

SURFACE VEHICLE INFORMATION REPORT

SAE J1583

**CAN-
CELLED
JUL1999**

Issued 1990-03
Cancelled 1999-07

Superseding J1583 MAR90

Controller Area Network (Can), An In-Vehicle Serial Communication Protocol

Foreword—This Document has not changed other than to put it into the new SAE Technical Standards Board format.

Controller Area Network (CAN) is an advanced serial communication protocol which efficiently supports distributed real-time control with a very high level of security.

Its domain of applications range from high speed networks (greater than 125K bits/s) to low speed multiplex wiring (10K bits/s or less).

TABLE OF CONTENTS

1.	Scope	4
2.	References	4
3.	General Features.....	4
4.	CAN Architecture.....	4
5.	Bit Representations, Coding/Decoding, and Stuffing	5
5.1	Bit Representations	5
5.2	Bit Coding/Decoding and Stuffing	5
6.	Arbitration	6
7.	CAN Message Frame Types.....	6
7.1	Data Frame.....	6
7.2	Remote Frame.....	9
7.3	Error Frame	10
7.4	Overload Frame.....	10
7.5	Interframe Space.....	11
8.	Error Detection	11
8.1	Bit Error	11
8.2	Bit Stuffing Error	12

SAE Technical Standards Board Rules provide that: "This report is published by SAE to advance the state of technical and engineering sciences. The use of this report is entirely voluntary, and its applicability and suitability for any particular use, including any patent infringement arising therefrom, is the sole responsibility of the user."

SAE reviews each technical report at least every five years at which time it may be reaffirmed, revised, or cancelled. SAE invites your written comments and suggestions.

QUESTIONS REGARDING THIS DOCUMENT: (724) 772-8512 FAX: (724) 776-0243
TO PLACE A DOCUMENT ORDER: (724) 776-4970 FAX: (724) 776-0790
SAE WEB ADDRESS <http://www.sae.org>

8.3	CRC Error	12
8.4	Form Error.....	12
8.5	Acknowledgement Error.....	12
8.6	Error Detection Capabilities	12
8.7	Error Confinement	13
8.8	Error Counting	13
9.	Bit Timing.....	14
9.1	INSYNC	14
9.2	SJW 1 \s& 2	14
9.3	TSEG 1	14
9.4	TSEG 2	15
9.5	Bit Time Calculation.....	15
10.	Synchronization	15
11.	Sleep Mode/Wake-Up	15
Figure 1	Bit Stuffing	5
Figure 2	Data Frame	7
Figure 3	Arbitration Field.....	7
Figure 4	Control Field	8
Figure 5	Data Length Code.....	8
Figure 6	CRC Field	8
Figure 7	Acknowledgement Field.....	9
Figure 8	Remote Frame	9
Figure 9	Error Frame.....	10
Figure 10	Overload Frame	10
Figure 11	Interframe Space	11
Figure 12	Bit Segments	14
Appendix A		16
A1.	82526 Functional Overview	16
A1.1	IMP (Interface Management Processor)	16
A1.2	Quasi Dual Port RAM.....	16
A1.3	TCL (Transceiver Control Logic)	16
A1.4	BSP (Bit Stream Processor)	16
A1.5	EML (Error Management Logic)	16
A1.6	BTL (Bus Timing Logic)	16
A1.7	PIU (Processor Interface Unit)	16
A1.8	CG (Clock Generator)	17

A2.	Quasi Dual Port RAM Memory Layout and Registers.....	18
A2.1	Control Register (00H)	19
A2.2	Status Register (01H)	20
A2.3	Interrupt Pointer (02H)	20
A2.4	Bus Timing Register 0 (03H)	21
A2.5	Bus Timing Register 1 (04H)	21
A2.6	Output Control Register (05H)	22
A2.7	Communication Object (06H).....	23
Figure A1	Functional Block Diagram	17
Figure A2	Buffer Memory Layout.....	18
Figure A3	Control Register (ooh)	19
Figure A4	Status Register	20
Figure A5	Interrupt Pointer (02H)	20
Figure A6	Bus Timing Register 0 (03H)	21
Figure A7	Baud Rate Prescaler.....	21
Figure A8	Bus Timing Register 1 (04H)	21
Figure A9	Output Control Register (05H)	22
Figure A10	Programmable Features	22
Figure A11	Output Drive Configuration	23
Figure A12	Output Control Logic	23
Figure A13	Descriptor Byte 1 (06H)	24
Figure A14	Descriptor Byte 2 (07H)	24
Figure A15	Descriptor Byte 3 (08H)	24

1. **Scope**—The scope of this specification is to define the transfer layer and the consequences of the Controller Area Network (CAN) protocol on the surrounding layers.

2. **References**—There are no referenced publications specified herein.

3. **General Features**—CAN has the following general features:

Multimaster Architecture: Assurance that any node can have access to the bus for transmission of messages.

Prioritization Of Messages: Bus access is determined by the selectable predetermined priority of the 2032 different messages allowable.

Guarantee Of Latency Time: Assurance that the maximum latency time for the highest priority message is less than 150 μ s (at 1M bits/s).

Broadcast Message Transfer: Any number of nodes can receive and simultaneously act upon the same message.

Message Arbitration: If two or more senders start transmitting messages at the same time, the bus access conflict is resolved by nondestructive prioritized bitwise arbitration via the message identifier.

Data Consistency: Assures that a message is simultaneously accepted either by all nodes or by no node.

Programmable Transfer Rate: The bit-rate is programmable up to 1m bit/s; once set it is uniform and fixed in the system.

Powerful Error Handling: Includes automatic retransmission of corrupted messages and distinction between temporary errors and permanent failures of nodes.

Configuration Flexibility: Nodes may be added without requiring any change in the software or hardware of any node or application layer.

Fault Detection: Distinction between temporary errors and permanent failures of node and autonomous switching off of defect nodes.

4. **CAN Architecture**—To achieve design transparency and implementation flexibility CAN has been developed structurally in layers. Different silicon implementations will integrate these layers to different depths in hardware. This information report will address the protocol, or bus communication rules, that all implementations must support or have the capability of supporting. One implementation example already in production is the Intel 82526. An overview of its functionality and use is provided in Appendix A. Another such implementation is BasicCAN. It is not within the scope of this document to cover BasicCAN in detail. Future updates may provide additional information.

The layers are:

a. Object Layer: The scope of the object layer includes:

1. Deciding which messages received by the transfer layer are actually to be used (acceptance filtering)
2. Determining which messages are to be transmitted (prioritized message handling)
3. Providing an interface to the host CPU if there is a host in the system.

b. Transfer Layer: The scope of the transfer layer includes:

1. Error checking
2. Error signalling and confinement
3. Message validation and acknowledgement
4. Performing arbitration
5. Controlling message framing
6. Fault confinement
7. General features of bit timing

c. Physical Layer: The scope of the physical layer is the actual transfer of the message bits between the different nodes in the network with respect to all electrical properties. This includes: Signal level, bit representation, and transmission medium. The protocol does not specify the signal level and transmission medium. This is done intentionally to allow the user or system designer to adapt CAN to the specific application and application environment. Within one network the physical layer, of course, has to be the same for all nodes. There may be, however, much freedom in selecting a physical layer.

5. Bit Representations, Coding/Decoding, and Stuffing

5.1 Bit Representations—There are two logical bit representations defined: “dominant” and “recessive.” A recessive bit on the bus line appears only if all connected nodes at that time send a recessive bit. If one or more nodes send a dominant bit, the bus line reflects a dominant bit. In other words, a dominant bit always overwrites a recessive bit if emitted on the bus at the same time by a different node. If the electrical bus is implemented as an npn-open-collector bus, a dominant bit corresponds to a low level signal and a recessive bit corresponds to a high level signal.

5.2 Bit Coding/Decoding and Stuffing—Bit encoding is performed with an NRZ bit waveform representation using bit stuffing. Bit stuffing is used in the following frame segments:

- a. Start of frame
- b. Arbitration field
- c. Control field
- d. Data field
- e. CRC sequence

Whenever the transmit logic of CAN device detects five consecutive bits of identical levels to be transmitted, it inserts a complement bit in the transmitted bit stream. Whenever a CAN device receives five identical consecutive bit levels in the bit stream, its receive logic will automatically delete the next bit from the data stream (destuffing). Figure 1 for further detail.

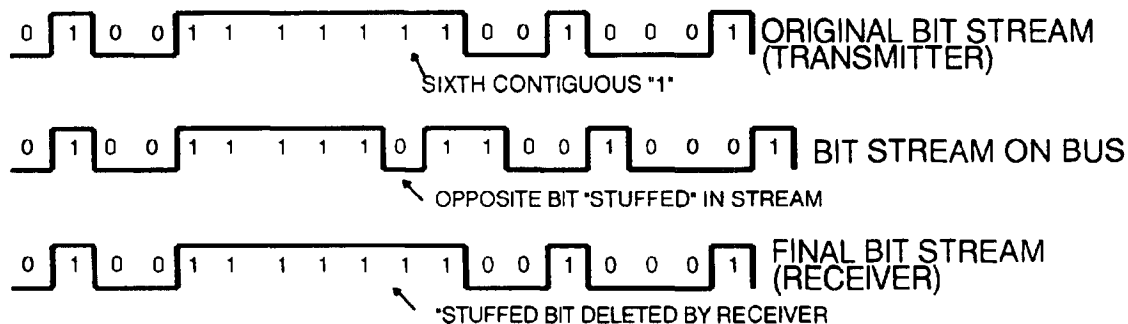


FIGURE 1—BIT STUFFING

6. **Arbitration**—The CAN protocol defines that each single message used in the communication network has a unique identifier. Using this method, the identifier assigns a name to the data frame and automatically implies the priority of the message. As a result, the identifier during bus access represents not only the message name but, more important, the priority of each specific message. Since the most significant bit (MSB) of an identifier is transmitted first, the identifier with the smallest digital value has the highest priority for bus access.

When bus idle is detected, any node may start to transmit a message. In a case where two or more CAN devices start a message transfer concurrently, the bus access conflict is solved by prioritized bitwise arbitration performed during the transmission of the arbitration field.

The transmit logic of a given node compares the bit level transmitted to the level monitored on the serial bus. The transmit logic immediately will stop a current message transfer if a recessive bit was sent but a dominant bit was monitored. This method guarantees the data transfer of the message with the highest priority even if there is a collision during the arbitration field of one or more messages.

An identifier can never be used for more than one specific message (total available number of identifiers = 2032). This guarantees that two or more nodes never simultaneously start a transmission of a data frame with the same priority of data. One exception would be a frame transfer simultaneously initiated by a transmitter and receiver. If one CAN device generates a request for actual data of a certain type by transmitting a remote frame and the CAN device normally transmitting that data transmits simultaneously, the CAN device responsible for this type of data will win the arbitration and continue the transmission. Hence, arbitration can not be solved by the identifier itself. For this reason, the remote transmission request (RTR-BIT) is included in the arbitration field. The RTR-BIT of the transmitter is always set dominant and, therefore, has a higher priority than the requesting RTR-BIT recessive. In this case, the remote transmission request by the receiver gets an immediate response by the transmitter.

7. **CAN Message Frame Types**—The CAN device supports four different frame types:

- a. Data frame
- b. Remote frame
- c. Error frame
- d. Overload frame

- 7.1 **Data Frame**—A Data Frame is composed of seven different fields:

- a. Start of frame
- b. Arbitration field
- c. Control field (Identifier)
- d. Data field (Data segment)
- e. CRC field
- f. Acknowledgement field
- g. End of frame

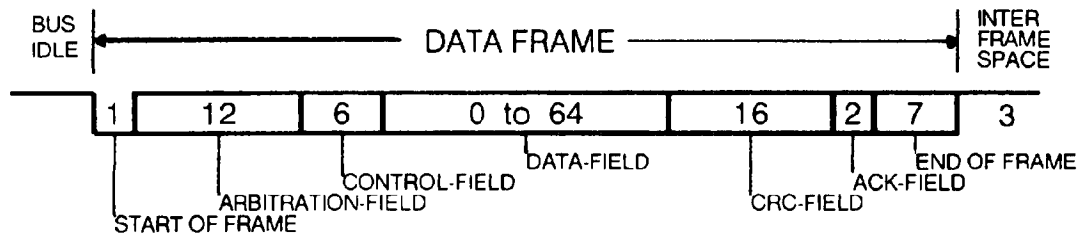


FIGURE 2—DATA FRAME

- 7.1.1 **START OF FRAME**—Signals the start of a data or remote frame. It consists of a single dominant bit used for a synchronization of receiving nodes. The CAN device will start a transmission only if a bus-idle state is detected. All nodes must synchronize to the leading edge of the node starting transmission first.
- 7.1.2 **ARBITRATION FIELD**—Consists of the message identifier and one additional control bit RTR-BIT. The simultaneous message transmission start of two or more nodes is solved by bitwise arbitration during the transmission of the arbitration field (for more details).

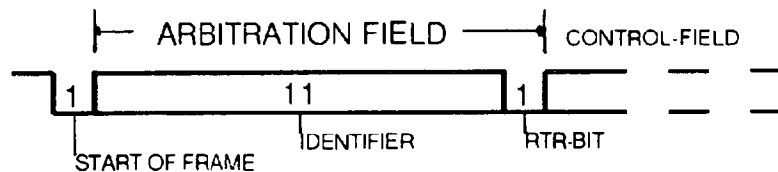


FIGURE 3—ARBITRATION FIELD

- 7.1.2.1 **Identifier**—This 11-bit field provides information about each individual message as well priority for the message. The identifier defines the corresponding specific message (e.g., engine speed, temperature, pressure, etc.) of a communication object. It's digital value represents the message priority for bus access.

NOTE—An identifier may be a physical or functional address or a combination of the two to determine a receiver of a frame on a multidrop line. The receiving CAN device decides, based on the acceptance filter process of a received identifier, whether data being received within a correct frame is to be accepted or not.

- 7.1.2.2 **RTR-BIT**—(Remote Transmission Request) A station, active as a receiver may initiate the transmission of the data by transmission of a remote frame to the network, addressing the data source via the identifier. Within a data frame, the RTR-BIT is transmitted as a dominant bit level (for more information see Section 7.2).
- 7.1.3 **CONTROL FIELD**—This field consists of 6 bits. It includes the data length code and two reserved bits. The reserved bits are automatically transmitted with a dominant bit level. The length of the data field (data segment) is coded in bytes.

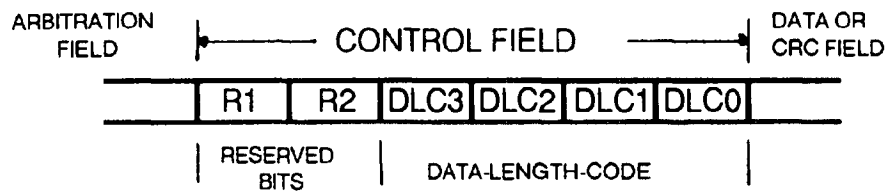


FIGURE 4—CONTROL FIELD

- 7.1.4 DATA FIELD—Data stored within the corresponding data segment in the Communication Buffer is transmitted within the data field. The length of the data field varies in the range from 0 to 8 bytes based on the value of the data length code. The byte at the lowest address of the data segment will be transmitted MSB first.

# DATA BYTES	DATA-LENGTH-CODE			
	DLC3	DLC2	DLC1	DLC0
0	d	d	d	d
1	d	d	d	r
2	d	d	r	d
3	d	d	r	r
4	d	r	d	d
5	d	r	d	r
6	d	r	r	d
7	d	r	r	r
8	r	d	d	d

"d" = dominant, "r" = recessive

FIGURE 5—DATA LENGTH CODE

- 7.1.5 CRC FIELD—Contains the CRC checksum (15 bits) followed by the CRC delimiter (1 bit). The cyclic redundancy code includes the start of frame bit, arbitration field, control field, data field and CRC field. The most significant bit (MSB) is transmitted first. The frame check sequence implemented in the CAN device is derived from a cyclic redundancy code best suited for frames with a total bit count of less than 127 bits (BCH Code). With start of frame included in the code word, any rotation of the code word can be detected by the absence of the CRC delimiter ("recessive" bit). (For further information about the CRC, See 8.6.)

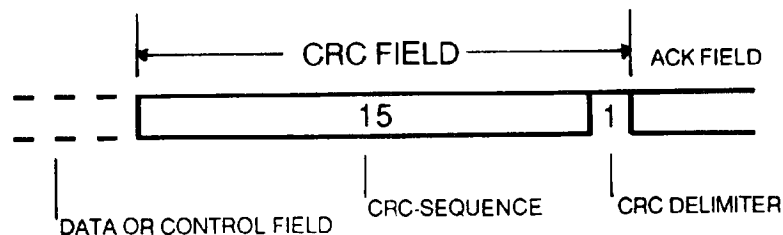


FIGURE 6—CRC FIELD