered" messages (non-deterministic, arbitration), "Time-Triggered" messages (deterministic, non-arbitration)

TABLE II. ATTRIBUTES OF "CYBER-PHYSICAL SYSTEMS" (AS PERTAINING TO TABLE I, ABOVE)

| | Ubiquitous, Wireless [10, 15] | Ad-hoc Reorganizing / Reconfiguring [10] | Distributed [13] | Resource Management (power) [10] | Real-time Response Time [10, 13, 15] | Scheduling (timing, latency, jitter) [10, 12] | Availability [15] | Security [10, 15] | Fault Tolerance, Reliability [10, 12, 14, 15] | Validation & Verification ("V&V"), Certification [13, 14, 15] |
|---|---|---|---|---|---|---|---|---|---|---|
| LIN | Local to Single Car, Wired | Autonomous Nodes, Centralized Registration Unnecessary, But Nodes Must Recognize Existing Network Message Types | Gateways Between Multiple, Disparate Wired Networks on a Single Car | Network Sleep-Mode Frame | Soft | **I**: A - D<br>**II**: E, F, G | **I**: D, E, G<br>**II**: A – C, F | **I**: E<br>**II**: G | **I**: D, E<br>**II**: A – C, F, G | LIN Spec. v. 2.0<br>LIN Protocol Conformance Tests 1.3 / 2.0<br>LIN Physical Conformance Tests 2.0 |
| CAN | | | | Error Confinement "Bus Off" Mode, "Standby / Sleep" Mode | Soft | **I**: D - G<br>**II**: A, B, C | **I**: A-G<br>**II**: A, B, C | **I**: E, G<br>**II**: none | **I**: A – G<br>**II**: none | ISO-11898-1/2/3/4/5/6/7 |
| TTCAN | | | | | Hard | **I**: A - F<br>**II**: none | **I**: A - G<br>**II**: none | **I**: E, G<br>**II**: none | **I**: D – G<br>**II**: A – C | ISO-11898-4 |
| FlexRay | | | | | Hard | **I**: A - F<br>**II**: none | **I**: A - G<br>**II**: none | **I**: E, G<br>**II**: none | **I**: D – G<br>**II**: A – C | Protocol Spec v. 2.1, Data Link Layer Conformance Spec v. 2.1.1 |

Note: "**I**" is a "better" fit and "**II**" is a "worse" fit within the CPS paradigm.

## VII. Error Management

Detection of an error message on the bus occurs because CAN specifies that no message on the bus may consist of 5 or more bits which have the same polarity (i.e.: ..00000.., and ..11111.. are considered errors). It happens that the first 6 bits of an error message are all dominant bits (000000). Whenever a node identifies a 5-bit sequence of mono-polarity bits (or 6-bit sequence from an error message), it halts whatever it is doing and issues an error message [1, 8].

"Bit stuffing" is the process of inserting a reverse polarity bit immediately following the sequence of mono-polarity bits. This "stuff bit" serves as a flag to receiving nodes that the prior mono-polarity bit sequence should not be an indication of an error state. Receiving nodes remove the identified "stuff bit" from the incoming message, and the remainder of the message is parsed.

LIN is capable of performing error detection similar to CAN. However, due to the "lightweight" nature of LIN, this functionality is severely reduced. Unlike CAN, LIN does not have error frames for signaling errors, but instead the slave LIN node that detects an error locally issues a diagnostic message, in the form of a standard data frame, to the master LIN node only [1].

CAN error frames and LIN diagnostic frames provide dependability and fault-tolerance, as well as security against network failures, attacks, or electromagnetic interference. CAN "bit stuffing" provides scalability to larger CAN networks which may use an increased number of message ids.

## VIII. Conclusions

In this paper, we have compared four important QoS related features among different fieldbus networks. In particular, CAN is scalable and dependable due to its bus topology. However, CAN lacks deterministic scheduling for real-time events because message arbitration can, in theory, delay message routing indefinitely. LIN, TTCAN and FlexRay all provide deterministic scheduling. FlexRay bandwidth is variable with multiple channels available. TTCAN, LIN and FlexRay have a master node for handling scheduling of messages.

CAN frames provide dependability and fault-tolerance via re-transmissions and error frames. CAN error frames and LIN diagnostic frames provide dependability and fault-tolerance, as well as security against network failures, attacks, or electromagnetic interference. The two tables given below summarize the comparison.

In particular, Table I, "Attributes of Automotive Networks", shows several key factors useful for describing and comparing LIN, CAN, TTCAN and FlexRay networks. "Messaging" compares event-triggered and time-triggered messages broadcast on the network. "Network Synchronization" compares the use of global reference time versus the use of message arbitration. "Node Control" shows how nodes in a network are controlled, via autonomous control or via a master/slave configuration. "Error Management" compares the strategies used by the networks to avoid, contain and recover from errors generated on the network. "Bandwidth" describes the various data rate capacities of the networks. "Physical" compares the limitations and features of the physical layer. "Applications" lists some of the typical applications for the networks, divided into low speed (LIN, CAN) and high speed (TTCAN, FlexRay) domains.

Table II, "Attributes of Cyber Physical Systems", attempts to relate some of the key factors which describe CPS to the automotive networks discussed in the first table. "Ubiquitous" illustrates that automotive networks are local to an automobile platform and are not wireless, which goes against the spirit of CPS. "Ad-Hoc" describes how nodes can be added and removed from a network without configuration. "Distributed" shows how multiple networks can intercommunicate on the same local automobile platform, but are not geographically distant as is found in a typical CPS. "Resource Management" shows the strategies used by the networks to conserve power. "Real-time Response Time" shows the typical response time capability of each network. "Scheduling", "Availability", "Security" and "Fault Tolerance" compare how automotive attributes perform within the CPS paradigm for each of the networks. Each of the automotive network attributes is labeled "A" through "H" and is cross-referenced with a CPS attribute. Each automotive attribute is a "better" (**I**) or "worse" (**II**) fit within the CPS paradigm as shown for each CPS attribute column. "Validation and Verification" shows the international standards, conformance tests and direction for certification for each network.

## Acknowledgment

This work is supported in part by NSF CAREER Award (CNS0746643).

## References

[1] D. Paret, Multiplexed Networks for Embedded Systems, Wiley, 2007

[2] D. Caro, Automation Network Selection, ISA, 2004, pages 93-106

[3] M. Dahleh, E. Frazzoli, A. Megretski, S. Mitter, A. Ozdaglar, P. Parrilo, D. Shah, "Network Control for Cyber Physical Systems: Position Paper", pages 1-3

[4] A. Easwaran, I. Lee, "Compositional Schedulability Analysis for Cyber-Physical Systems", ACM SIGBED Review, 5(1): 1-2

[5] T.A. Henzinger, J. Sifakis, "The Discipline of Embedded Systems Design", Computer, 2007, 40(10): 32-40

[6] E. Lee, "Cyber Physical Systems: Design Challenges", Object Oriented Real-Time Distributed Computing (ISORC), 11th IEEE International Symposium, 2008, pages 363-369

[7] Y. Tan, S. Goddard, L. Perez, "A Prototype Architecture for Cyber-Physical Systems", ACM SIGBED Review, 5(1): 1-2

[8] W. Voss, A Comprehensible Guide to Controller Area Network, Copperhill Technologies Corporation, 2005

[9] Automotive Handbook, Bentley Publishers, 2007, pages 1092-1097

[10] Goddard, Steve, Deogun, Jitender, S., "Future Mobile Cyber-Physical Systems: spatio-temporal computational environments", 2008, pages 1-3

[11] Cook, Jeffrey, A., "Cyber-Physical Systems and the Twenty-First Century Automobile", NSF Workshop on Cyber-Physical Systems, 2006, pages 1-3

[12] Fuhrman, Tom, "Position Paper for NSF Workshop on Cyber-Physical Systems", Workshop on Cyber-Physical Systems, 2006, page 1-3

[13] Gill, Helen, "From Vision to Reality: Cyber- Physical Systems", Presentation, HCSS National Workshop on New Research Directions for High Confidence Transportation CPS: Automotive, Aviation and Rail, 2008

[14] Rajkumar, Raj, "A Vision for Automotive CPS", Presentation, HCSS National Workshop, on New Research Directions for High Confidence Transportation CPS: Automotive, Aviation and Rail, 2008

[15] Wing, Jeanette, M., Presentation, "Cyber-Physical Systems Research Charge", Cyber-Physical Systems Summit, 2008