INTERNATIONAL STANDARD

ISO 11519-2

First edition
1994-06-15

Road vehicles — Low-speed serial data communication —

Part 2:
Low-speed controller area network (CAN)

Véhicules routiers — Communication en série de données à basse vitesse —

Partie 2: Réseau local à commande à basse vitesse (CAN)

Reference number
ISO 11519-2:1994(E)

Contents

International Organization for Standardization
Case Postale 56 • CH-1211 Genève 20 • Switzerland

Printed in Switzerland

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

International Standard ISO 11519-2 was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Sub-Committee SC 3, *Electrical and electronic equipment*.

ISO 11519 consists of the following parts, under the general title *Road vehicles — Low-speed serial data communication*:

— *Part 1: General and definitions*

— *Part 2: Low-speed controller area network (CAN)*

— *Part 3: Vehicle area network (VAN)*

— *Part 4: Class B data communication network interface (J1850)*

Road vehicles — Low-speed serial data communication —

Part 2:
Low-speed controller area network (CAN)

1 Scope

This part of ISO 11519 specifies the data link layer and the physical layer of the low-speed Controller Area Network (CAN), a communications network up to 125 kbit/s, for road vehicle application. The low-speed CAN is a serial communication protocol supporting distributed real-time control and multiplexing.

This specification describes the general architecture of CAN in terms of the hierarchical layers defined in the ISO-OSI model according to ISO 7498.

2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this part of ISO 11519. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this part of ISO 11519 are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below. Members of IEC and ISO maintain registers of currently valid International Standards.

ISO 7498:1984, *Information processing systems — Open Systems Interconnection — Basic Reference Model.*

ISO 8802-2:1989, *Information processing systems — Local area networks — Part 2: Logical link control.*

ISO/IEC 8802-3:1993, *Information technology — Local and metropolitan area networks — Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications.*

ISO 11519-1:1994, *Road vehicles — Low-speed serial data communication — Part 1: General and definitions.*

3 Definitions and abbreviations

For the purposes of this part of ISO 11519 the definitions given in ISO 11519-1 apply. The following additional definitions and abbreviations are specific to this part of ISO 11519.

3.1 Data link layer definitions

3.1.1 bit stuffing: Technique used in bit-oriented protocols in order

— to achieve data transparency (arbitrary bit patterns may not be interpreted as protocol information) and

— to provide "dominant" to "recessive" edges, and vice versa, which are necessary for correct resynchronization when using a non-return-to-zero bit representation.

Whenever the transmitting logic encounters a certain number (stuff width) of consecutive bits of equal value in the data, it automatically stuffs a bit of complementary value — a stuffbit — into the outgoing bit stream. Receivers destuff the frame, i.e. the inverse procedure is carried out.

3.1.2 bus value: One of two complementary logical values: "dominant" or "recessive". The "dominant" value represents the logical "0" and the "recessive" represents the logical "1". During simultaneous transmission of "dominant" and "recessive" bits, the resulting bus value will be "dominant".

3.1.3 multicast: Method by which a single frame is addressed to a group of nodes simultaneously.

3.1.4 broadcast: Special case of multicast, whereby a single frame is addressed to all nodes simultaneously.

3.1.5 receiver: Device that converts signals used for transmission back into logical information or data signals.

3.1.6 transmitter: Device that converts information or data signals to electrical or optical signals so that these signals can be transferred across the communication medium.

3.2 Physical layer definitions

3.2.1 common mode bus voltage range: Boundary voltage levels of V_{CAN_L} and V_{CAN_H}, for which proper operation is guaranteed if the maximum number of electronic control units (ECUs) are connected to the bus line.

3.2.2 differential internal capacitance, C_{diff}: Capacitance of an ECU which is seen between CAN_L and CAN_H during the recessive state when the ECU is disconnected from the bus line (see figure 1).

3.2.3 differential internal resistance, R_{diff}: Resistance of an ECU which is seen between CAN_L and CAN_H during the recessive state when the ECU is disconnected from the bus line (see figure 1).

3.2.4 differential voltage, V_{diff}:

$$V_{diff} = V_{CAN\_L} - V_{CAN\_H}$$

where V_{CAN_L} and V_{CAN_H} are the voltages of CAN_L and CAN_H, respectively, relative to ground of each individual ECU.

3.2.5 internal capacitance, C_{in}: Capacitance of an ECU which is seen between CAN_L (or CAN_H) and ground during the recessive state when the ECU is disconnected from the bus line (see figure 1).

3.2.6 internal delay time, t_{ECU}: Sum of all asynchronous delay times of an ECU occurring on the transmitting and receiving paths, relative to the bit timing logic unit of the protocol IC of each individual ECU disconnected from the bus line.

3.2.7 internal resistance, R_{in}: Resistance of an ECU which is seen between CAN_L (or CAN_H) and ground during the recessive state when the ECU is disconnected from the bus line (see figure 1).

3.2.8 physical layer: Electrical circuit realization that connects an ECU to a bus. The total number of ECUs connected on a bus is limited by electric loads on the bus line.

3.2.9 physical media: Pair of parallel wires of the bus, shielded or unshielded, depending on EMC requirements. The individual wires are denoted as CAN_L and CAN_H. The corresponding pins of ECUs are also denoted by CAN_L and CAN_H.

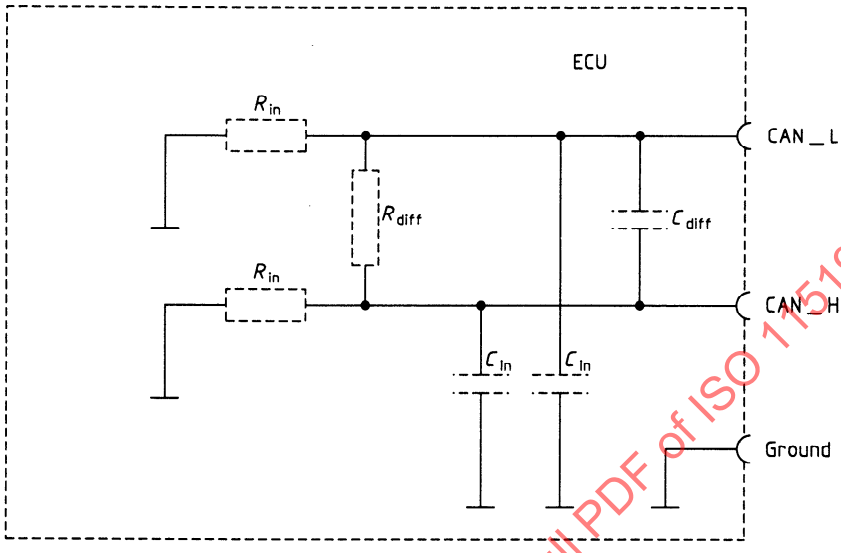


Figure 1 — Definitions of internal capacitances and internal resistances of an ECU

3.3 List of abbreviations

ACK	Acknowledgement
ECU	Electronic Control Unit
EOF	End Of Frame
CAN	Controller Area Network
CRC	Cyclic Redundancy Check
DLC	Data Length Code
IC	Integrated Circuit
FCE	Fault Confinement Entity
LAN	Local Area Network
LLC	Logical Link Control
LME	Layer Management Entity
LPDU	LLC Protocol Data Unit
LSB	Least Significant Bit
LSDU	LLC Service Data Unit
MAC	Medium Access Control
MAU	Medium Access Unit

MDI	Medium-Dependent Interface
MPDU	MAC Protocol Data Unit
MSB	Most Significant Bit
MSDU	MAC Service Data Unit
NRZ	Non-Return-to-Zero
OSI	Open System Interconnection
PL	Physical Layer
PLS	Physical Signalling
PMA	Physical Medium Attachment
RTR	Remote Transmission Request
SOF	Start Of Frame

4 Basic concepts of CAN

CAN has the following properties:

— multimaster priority-based bus access;

— non-destructive contention-based arbitration;

— multicast frame transfer by acceptance filtering;

— remote data request;

— configuration flexibility;

— system-wide data consistency;

— error detection and error signalling;

— automatic retransmission of frames that have lost arbitration or have been destroyed by errors during transmission;

— distinction between temporary errors and permanent failures of nodes and autonomous switching-off of defective nodes.

This part of ISO 11519 specifies the low-speed CAN for applications up to 125 kbit/s.

4.1 Frames

Information on the bus is sent in fixed format frames of different but limited lengths. When the bus is free, any connected node may start to transmit a new frame.

4.2 Bus access method

When the bus is free, any node may start to transmit a frame. If two or more nodes start to transmit frames at the same time, the bus access conflict is resolved by contention-based arbitration using the identifier. The mechanism of arbitration guarantees that neither information nor time is lost. The transmitter with the frame of highest priority gains bus access.

4.3 Information routing

In CAN systems a node does not make use of any information about the system configuration (e.g. node address). Instead, receivers accept or do not accept information based upon a process called "Frame Acceptance Filtering", which decides whether the received information is relevant or not. There is no need for receivers to know the transmitter of the information, and vice versa.

4.4 System flexibility

Nodes can be added to the CAN network without requiring any change in the software or hardware of any node, if the added node is not the transmitter of any data frame (see 8.3.1).

4.5 Data consistency

Within a CAN network it is guaranteed that a frame is simultaneously accepted either by all nodes or by no node. Thus data consistency is a property of the system, achieved by the concepts of multicast and by error handling.

4.6 Remote data request

By sending a remote frame, a node requiring data may request another node to send the corresponding data frame. The data frame and the corresponding remote frame are named by the same identifier.

4.7 Error detection

The following measures are provided for detecting errors:

— monitoring (transmitters compare the bit levels to be transmitted with the bit levels detected on the bus);

— 15-bit cyclic redundancy check;

— variable bit stuffing with a stuff width of 5;

— frame check.

4.8 Error signalling and recovery time

Corrupted frames are flagged by any transmitting node and any normally operating (error-active) receiving node. Such frames are aborted and will be retransmitted according to the implemented recovery procedure (see 6.4.3). The recovery time, from detecting an error until the possible start of the next frame, is typically 17 bit times to 23 bit times (in the case of a heavily disturbed bus up to 29 bit times), if there are no further errors.

4.9 Acknowledgement

All receivers check the consistency of the received frame, and will acknowledge a consistent frame and flag an inconsistent frame.

4.10 Automatic retransmission

Frames that have lost arbitration and frames that have been disturbed by errors during transmission will be re-transmitted automatically when the bus is idle again. A frame that will be retransmitted is handled like any other frame, i.e. it participates in the arbitration process in order to gain bus access.

4.11 Fault confinement

CAN nodes are able to distinguish short disturbances from permanent failures. Defective transmitting nodes are switched off. "Switched off" means that a node is logically disconnected from the bus line, so that it can neither send nor receive any frames.

4.12 "error-active"

An "error-active" node can normally take part in bus communication and send an active error flag when an error has been detected. The active error flag consists of six dominant consecutive bits and violates the rule of bit stuffing, all fixed formats appearing in a regular frame (see 11.1.5).

4.13 "error-passive"

An "error-passive" node shall not send an active error flag. It takes part in bus communication, but when an error has been detected, a passive error flag is sent. The passive error flag consists of six recessive consecutive bits. After transmission, an "error-passive" node will wait an additional time before initiating a further transmission (see suspend transmission in 8.3.5 and 11.1.5).

4.14 "bus off"

A node is in the state "bus off" when it is switched off from the bus due to a request of a fault confinement entity. In the "bus off" state, a node can neither send nor receive any frames. A node can leave the "bus off" state only upon a user request.

5 Layered architecture of CAN

5.1 Reference to the OSI model

According to the OSI reference model, the CAN architecture represents two layers:

— data link layer,

— physical layer.

This standard describes the data link layer and physical layer of CAN, as shown in figure 2.

According to ISO 8802-2 and ISO 8802-3 (LAN standards), the data link layer is subdivided into:

— Logical Link Control (LLC);

— Medium Access Control (MAC).

The physical layer is subdivided into:

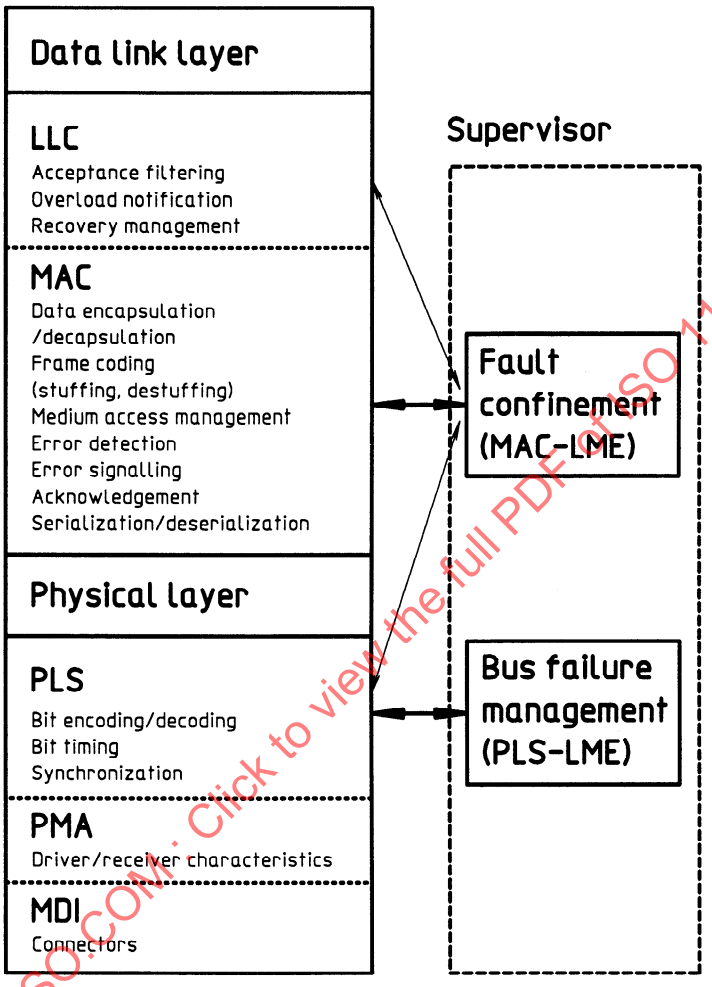
— Physical Signalling (PLS);

— Physical Medium Attachment (PMA);

— Medium-Dependent Interface (MDI).

The MAC sublayer operations are supervised by a management entity called the "Fault Confinement Entity" (FCE). Fault confinement is a self-checking mechanism that makes it possible to distinguish short disturbances from permanent failures (fault confinement, see 11.1).

The physical layer may be supervised by an entity that detects and manages failures of the physical medium (for example shorted or interrupted bus lines — bus failure management, see 11.2).



Key
LLC = Logical Link Control
MAC = Medium Access Control
PLS = Physical Signalling
PMA = Physical Medium Attachment
MDI = Medium-Dependent Interface
LME = Layer Management Entity

Figure 2 — Layered architecture of CAN

5.2 Protocol specification

Two peer protocol entities communicate with each other by exchanging frame or protocol data units (PDU). An N-layer protocol data unit (NPDU) consists of an N-layer-specific protocol control information (n-PCI) and N-user data. In order to transfer a NPDU it must be passed to an N-1 layer entity via an N-1 service access point [(N-1)-SAP]. The NPDU is passed by means of the N-1 layer Service Data Unit [(N-1)-SDU] to the N-1 layer, the services of which allow the transfer of the NPDU. The service data unit is the interface data whose identity is preserved between N layer entities, i.e. it represents the logical data unit transferred by a service. The data link layer of the CAN protocol does not provide means for mapping one SDU into multiple PDUs, nor means for mapping multiple SDUs into one SDU, i.e. an NPDU is directly constructed from the associate NSDU and the layer-specific control information N-PCI. Figure 3 illustrates the data link sublayer interactions.

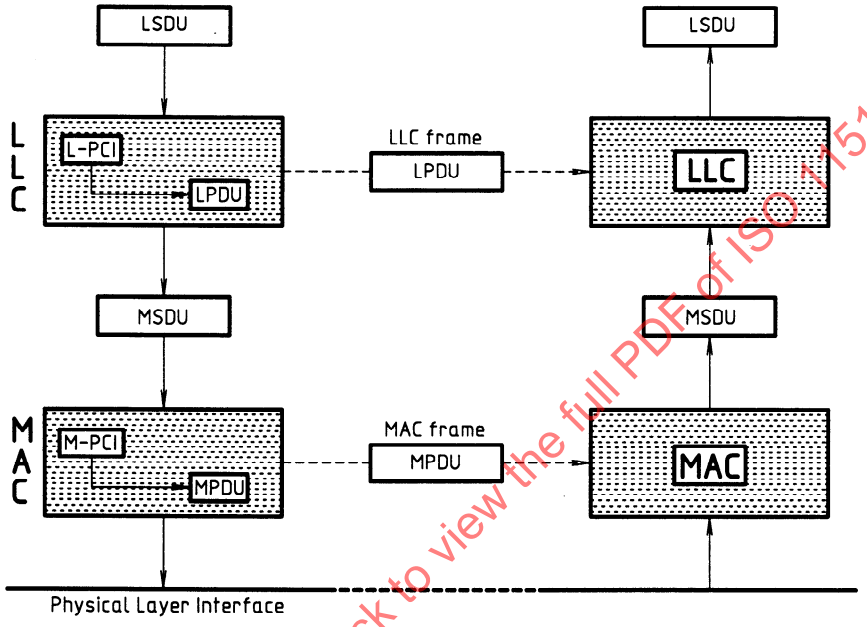


Figure 3 — Protocol layer interactions

5.3 Format description of services

5.3.1 Format description of service primitives

Service primitives are written in the form:

```
service.type(
            [parameter,...]
            ).
```

where

— "service" indicates the name of the service, e.g. L_DATA for data transfer service provided by the LLC sub-layer;

— "type" indicates the type of the service primitives (see 5.3.2);

— "[parameter,...]" is the list of values passed to the service primitives. The brackets indicate that this parameter list may be empty.

5.3.2 Types of service primitives

Service primitives are of three generic types.

a) *Service.request*

The request primitive is passed from the N-user (service user) to the N-layer (service provider) to request initiation of the service.

b) *Service.indication*

The indication primitive is passed from the N-layer to the N-user to indicate an internal N-layer (or sublayer) event which is significant to the N-user. This event may be logically related to a remote service request, or may be caused by an event internal to the N-layer (or sublayer).

c) *Service.confirm*

The confirm primitive is passed from the N-layer (or sublayer) to the N-user to convey the results of one or more associated previous service request(s). This primitive may indicate either failure to comply or some level of compliance. It does not necessarily indicate any activity at the remote peer interface.

5.4 LLC interface

The LLC sublayer offers two types of connectionless transmission services to the LLC user:

— unacknowledged data transfer service,

— unacknowledged remote data request service.

The interface service data from or to the user is described in 6.2. The messages that can be exchanged between LLC user and LLC sublayer are given in tables 1 and 2.

Table 1 — Message sent from LLC user to LLC sublayer

User-to-LLC message	Meaning
Reset_Request	Request to set the node into an initial state

Table 2 — Message sent from LLC sublayer to LLC user

LLC-to-user message	Meaning
Reset_Response	Response to the Reset_Request
Node_Status	Indicates the current status of the node, i.e. it signals whether or not the node is in the status "bus off"

The LLC interface messages from and to the supervisor fault confinement entity are described in 11.1.3.1.

6 Description of the LLC sublayer

The LLC (logical link control) sublayer describes the upper part of the OSI data link layer. It is concerned with those protocol issues that are independent of the type of medium access method.

6.1 Service of the LLC sublayer

The LLC sublayer offers two types of connectionless-mode transmission services:

a) Unacknowledged data transfer service

This service provides means by which LLC users can exchange link service data units (LSDU) without the establishment of a data link connection. The data transfer can be point-to-point, multicast or broadcast.

b) Unacknowledged remote data request service

This service provides means by which LLC users can request another remote node to transmit a link service data unit (LSDU) without the establishment of a data link connection.

The way in which the remote node serves the data request is not specified here. Basically, there are two ways.

1) The requested data is prepared by the remote user for transmission. In this case the data is located in a remote node buffer and will be transmitted by the LLC entity upon reception of the remote request frame.

2) The requested data will be transmitted by the remote user upon reception of the remote request frame.

According to the two different LLC services there are two types of frames from or to the user:

— LLC data frame,

— LLC remote frame.

The LLC data frame carries data from a transmitter to a receiver. The LLC remote frame is transmitted to request the transmission of a data frame (with the same identifier) from a (single) remote node. In both cases, the LLC sublayer notifies the successful transmission (or reception) of a frame to (or from) the user.

6.2 Service primitive specification

This section describes in detail the LLC service primitives and their associated parameters. The complete list of LLC service primitives is given in table 3.

Table 3 — LLC service primitives overview

Unacknowledged data transfer service	
L_DATA.request	Request for data transfer
L_DATA.indication	Indication of data transfer
L_DATA.confirm	Confirmation of data transfer
Unacknowledged remote data request service	
L_REMOTE.request	Request for remote data transfer
L_REMOTE.indication	Indication of remote data request
L_REMOTE.confirm	Confirmation of remote data request

The parameters that are associated with the different LLC service primitives are listed in table 4.

Table 4 — List of LLC service primitive parameters

LLC service primitive parameters	
IDENTIFIER	Identifies the data and its priority
DLC	Data length code
DATA	Data the user wants to transmit
TRANSFER_STATUS	Confirmation parameter

6.2.1 L_DATA.request

a) Function

The L_DATA.request primitive is passed from the LLC user to the LLC sublayer to request that an LSDU be sent to one or more remote LLC entities.

b) Semantics of the L_DATA.request primitive

The primitive shall provide parameters as follows.

```
L_DATA.request(
                IDENTIFIER
                DLC
                DATA
                )
```

The parameter DATA is insignificant if the associated LLC data frame is of data length zero.

c) Effect on receipt

Receipt of this primitive causes the LLC sublayer to initiate the transfer of an LLC data frame by use of the data transfer service provided by the MAC sublayer (see 8.1.1).

6.2.2 L_DATA.indication

a) Function

The L_DATA.indication primitive is passed from the LLC sublayer to the LLC user to indicate the arrival of an LSDU.

b) Semantics of the L_DATA.indication primitive

The primitive shall provide parameters as follows.

```
L_DATA.indication(
                  IDENTIFIER
                  DLC
                  DATA
                  )
```

The parameter DATA is insignificant if the associated LLC data frame is of data length zero.

c) Effect on receipt

The effect of receipt of this primitive by the LLC user is unspecified.

6.2.3 L_DATA.confirm

a) Function

The L_DATA.confirm primitive is passed from the local LLC sublayer to the LLC user to convey the results of the previous L_DATA.request primitive. This primitive is a local confirmation, i.e. it does not imply that the remote LLC entity or entities have passed the associated indication primitive to the corresponding LLC user(s).

b) Semantics of the L_DATA.confirm primitive

The primitive shall provide parameters as follows.

```
L_DATA.confirm(
                IDENTIFIER
                TRANSFER_STATUS
                )
```

The TRANSFER_STATUS is used to indicate the completion of the transaction initiated by the previous L_DATA.request primitive.

```
TRANSFER_STATUS:[COMPLETE,NOT_COMPLETE]
```

c) Effect on receipt

The effect of receipt of this primitive by the LLC user is unspecified.

6.2.4 L_REMOTE.request

a) Function

The L_REMOTE.request primitive is passed from the LLC user to the LLC sublayer to request a single remote LLC entity to transmit an LSDU.

b) Semantics of the L_REMOTE.request primitive

The primitive shall provide parameters as follows.

```
L_REMOTE.request(
                  IDENTIFIER
                  DLC
                  )
```

The value of DLC equals the length of the data field of the requested data frame.

c) Effect on receipt

Receipt of this primitive causes the LLC sublayer to initiate the transfer of an LSDU by use of the remote data transfer service provided by the MAC sublayer (see 8.1.1).

6.2.5 L_REMOTE.indication

a) Function

The L_REMOTE.indication primitive is passed from the LLC sublayer to the LLC user to indicate the arrival of a request for data transmission of an LSDU.

b) Semantics of the L_REMOTE.indication primitive

The primitive shall provide parameters as follows.

```
L_REMOTE.indication(
                    IDENTIFIER
                    DLC
                    )
```

The identifier identifies the LSDU to be sent. The value of DLC equals the length of the data field of the requested data frame.

c) Effect on receipt

The effect of receipt of this primitive by the LLC user is unspecified.

6.2.6 L_REMOTE.confirm

a) Function

The L_REMOTE.confirm primitive is passed from the local LLC sublayer to the LLC user to convey the results of the previous L_REMOTE.request primitive. This primitive is a local confirmation, i.e. it does not imply that the remote LLC entity has passed the associated indication primitive to the corresponding LLC user.

b) Semantics of the L_REMOTE.confirm primitive

The primitive shall provide parameters as follows.

```
L_REMOTE.confirm(
                 IDENTIFIER
                 TRANSFER_STATUS
                 )
```

The TRANSFER_STATUS is used to indicate the completion of the transaction initiated by the previous L_REMOTE.request primitive.

```
TRANSFER_STATUS:[COMPLETE,NOT_COMPLETE]
```

c) Effect on receipt

The effect of receipt of this primitive by the LLC user is unspecified.

6.3 Structure of LLC frames

LLC frames are the data units that are exchanged between peer LLC entities (LPDU). The structure and format of the LLC data and remote frame are specified subsequently.

6.3.1 Specification of the LLC data frame

An LLC data frame is composed of three bit fields (see figure 4):

— Identifier field,

— Data Length Code (DLC) field,

— LLC data field.

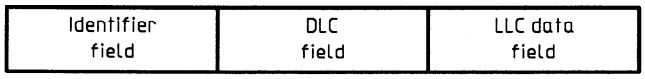


Figure 4 — LLC data frame

a) Identifier

The length of the identifier is 11 bits. The most significant bits (ID-10 to ID-4) shall not all be "1".

b) DLC field

The number of bytes in the data field is indicated by the data length code (see table 5). This data length code consists of 4 bits. The data field can be of length zero. The admissible number of data bytes for a data frame ranges from 0 to 8. Other values shall not be used.

Table 5 — Coding of the numbers of data bytes by the data length code

Number of data bytes	Data length code			
	DLC3	DLC2	DLC1	DLC0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0

c) Data field

The data field consists of the data to be transferred within a data frame. It can contains from 0 bytes to 8 bytes and each byte contains 8 bits.

6.3.2 Specification of the LLC remote frame

An LLC remote frame is composed of two bit fields (see figure 5):

— Identifier field,

— DLC field.

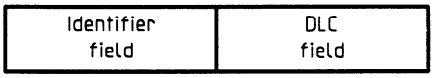


Figure 5 — LLC remote frame

The format of the LLC remote frame identifier is identical to the format of the LLC data frame identifier (see 6.3.1). There is no data field independent of the value of the data length code. This value is the data length code of the corresponding data frame.

6.4 Functions of the LLC sublayer

The LLC sublayer provides the following functions:

a) frame acceptance filtering,

b) overload notification,

c) recovery management.

6.4.1 Frame acceptance filtering

A frame transaction initiated at the LLC sublayer is a single, self-contained operation independent of previous frame transactions. The content of a frame is named by its identifier. The identifier does not indicate the destination of the frame but describes the meaning of the data. Each receiver decides by a frame acceptance filtering whether the frame is relevant to it or not.

6.4.2 Overload notification

The transmission of an overload frame will be initiated by the LLC sublayer if internal conditions of a receiver require delay of the next LLC data or LLC remote frame. At most, two overload frames may be generated to delay the next data frame or remote frame.

6.4.3 Recovery management

The LLC sublayer provides means for automatic retransmission of frames that have lost arbitration or that have been disturbed by errors during transmission. The frame transmission service will not be confirmed to the user before the transmission is successfully completed.

7 Interface between LLC and MAC

The MAC sublayer provides services to the local LLC for:

— acknowledged transfer of LLC frames,

— transmission of overload frames.

The interface service data from or to the LLC sublayer is described in 8.1.

8 Description of the MAC sublayer

The MAC (medium access control) sublayer represents the lower part of the OSI data link layer. It services the interface to the LLC sublayer and the physical layer, and comprises the functions and rules that are relevant to:

— encapsulation/decapsulation of the transmit/receive data,

— error detection and signalling,

— management of the transmit/receive medium access.

8.1 Services of the MAC sublayer

The services provided by the MAC sublayer allow the local LLC sublayer entity to exchange MAC service data units (MSDU) with the peer LLC sublayer entities. The MAC sublayer services are:

a) Acknowledged data transfer

This service provides means by which LLC entities can exchange MSDUs without the establishment of a data link connection. The data transfer can be point-to-point, multicast or broadcast.

b) Acknowledged remote data request

This service provides means by which an LLC entity can request another remote node to transmit an LSDU without the establishment of a data link connection. The remote LLC entity uses the MAC service "acknowledged data transfer" for the transmission of the requested data. In both cases acknowledgement of a service is generated by the MAC sublayer(s) of the remote node(s). Acknowledgement does not contain any data of the remote node user.

c) Overload frame transfer

This service provides means by which an LLC entity can initiate the transmission of an overload frame, a special fixed format LPDU, to cause the delay of the next data frame or remote frame.

8.1.1 Service primitive specification

The service primitives that the MAC sublayer provides to the LLC sublayer are given in table 6.

Table 6 — MAC sublayer service primitives

Acknowledged data transfer		
MA_DATA.request	MA_DATA.indication	MA_DATA.confirm
Acknowledged remote data request		
MA_REMOTE.request	MA_REMOTE.indication	MA_REMOTE.confirm
Overload frame transfer		
MA_OVLD.request	MA_OVLD.indication	MA_OVLD.confirm

8.1.1.1 MA_DATA.request

a) Function

The MA_DATA.request primitive is passed from the LLC sublayer to the MAC sublayer to request that an MSDU be sent to one or more remote MAC entities.

b) Semantics of the MA_DATA.request primitive

The primitive shall provide parameters as follows.

```
MA_DATA.request(
                IDENTIFIER
                DLC
                DATA
                )
```

The parameter DATA is insignificant for MAC data frames of data length zero.

c) Effect on receipt

Receipt of this primitive causes the MAC sublayer to prepare a protocol data unit by adding all MAC specific control information (SOF, RTR bit, reserved bits, CRC, "recessive" bit during ACK-Slot, EOF) to the MSDU coming from the LLC sublayer. The MAC PDU will be serialized and passed bit by bit as a service data unit to the physical layer for transfer to the peer MAC sublayer entity or entities.

8.1.1.2 MA_DATA.indication

a) Function

The MA_DATA.indication primitive is passed from the MAC sublayer to the LLC sublayer to indicate the arrival of an MSDU.

b) Semantics of the MA_DATA.indication primitive

The primitive shall provide parameters as follows.

```
MA_DATA.indication(
                   IDENTIFIER
                   DLC
                   DATA
                   )
```

The parameter DATA is insignificant if the associated MAC data frame is of data length zero. The arrival of an MSDU is indicated to the LLC sublayer only if it has been received correctly.

c) Effect on receipt

The effect of receipt of this primitive by the LLC sublayer is unspecified.

8.1.1.3 MA_DATA.confirm

a) Function

The MA_DATA.confirm primitive is passed from the local MAC sublayer to the LLC sublayer to convey the results of the previous MA_DATA.request primitive. This primitive is a remote confirmation, i.e. it indicates that the remote MAC entity or entities have passed the associated indication primitive to the corresponding user(s).

b) Semantics of the MA_DATA.confirm primitive

The primitive shall provide parameters as follows.

```
MA_DATA.confirm(
                IDENTIFIER
                TRANSMISSION_STATUS
                )
```

The TRANSMISSION_STATUS is used to indicate the success or failure of the previous MA_DATA.request primitive.

```
TRANSMISSION_STATUS: [SUCCESS,NO_SUCCESS]
```

Failures are either errors which occurred during transmission or loss of arbitration.

c) Effect on receipt

The effect of receipt of this primitive by the LLC sublayer is unspecified.

8.1.1.4 MA_REMOTE.request

a) Function

The MA_REMOTE.request primitive is passed from the LLC sublayer to the MAC sublayer to request a single remote MAC entity to transmit an MSDU.

b) Semantics of the MA_REMOTE.request primitive

The primitive shall provide parameters as follows.

```
MA_REMOTE.request(
                  IDENTIFIER
                  DLC
                  )
```

The identifier identifies the MSDU to be sent. The value of DLC equals the length of the data of the requested MSDU.

c) Effect on receipt

Receipt of this primitive causes the MAC sublayer to prepare a protocol data unit by adding all MAC specific control information (SOF, RTR bit, reserved bits, CRC, "recessive" bit during ACK-Slot, EOF) to the MSDU coming from the LLC sublayer. The MAC PDU will be serialized and passed bit by bit as a service data unit to the physical layer for transfer to the peer MAC sublayer entity or entities.

8.1.1.5 MA_REMOTE.indication

a) Function

The MA_REMOTE.indication primitive is passed from the MAC sublayer to the LLC sublayer to indicate the arrival of a request for transmission of an MSDU.

b) Semantics of the MA_REMOTE.indication primitive

The primitive shall provide parameters as follows.

```
MA_REMOTE.indication(
                    IDENTIFIER
                    DLC
                    )
```

The arrival of an MSDU transmission request is indicated to the LLC sublayer only if it has been received correctly.

c) Effect on receipt

The effect of receipt of this primitive by the LLC sublayer is unspecified.

8.1.1.6 MA_REMOTE.confirm

a) Function

The MA_REMOTE.confirm primitive is passed from the local MAC sublayer to the LLC sublayer to convey the results of the previous MA_REMOTE.request. This primitive is a remote confirmation, i.e. it indicates that the remote MAC entity or entities have passed the associated indication primitive to the corresponding user(s).

b) Semantics of the MA_REMOTE.confirm primitive

The primitive shall provide parameters as follows.

```
MA_REMOTE.confirm(
                  IDENTIFIER
                  TRANSMISSION_STATUS
                  )
```

The TRANSMISSION_STATUS is used to indicate the success or failure of the previous MA_REMOTE.request primitive.

```
TRANSMISSION_STATUS: [SUCCESS,NO_SUCCESS]
```

Failures are either errors which occurred during transmission or loss of arbitration.

c) Effect on receipt

The effect of receipt of this primitive by the LLC sublayer is unspecified.

8.1.1.7 MA_OVLD.request

a) Function

The MA_OVLD.request primitive is passed from the LLC sublayer to the MAC sublayer to request the transmission of a MAC overload frame (see 8.3.4). The overload frame is a fixed format frame and is completely constructed in the MAC sublayer.

b) Semantics of the MA_OVLD.request primitive

The primitive does not provide any parameter:

```
MA_OVLD.request(
                )
```

c) Effect on receipt

Receipt of this primitive causes the MAC sublayer to form an overload frame. The overload frame will be passed to the lower protocol layers for transfer to the peer MAC sublayer entities.

8.1.1.8 MA_OVLD.indication

a) Function

The MA_OVLD.indication primitive is passed from the MAC sublayer to the LLC sublayer to indicate that an overload frame has been received (see 8.3.4).

b) Semantics of the MA_OVLD.indication primitive

The primitive does not provide any parameters:

```
MA_OVLD.indication(
                   )
```

c) Effect on receipt

The effect of receipt of this primitive by the LLC sublayer is unspecified.

8.1.1.9 MA_OVLD.confirm

a) Function

The MA_OVLD.confirm primitive is passed from the MAC sublayer to the LLC sublayer to indicate that an overload frame has been sent. This confirmation is local, i.e. it does not imply that the remote peer protocol entities have received correctly the overload frame.

b) Semantics of the MA_OVLD.confirm primitive

The primitive shall provide parameters as follows.

```
MA_OVLD.confirm(
                TRANSMISSION_STATUS
                )
```

The TRANSMISSION_STATUS is used to indicate the success or failure of the previous MA_OVLD.request primitive.

```
TRANSMISSION_STATUS: [SUCCESS,NO_SUCCESS]
```

c) Effect on receipt

The effect of receipt of this primitive by the LLC sublayer is unspecified.

8.2 Functional model of the MAC sublayer architecture

The functional capabilities of the MAC sublayer are described by use of the functional model specified in ISO 8802-3 (see figure 6). In this model the MAC sublayer is divided into two fully independently-operating parts, i.e. the transmit and the receive parts. The functions of both transmit and receive parts are described below.

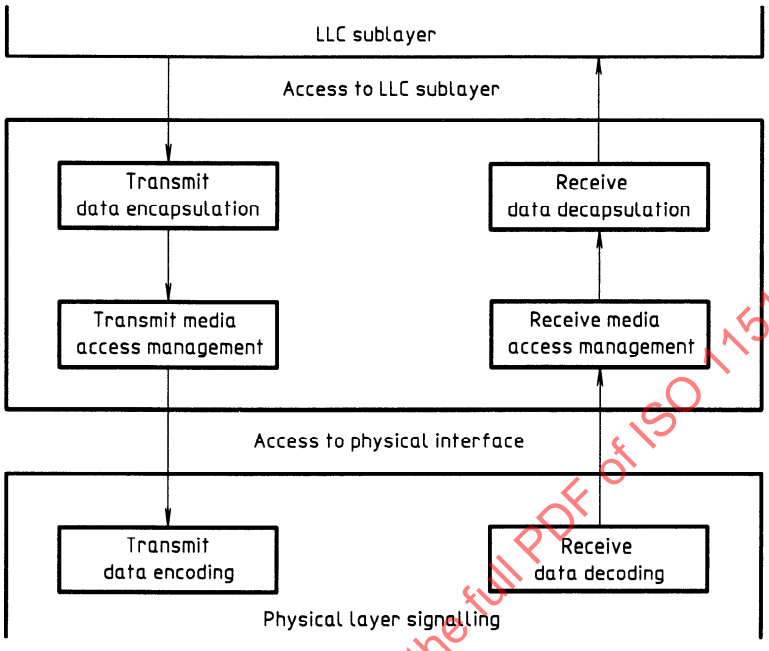


Figure 6 — Media access control functions

8.2.1 Frame transmission

a) Transmit data encapsulation:

1) acceptance of LLC frames and interface control information;

2) CRC sequence calculation;

3) construction of MAC frame by adding SOF, RTR bit, reserve bits, CRC, ACK and EOF to the LLC frame.

b) Transmit media access management:

1) initiation of the transmission process after recognizing bus idle (compliance with interframe space);

2) serialization of the MAC frame;

3) insertion of stuffbits (bit stuffing);

4) arbitration and passing into receiver mode in case of loss of arbitration;

5) error detection (monitoring, format check);

6) acknowledgement check;

7) recognition of an overload condition;

8) overload frame construction and initiation of transmission;

9) error frame construction and initiation of transmission;

10) presentation of a serial bit stream to the physical layer for transmission.

8.2.2 Frame reception

a) Receive media access management:

1) reception of a serial bit stream from the physical layer;

2) deserialization and recompiling of the frame structure;

3) deletion of stuffbits (bit destuffing);

4) error detection (CRC, format check, stuff rule check);

5) transmission of acknowledgement;

6) error frame construction and initiation of transmission;

7) recognition of an overload condition;

8) reactive overload frame construction and initiation of transmission.

b) Receive data decapsulation:

1) removal of the MAC specific information from the received frame;

2) presentation of the LLC frame and interface control information to the LLC sublayer.

8.3 Structure of MAC frames

Data transmission and reception between nodes in a CAN system is manifested and controlled by four different frame types:

— a data frame carries data from a transmitter to the receiver;

— a remote frame is transmitted by a node to request the transmission of the data frame with the same identifier;

— an error frame is transmitted by any node on detecting a bus error;

— an overload frame is used to provide for an extra delay between the preceding and succeeding data frames or remote frames.

Data frames and remote frames are separated from preceding frames by an interframe space.

8.3.1 Specification of the MAC data frame

A MAC data frame is composed of seven different bit fields (see figure 7);

— Start Of Frame (SOF),

— arbitration field,

— control field (two reserve bits + DLC field),

— data field,

— CRC field,

— ACK field,

— End Of Frame (EOF).

Figure 7 — MAC data frame

a) Start Of Frame (SOF) marks the beginning of data and remote frames. It consists of a single "dominant" bit. A node is only allowed to start transmission when the bus is idle (see bus idle in 8.3.5). All nodes have to synchronize to the leading edge caused by start of frame of the node starting transmission first.

b) Arbitration field is composed of the identifier field, passed from the LLC sublayer, and the RTR (remote transmisssion request) bit. The value of the RTR bit is "0" in a MAC data frame.

c) Control field consists of six bits. It includes two bits reserved for future expansion followed by data length code (see 6.2). Receivers accept "0" and "1" bits as reserved bits in all combinations. Until the function of the reserved bits are defined, the transmitter will send only "0" bits.

d) Data field is equivalent to the LLC data field (see 6.3.1).

e) CRC field contains the CRC sequence followed by a CRC delimiter.

1) CRC sequence, in which the frame check sequence is derived from a cyclic redundancy check (BCH-code) best suited for frames with bit counts less than 127 bits. In order to carry out the CRC calculation, the polynomial to be divided is defined as the polynomial, the coefficients of which are given by the destuffed bit stream consisting of start of frame, arbitration field, control field, data field (if present), and, for the 15 lowest coefficients, by 0. This polynomial is divided (the coefficients are calculated modulo-2) by the generator-polynomial:

$$X^{15}+X^{14}+X^{10}+X^{8}+X^{7}+X^{4}+X^{3}+1$$

The remainder of this polynomial division is the CRC sequence transmitted over the bus. In order to implement this function, a 15-bit shift register CRC_RG(14:0) can be used. If NXTBIT denotes the next bit of the bit-stream given by the destuffed bit sequence from start of frame until the end of the data field, the CRC sequence is calculated as follows.

```
CRC_RG(14:0)=(0,...,0;                    //initialize shift register
REPEAT
   CRCNXT = NXTBIT EXOR CRC_RG(14);
   CRC_RG(14:1) = CRC_RG(13:0);           //shift left by one position
   CRC_RG(0) = 0;
   IF CRCNXT THEN
   CRC_RG(14:0) = CRC_RG(14:0) EXOR (4599hex);
   ENDIF
UNTIL (NXTBIT = End of data or there is an error condition).
```

After the transmission/reception of the last bit of the data field, CRC_RG contains the CRC sequence.

2) CRC delimiter. The CRC sequence is followed by the CRC delimiter which consists of a single "recessive" bit.

f) ACK field, two bits long and containing the ACK slot and the ACK delimiter. In the ACK field the transmitting node sends two "recessive" bits.

   1) ACK slot. All nodes that have received the matching CRC sequence send an acknowledgement within the ACK slot by overwriting the "recessive" bit of the transmitter by a "dominant" bit.

   2) ACK delimiter, the second bit of the ACK field and necessarily a "recessive" bit. As a consequence, the ACK slot is surrounded by two "recessive" bits (CRC delimiter, ACK delimiter).

g) End Of Frame. Each data frame and remote frame is delimited by a flag sequence consisting of seven "recessive" bits.

8.3.2 Specification of the MAC remote frame

A node acting as a receiver for certain data can initiate the transmission of the respective data by its source node by sending a remote frame. A remote frame is composed of six different bit fields (see figure 8):

— Start Of Frame (SOF);

— arbitration field;

— control field (two reserve bits + DLC field);

— CRC field;

— ACK field;

— End Of Frame (EOF).

Figure 8 — MAC remote frame

The arbitration field is composed of the identifier field, passed from the LLC sublayer, and the RTR remote transmission request) bit. The value of the RTR bit in a MAC remote frame is "1".

The bit fields Start Of Frame (SOF), control field, CRC field, ACK field and End Of Frame (EOF) are equivalent to the corresponding bit fields of the MAC data frame (see 8.3.1).

8.3.3 Specification of the error frame

The error frame consists of two different fields. The first field is given by the superposition of error flags contributed from different nodes. The following second field is the error delimiter.

a) Error flag

   There are two forms of error flag: the active error flag and the passive error flag.

   — The active error flag consists of six consecutive "dominant" bits.

   — The passive error flag consists of six consecutive "recessive" bits. Some or all bits of a passive error flag may be overwritten by "dominant" bits from other nodes.

An "error-active" node detecting an error condition signals this by transmission of an active error flag. The error flag's form violates the rule of bit stuffing or destroys the bit field requiring fixed form. As a consequence, all other nodes also detect an error condition and likewise start transmission of an error flag. So the sequence of "dominant" bits, which actually can be monitored on the bus, results from a superposition of different error flags transmitted by individual nodes. The total length of this sequence varies between a minimum of six and a maximum of twelve bits.

Passive error flags initiated by a transmitter cause errors at the receivers when they start in a frame field which is encoded by the method of bit stuffing, because they then lead to stuff errors detected by the receivers. This requires, however, that such an error flag does not start during arbitration and another node continues transmitting, or that it starts very few bits before the end of the CRC sequence and the last bits of the CRC sequence happen to be all "recessive". Passive error flags initiated by receivers are not able to prevail in any activity on the bus line. Therefore, "error-passive" receivers always have to wait for six subsequent equal bits after detecting an error condition, until they have completed their error flag.

b) Error delimiter

The error delimiter consists of eight "recessive" bits.

After transmission of an error flag, each node sends "recessive" bits and monitors the bus until it detects a "recessive" bit. It then starts transmitting seven more "recessive" bits.

8.3.4 Specification of the overload frame

There are two types of overload frames having the same format:

a) LLC-requested overload frame

This overload frame will be requested by LLC sublayer to indicate an internal overload situation (see 8.10).

b) Reactive overload frame

The transmission of the active overload frame will be initated by the MAC sublayer upon certain error conditions (see 8.10).

The overload frame contains two bit fields: overload flag and overload delimiter. The overall form of the overload flag corresponds to that of the active error flag. The overload delimiter is of the same form as the error delimiter.

a) Overload flag consists of six "dominant" bits. The overload flag form destroys the fixed form of the intermission field (see 8.3.5). As a consequence, all other nodes also detect an overload condition and their part start transmission of an overload flag.

b) Overload delimiter consists of eight "recessive" bits. After transmission of an overload flag, every node monitors the bus until it detects a "recessive" bit. At this point in time every node has finished sending its overload flag and all nodes start transmission of seven more "recessive" bits simultaneously, to complete the eight-bit-long overload delimiter.

8.3.5 Specification of the interframe space

Data frame and remote frames are separated from preceding frames, of whatever type (data frame, remote frame, error frame, overload frame), by a bit field called an interframe space. In contrast to this, overload frames and error frames are not preceded by an interframe space, and multiple overload frames are not separated by an interframe space.

The interframe space contains the bit fields "intermission" and "bus idle", and "suspend transmission" for error-passive nodes which have been a transmitter of the previous frames (see figures 9 and 10).

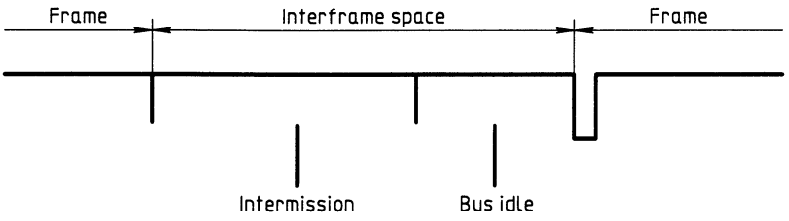


Figure 9 — Interframe space for nodes which are not "error passive" or have been a receiver of the previous frame

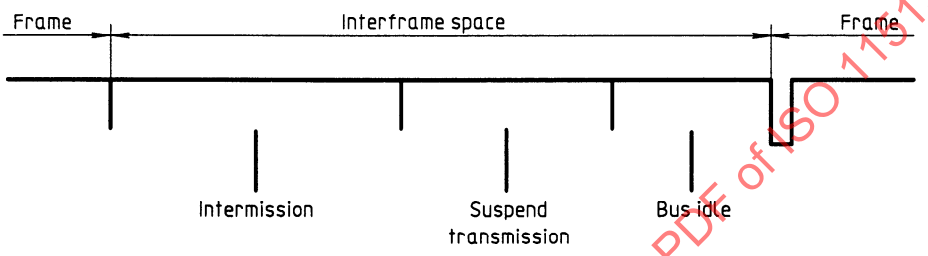


Figure 10 — Interframe space for "error-passive" nodes which have been a transmitter of the previous frame

a) Intermission consists of three "recessive" bits. During intermission no node is allowed to start transmission of a data frame or remote frame. The only action to be taken is signalling an overload condition.

b) Bus idle. The period of bus idle may be of arbitrary length. The bus is recognized to be free and any node can access the bus in order to transmit. A frame which is pending for transmission during the transmisssion of another frame is started in the first bit following intermission. The detection of a "dominant" bit on the bus during bus idle is interpreted as Start Of Frame.

c) Suspend transmission. After an "error-passive" node has transmitted a frame, it sends eight "recessive" bits following intermission before it is allowed to transmit a further frame. If meanwhile a transmission (caused by another node) starts, the node will become receiver of this frame.

8.4 Frame coding

The frame segments start of frame, arbitration field, control field, data field and CRC sequence are coded by the method of bit stuffing. Whenever a transmitter detects five consecutive bits (including stuffbits) of identical value in the bit stream to be transmitted, it automatically inserts a complementary bit in the bit stream actually transmitted (see figure 11).

Destuffed bit stream	1 0 0 0 0 0 a b c	0 1 1 1 1 1 a b c
Stuffed bit stream	1 0 0 0 0 0 1 a b c	0 1 1 1 1 1 0 a b c
	a, b, c ∈ {0,1}	

Figure 11 — Bit stuffing

The remaining bit fields of the data frame or remote frame (CRC delimiter, ACK field and end of frame) are of fixed form and are not stuffed. The error frame and the overload frame are of fixed form as well, and are not coded by the method of bit stuffing. The bit stream in a frame is coded according to the Non-Return-to-Zero (NRZ) method. This means that the generated bit level is constant during the total bit time.

8.5 Order of the bit transmission

A frame shall be transferred bit field by bit field, starting with its SOF field. Within a field the most significant bit shall be transmitted first (see figure 12).

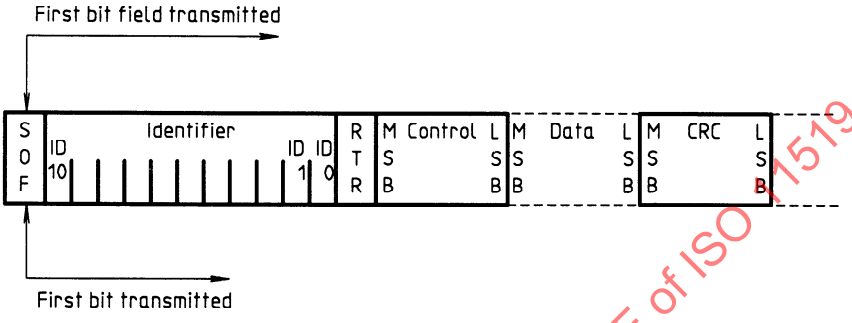


Figure 12 — Order of bit transmission

8.6 Frame validation

The point in time at which a frame is taken to be valid is different for the transmitter and the receiver of the frame.

a) Transmitter

The frame is valid for the transmitter if there is no error until the end of EOF. If a frame is corrupted, recovery is processed as described in 6.4.3.

b) Receiver

The frame is valid for the receiver if there is no error until the next to last bit of EOF.

8.7 Medium access method

This subclause describes the functions and characteristics that are related to the medium access method of CAN.

8.7.1 Multimaster

Every node transmitting a data frame or a remote frame is bus master during transmission.

8.7.2 Bus free detection

The bus is considered to be free by any node after having detected that the bit field intermission has not been interrupted by a "dominant" bit.

8.7.3 Bus access

An "error-active" node may access the bus as soon as the bus is free. An "error-passive" node, which is the receiver of the current or previous frame, may access the bus as soon as the bus is free. An "error-passive" node, which is the transmitter of the current frame or has been the transmitter of the previous frame, may access the bus as soon as suspend transmission is finished, provided that no other node has started transmission meanwhile. Whenever several nodes start transmitting simultaneously, that node transmitting the frame with the highest priority at this point of time will become the only bus master. The mechanism used to resolve the resulting bus access conflict is contention-based arbitration.

8.7.4 Transmission of MAC frames

MAC data frames and MAC remote frames may be started when the node is allowed to access the bus according to 8.7.3. A MAC error frame is transmitted as specified in 8.9. A MAC overload frame is transmitted as specified in 8.10.

8.7.5 Contention-based arbitration

During arbitration every transmitter compares the level of the bit transmitted with the level that is monitored on the bus. If these levels are equal, the node may continue to send. When a "recessive" level is sent and a "dominant" level is monitored, the node has lost arbitration and must withdraw without sending one more bit. When a "dominant" level is sent and a "recessive" level is monitored, the node detects a bit error.

8.7.6 Frame priority

Contention-based arbitration is performed on the identifier and on the RTR bit following the identifier. Among two frames with different identifiers, the higher priority is assigned to the frame whose identifier has the lower binary value. If a data frame and a remote frame with the same identifier are initiated at the same time, the data frame has a higher priority than the remote frame. This is achieved by assigning according values to the RTR bits.

8.7.7 Collision resolution

In addition to the principle that transmission may be initiated only when the bus is free, further principles for the resolution of collision exist:

— within one system each information must be assigned a unique identifier;

— a data frame with a given identifier and a non-zero data length code may only be initiated by one node;

— remote frames may only be transmitted with a system-wide determined data length code, which is the data length code of the corresponding data frame. Simultaneous transmission of a remote frame with identical identifiers and different data length codes leads to unresolvable collisions.

8.8 Error detection

The MAC sublayer provides the following mechanisms for error detection:

— monitoring;

— stuff rule check;

— frame check;

— 15 bit cyclic redundancy check;

— acknowledgement check.

There are five different error types (which are not mutually exclusive):

a) *Bit error*

A node that is sending a bit on the bus also monitors the bus. A bit error is detected at that bit time, when the bit value that is monitored differs from the bit value sent.

Exceptions: A dominant bit does not lead to a bit error when a recessive information bit is sent during arbitration, or a recessive bit is sent during ACK slot. A node does not interpret a detected dominant bit as an error, when sending a passive error flag.

b) *Stuff error*

A stuff error is detected at the bit time of the sixth consecutive equal bit level in a frame field that should be coded by the method of bit stuffing.

c) *CRC error*

The CRC sequence consists of the result of the CRC calculation of the transmitter. The receivers calculate the CRC in the same way as the transmitter. A CRC error is detected when the calculated CRC sequence is not equal to the received one.

d) *Form error*

A form error is detected when a fixed-form bit field contains one or more illegal bits.

Exceptions: a receiver does not interpret a monitored "dominant" bit as a form error when it is the last bit of EOF.

e) *Acknowledgement error*

An acknowledgement error is detected by a transmitter whenever it does not monitor a "dominant" bit during ACK slot.

Whenever one of these errors is detected, the LLC sublayer will be informed. As a consequence, the MAC sublayer initiates the transmission of an error flag.

8.9 Error signalling

Whenever a bit error, stuff error, form error or acknowledgement error is detected by any node, transmission of an error flag is started by the respective node at the next bit.

Whenever a CRC error is detected, transmission of an error frame starts at the bit following the ACK delimiter, unless an error frame for another error condition has already been started.

8.10 Overload signalling

The following conditions lead to the transmission of an overload frame.

a) LLC-requested overload frame (initiated by the LLC sublayer)

Internal conditions of a receiver, which require a delay of the next MAC data frame or MAC remote frame.

b) Reactive overload frame (initiated by the MAC sublayer)

— Detection of a "0" bit during intermission,

— Detection of a "0" bit in the last bit of EOF by a receiver.

An LLC-requested overload frame is only allowed to be started at the first bit of an expected intermission, whereas reactive overload frames start one bit after detecting the "0" bit due to condition b). The start of reactive overload frames due to condition b) is allowed but not required to be implemented.

At most two LLC overload frames may be generated to delay the next MAC data frame or MAC remote frame.

9 LLC and MAC sublayer conformance

For an implementation to be in conformance it must comply with all specifications and values given in this part of ISO 11519.

10 Description of the physical layer

The physical layer is an electrical circuit realization that connects an Electronic Control Unit (ECU) to a bus. The total number of ECUs will be limited by the electric loads on the bus line. The physical layer specified in this part of ISO 11519 is for low-speed applications (up to 125 kbits/s).

10.1 Functional model of the physical layer

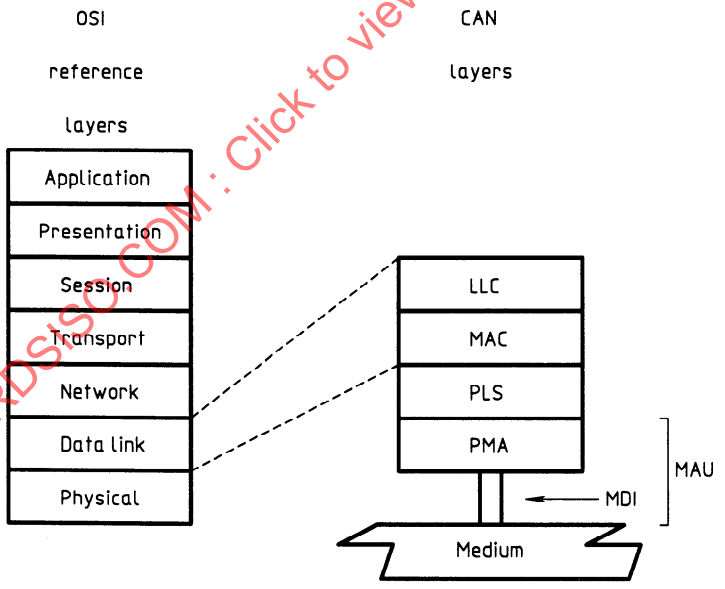
The physical layer is modelled according to the LAN standard specification as in ISO 8802-3 (see figure 13). The physical layer is divided into three parts:

a) Physical Signalling (PLS) encompasses those functions related to bit representation, timing and synchronization.

The Medium Access Unit (MAU) denotes the functional part of the physical layer used to couple the node to the transmission medium. The MAU consists of the following parts:

b) Physical Medium Attachment (PMA) sublayer, encompassing the functional circuitry for bus line transmission/reception and may provide means for bus failure detection.

c) Medium-Dependent Interface (MDI), encompassing the mechanical and electrical interfaces between the physical medium and the MAU.



Key

LLC = Logical Link Control
MAC = Medium Access Control
PLS = Physical Signalling
MAU = Medium Access Unit
MDI = Medium-Dependent Interface
PMA = Physical Medium Attachment

Figure 13 — Model of the physical layer architecture

10.2 Services of the physical layer

The services provided by the physical layer allow the local MAC sublayer entity to exchange data bits with peer MAC sublayer entities. The physical layer provides the following service primitives to the MAC sublayer.

```
PLS_DATA.request,
PLS_DATA.indicate.
```

10.2.1 PLS_DATA.request

The PLS_DATA.request primitive is passed from the MAC sublayer to the physical layer to request for transmission of a "dominant" or "recessive" bit. The primitive provides the following parameter.

```
PLS_DATA.request(
                OUTPUT_UNIT
                )
```

The OUTPUT_UNIT parameter can assume one of two values: DOMINANT or RECESSIVE.

10.2.2 PLS_DATA.indicate

The PLS_DATA.indicate primitive is passed from the physical layer to the MAC sublayer in order to indicate the arrival of a "dominant" or "recessive" bit. The primitive provides the following parameter.

```
PLS_DATA.indicate(
                 INPUT_UNIT
                 )
```

The INPUT_UNIT parameter can assume one of the two values each representing a single bit: DOMINANT or RECESSIVE.

10.3 Physical Signalling (PLS) sublayer specification

10.3.1 Bit encoding/decoding

10.3.1.1 Definition of the bit time

The bit time t_B is defined as the duration of one bit. Bus management functions executed within the bit time frame, such as ECU synchronization behaviour, network transmission delay compensation and sample point positioning, are defined by the programmable bit timing logic of the CAN protocol IC (integrated circuit).

a) Nominal bit rate

The nominal bit rate gives the number of bits per second transmitted in the absence of resynchronization by an ideal transmitter.

b) Nominal bit time

Nominal bit time = 1 nominal bit rate. The nominal bit time can be thought of as being divided into separate non-overlapping time segments. The segments form the bit time as shown in figure 14:

— synchronization segment (SYNC_SEG);

— propagation time segment (PROP_SEG);

— phase buffer segment 1 (PHASE_SEG1);

— phase buffer segment 2 (PHASE_SEG2).

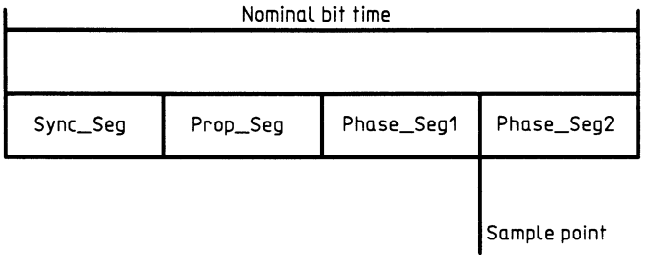


Figure 14 — Partition of the bit time

1) SYNC_SEG

This part of the bit time is used to synchronize the various ECUs on the bus. An edge is expected within this segment.

2) PROP_SEG

This part of the bit time is used to compensate physical delay times within the network. These delay times consist of the signal propagation time on the bus line and the internal delay time of the ECUs.

3) PHASE_SEG1, PHASE_SEG2

These phase buffer segments are used to compensate for edge phase errors. These segments can be lengthened or shortened by resynchronization.

c) Sample point

The sample point is the point in time at which the bus level is read and interpreted as the value of that respective bit. Its location is at the end of PHASE_SEG1.

d) Information processing time

The information processing time is the time segment starting with the sample point reserved for calculation of the subsequent bit level.

10.3.1.2 Programming of the bit time

a) Time quantum

The time quantum is a fixed unit of time derived from the oscillator period. There exists a programmable prescaler, with integral values, ranging from at least 1 to 32. Starting with the minimum time quantum, the time quantum can have a length of:

$$\text{Time quantum} = m \times \text{minimum time quantum}$$

where m is the value of the prescaler.

b) Length of time segments

— SYNC_SEG is one time quantum long;

— PROP_SEG is programmable to be 1, 2, ..., 8 time quanta long;

— PHASE_SEG1 is programmable to be 1, 2, ..., 8 time quanta long;

— PHASE_SEG2 is the maximum of PHASE_SEG1 and the information processing time;

— the information processing time is less than or equal to two time quanta.

The total number of time quanta in a bit time has to be programmable at least from 8 to 25.

The frequencies of the oscillators in the different ECUs shall be coordinated in order to provide a system-wide specified time quantum.

10.3.2 Synchronization

a) Synchronization rules

Hard synchronization and resynchronization are two forms of synchronization. They obey the following rules:

1) only one synchronization within one bit time is allowed;

2) an edge will be used for synchronization only if the value detected at the previous sample point (previously read bus value) differs from the bus value immediately after the edge;

3) hard synchronization is performed during bus idle whenever there is a "recessive" to "dominant" edge;

4) all other "recessive" to "dominant" edges (and optionally "dominant" to "recessive" edges in case of low bit rates) fulfilling rules 1) and 2) will be used for resynchronization, with the exception that a transmitter will not perform resynchronization as a result of a "recessive" to "dominant" edge with a positive phase error (see below), if only "recessive" to "dominant" edges are used for resynchronization.

b) Resynchronization jump width

As a result of resynchronization, PHASE_SEG1 may be lengthened or PHASE_SEG2 may be shortened. The amount of lengthening or shortening of the phase buffer segments has an upper limit, given by the re-synchronization jump width. The resynchronization jump width shall be programmable between 1 and min. (4,PHASE_SEG1). Clocking information is derived from transitions from one bit value to the other. The property that (due to the bit stuffing) only a fixed maximum number of successive bits have the same value, provides the possibility of resynchronizing a bus unit to the bit stream during a frame. The maximum length between two transitions which can be used for resynchronization is 29 bit times.

c) Phase error of a synchronization edge

The phase error of an edge is given by the position of the edge relative to SYNC_SEG, measured in time quanta. The sign of phase error is defined as follows:

$e = 0$ if the edge lies within SYNC_SEG;

$e > 0$ if the edge lies before the sample point;

$e < 0$ if the edge lies after the sample point.

10.3.2.1 Hard synchronization

After a hard synchronization, the bit time is restarted by each bit timing logic unit with SYNC_SEG. Thus hard synchronization forces the edge which has caused the hard synchronization to lie within the synchronization segment of the restarted bit time.

10.3.2.2 Bit resynchronization

The effect of resynchronization is the same as that of hard synchronization, when the magnitude of the phase error of the edge which causes resynchronization is less than or equal to the programmed value of the resynchronization jump width. When the magnitude of the phase error is larger than the resynchronization jump width,

— and if the phase error e is positive, then PHASE_SEG1 is lengthened by an amount equal to the resynchroniz-ation jump width;

— and if the phase error e is negative, then PHASE_SEG2 is shortened by an amount equal to the resynchronization jump width.

10.4 PLS-PMA interface specification

10.4.1 PLS to PMS messages

a) Output message

The PLS sublayer sends an output message to the PMA sublayer whenever it receives an OUTPUT_UNIT from the MAC sublayer. The output message causes the PMA to send a "dominant" or "recessive" bit.

b) BUS_OFF message

The PLS sublayer sends a BUS_OFF message to the PMA sublayer whenever it receives a BUS_OFF_REQUEST from the supervisor (see 11.1).

c) BUS_OFF_RELEASE message

The PLS sublayer sends a BUS_OFF_RELEASE message to the PMA sublayer whenever it receives a BUS_OFF_RELEASE_REQUEST from the supervisor (see 11.1).

10.4.2 PMA to PLS message

The PMA sublayer sends an input message to the PLS sublayer whenever the MAU has received a bit from the medium. The input signal indicates to the PLS the arrival of a "dominant" or "recessive" bit.

10.5 Description of the low-speed medium access unit (LS-MAU)

10.5.1 Physical medium attachment sublayer specification

10.5.1.1 Functional description

As shown in figure 15, the bus line is terminated to one termination network. A "recessive" bit is transmitted on the bus line if the transistor pairs of all ECUs are switched off. In this case the voltages on the bus line are generated by the bias voltage of the termination network. A "dominant" bit is sent to the bus line if the TR0 is switched to ground and TR1 is switched to V_{CC} by at least one ECU. This induces a current flow into the termination network and consequently a polarity change of the differential voltages between two wires of the bus line. These two bit states can be detected by a simple resistor network which transforms the differential voltages of the bus line to the corresponding "recessive" and "dominant" voltage levels at the comparator input of the CAN modules.

NOTE 1 Figure 15 describes a principal electrical realization of the coupling circuitry for both 12 V and 24 V nominal battery voltage. For 24 V nominal battery voltage, however, external protection circuitries are recommended.

10.5.1.2 Electrical specification

All data given in tables 7 to 11 are independent of a specific implementation. The parameters specified in these tables shall be fulfilled for each individual ECU within the ambient temperature range specified. The parameters are chosen such that a maximum number of 20 ECUs may be connected to one bus line. The values specified refer to a singular physical medium with unshielded wires.

10.5.1.2.1 Bus levels

The bus line can have one of the two logical states "recessive" or "dominant". The "recessive" state is represented by $V_{diff} = 1{,}5$ V, the absolute values are symmetrical around the mean voltage level. In the dominant state the differential voltage changes to $V_{diff} = -3$ V (see figure 16).

During arbitration, more than one ECU can start transmitting a "dominant" bit at the same time on the bus line. In this case V_{diff} during arbitration exceeds V_{diff} during single operation. Single operation means that the bus line is driven by one ECU only.

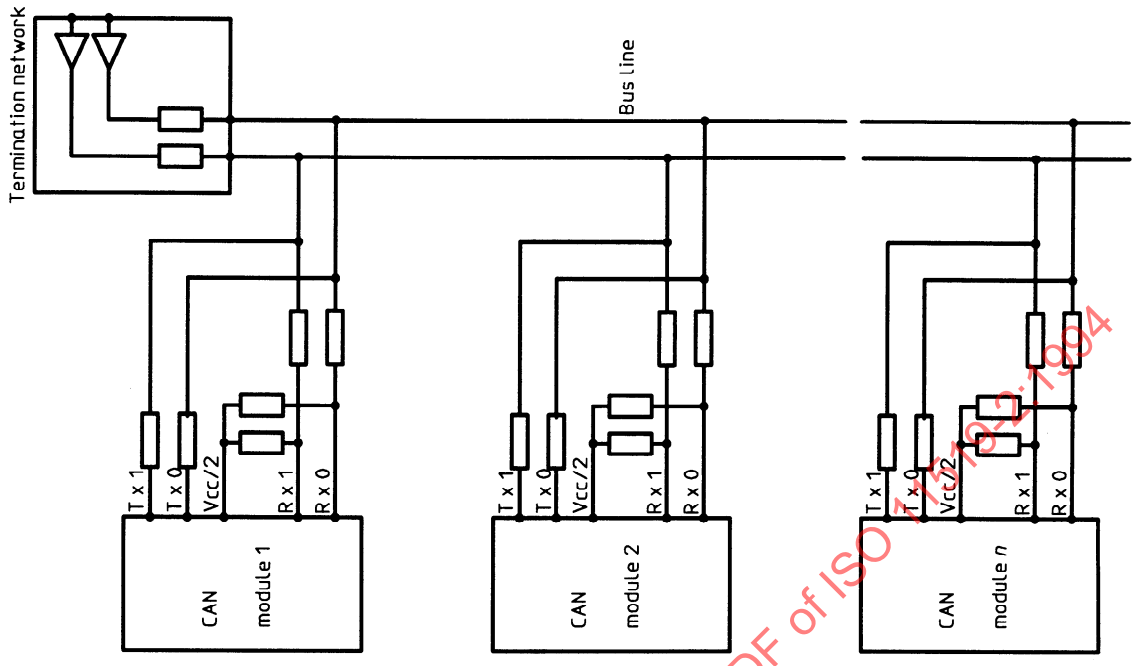


Figure 15 — Suggested electrical interconnection

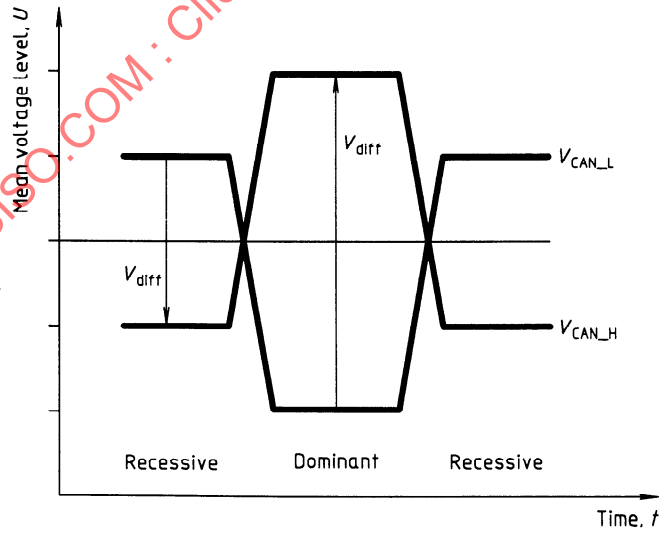


Figure 16 — Physical bit representation

10.5.1.2.2 Electronic control unit (ECU)

The parameters given in table 7 shall be tested at the CAN_L and CAN_H contacts of each ECU (see 10.5.4).

Table 7 — Ratings of V_{CAN_L} and V_{CAN_H} of an ECU

Nominal battery voltage V	Notation	Voltage min. V	Voltage max. V
12	V_{CAN_L}	− 16,0	16,0
	V_{CAN_H}	− 16,0	16,0
24	V_{CAN_L}	− 32,0	32,0
	V_{CAN_H}	− 32,0	32,0

NOTES

1 Undisturbed operation need not be guaranteed.

2 No destruction of transmitter circuit occurs.

3 Time is unrestricted.

Table 8 — DC parameters for the recessive state of an ECU connected to the termination network via the bus line

Parameter	Notation	Unit	Value min.	Value nominal	Value max.
Bus voltage	V_{CAN_L}	V	3,10	3,25	3,40
	V_{CAN_H}	V	1,60	1,75	1,90
Differential bus voltage	V_{diff} 1)	V	0,3	1,5	
Differential internal resistance	R_{diff}	kΩ	360		
Internal resistor	R_{in}	kΩ	180		

1) The differential bus voltage is determined by the input load of all ECUs during the recessive state. Therefore V_{diff} decreases slightly as the number of ECUs connected to the bus increases.

Table 9 — DC parameters for the dominant state of an ECU connected to the termination network via the bus line

Parameter	Notation	Unit	Value		
			min.	nominal	max.
Bus voltage	V_{CAN_L}	V	0,00	1,00	1,15
	V_{CAN_H}	V	3,85	4,00	5,00
Differential bus voltage	V_{diff} 1)	V	− 0,3	− 3,00	

1) Reception must be ensured within the common mode voltage range specified in table 11.

Table 10 — AC parameters of an ECU connected to the bus line

Parameter	Notation	Unit	Value			Conditions
			min.	nominal	max.	
Bit time	t_B	µs	8,0			1)
Internal load capacitance	C_{in}	pF		20		2) 3) 4)
	C_{diff}			10		

1) The maximum baud rate depends on:

— the total capacitive load of the nodes: $C_n \times n$;
— the capacitive load of the cable: $l \times c_c$;
— the internal resistance of the termination network R_{in} (see 10.5.3.1.1).

Maximum baud rate = $1/t_{Bmin.} = k \times l/(C \times R_{in})$; where:

$C = n \times c_n + 1 \times c_c$;
n = number of nodes;
l = cable length (in metres);
$c_n = C_{in} + 2 \times C_{diff}$;
$c_c = c_H + 2 \times c_{HL}$ or $c_c = c_L + 2 \times c_{HL}$ (see 10.5.3.1);
$k = 1/3$.
k is a function of the sample point.
c_n and c_c depend on the details of the implementation.

2) The minimum bit time corresponds to the maximum bit rate of 125 kbit/s. The lower end of the bit rate depends on the protocol IC.

3) In addition to a realization with the low internal capacitances, a bus connection with an inductance as low as possible must be realized. This is mainly important for high bit rates.

4) The maximum capacitance measured on each contact relative to ground is:

$C_{busin} = C_{in} + 2 \times C_{diff}$

10.5.1.2.3 Common mode voltages

The common mode voltage is defined as:

$$V_{com} = (V_{CAN\_L} + V_{CAN\_H})/2$$

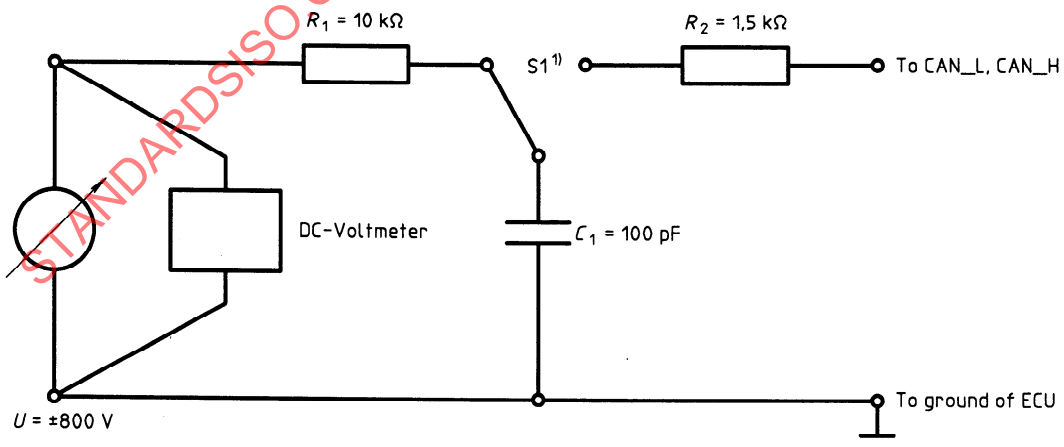
The common mode voltage must be ensured within the ratings specified in table 11.

Table 11 — Common mode voltages

Parameter	Notation	Unit	Value		
			min.	nominal	max.
Common mode voltage	V_{com}	V	– 1	2,5	6

10.5.1.2.4 Electrostatic discharge (ESD)

Due to ESD, the CAN_L and CAN_H of each ECU shall be tested against pulses as specified in figure 17. This ESD test shall be performed with an ECU disconnected from the bus line.



1) The switch S1 must be of mercury type. This prevents a high voltage peak down during the change of the switch position.

Figure 17 — EST test, generation of ESD pulses

10.5.2 Medium-Dependent Interface (MDI) specification

10.5.2.1 Connector parameters

The parameters specified in table 12 shall be met if a physical connector is chosen to attach the ECUs to the bus line. The aim of this part of the specification is to standardize the most important electrical parameters and not to define mechanical and material parameters.

Table 12 — Connector parameters

Parameter	Notation	Unit	Value		
			min.	nominal	max.
Voltage	U	V			32
Current	I	mA	0	1	80
Transmission	R_T	mΩ		7	
NOTE — The capacitances of the connector are a part of the internal load capacitances (see AC parameters).					

10.5.3 Physical medium specification

The specifications given shall be met by the physical cables chosen for the CAN bus line. The aim of these specifications is to standardize the most important electrical parameters and not to specify mechanical and material parameters of the cable.

10.5.3.1 Bus line

Table 13 — Physical media parameters

Parameter	Notation	Unit	Value			Conditions
			min.	nominal	max.	
Signal wire				2		1)
Length-related resistance	r	mΩ/m		90		
Length-related capacitances between:						The total capacitance of the signal wires is a limiting factor of the max. baud rate.
CAN_L and ground	c_L	pF/m		30		
CAN_H and ground	c_H	pF/m		30		
CAN_L and CAN_H	c_{LH}	pF/m		30		
1) The signal wires should be routed in parallel.						

10.5.3.1.1 Termination network

Each bus line is connected to a bias voltage of the termination network (see figure 18) which represents the recessive state of V_{CAN_L} and V_{CAN_H}. Each bias voltage is a power supply with an internal resistance $R_{in} = 2{,}2\ k\Omega \pm 5\ \%$. The termination network is needed only once per bus system. It can be placed anywhere on the bus line, usually inside one ECU. Mainly this network is responsible for the bus levels in the recessive state.

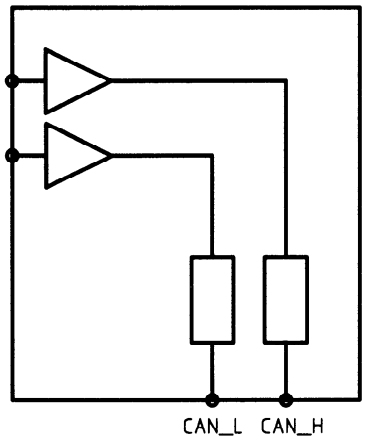


Figure 18 — Termination network

10.5.4 Conformance test

The subsequent figures and formulae show in principle how the electrical parameters specified in this paragraph may be tested.

10.5.4.1 Recessive output of the termination network

Recessive output of the termination network, V_{CAN_L} and V_{CAN_H}, (see figure 19) is measured unloaded while the bus is idle. V_{diff} is given by:

$$V_{diff} = V_{CAN\_L} - V_{CAN\_H}$$

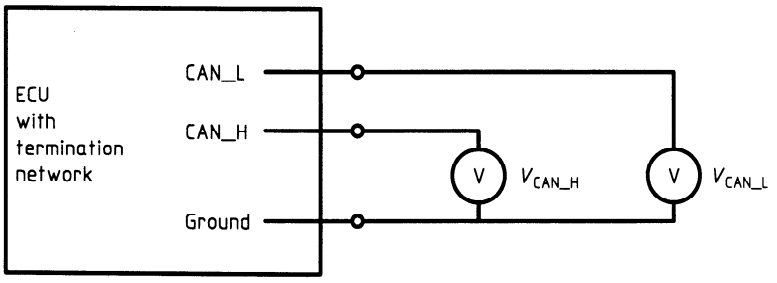


Figure 19 — Measurements of V_{CAN_L} and V_{CAN_H} during the bus idle state

10.5.4.2 Internal termination resistance of CAN_L and CAN_H

The ground-related internal termination resistance, R_{in}, of CAN_L and CAN_H is measured as shown in figure 20, with the ECU protocol IC set to bus idle.

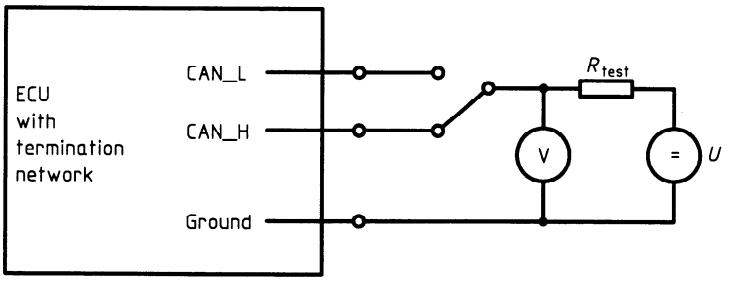


Figure 20 — Measurement of R_{in} of ECU with termination network

For U and R_{test}, values of 2,5 V and 2,2 kΩ respectively are recommended. The internal termination resistance of CAN_L and CAN_H is calculated by:

$$R_{in} = R_{test} \times (V_{CAN\_L.H} - V)/(V - U)$$

where V_{CAN_L} and V_{CAN_H} are the open circuit voltages according to figure 19.

10.5.4.3 Dominant output of an ECU

10.5.4.3.1 ECU with an integrated termination network

The measurements of V_{CAN_L} and V_{CAN_H} while the ECU is sending a dominant bit may be carried out as shown in figure 21.

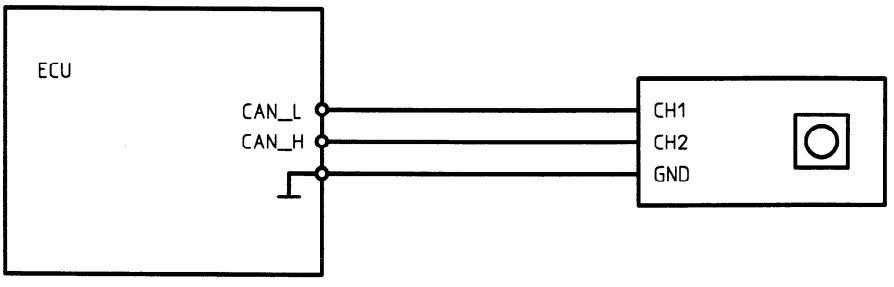


Figure 21 — Measurements of V_{CAN_L} and V_{CAN_H} while the ECU is sending a dominant bit

10.5.4.3.2 ECU without an integrated termination network

The measurements of V_{CAN_L} and V_{CAN_H} of an ECU without an integrated termination network are carried out as shown in figure 22.

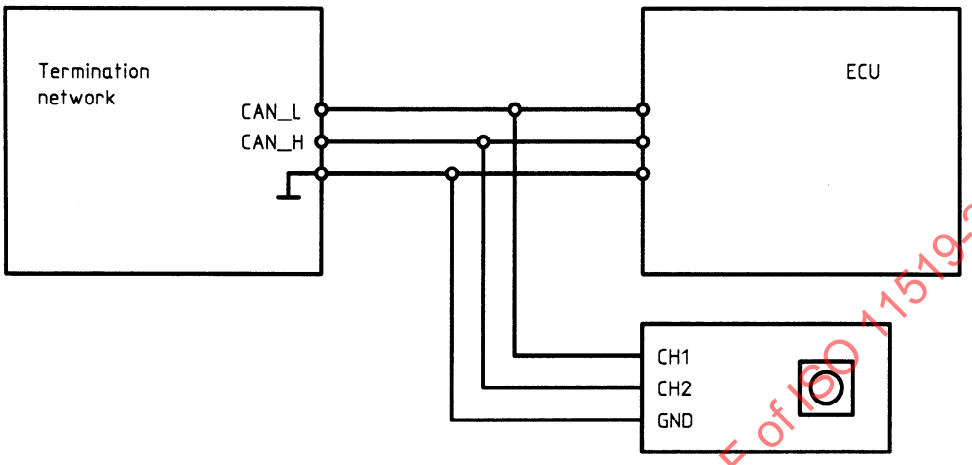


Figure 22 — Measurements of V_{CAN_L} and V_{CAN_H} while the ECU is sending a dominant bit

10.5.4.4 Dominant and recessive input threshold of an ECU

The test method specified is based on the fact that only a conforming ECU is able to send the acknowledgement bit when receiving a frame from the transmitter. The test transmitter shown in figures 23 and 24 sends a frame periodically. By controlling the ACK bit, it is possible to distinguish between conforming and nonconforming ECUs. Test voltages are shown in table 14.

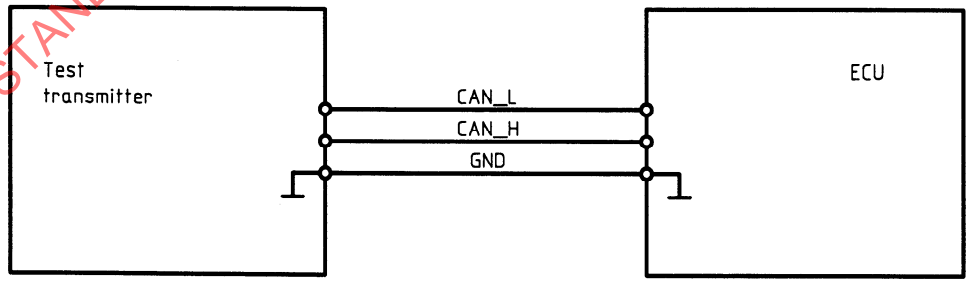


Figure 23 — Testing of the input threshold during transmission

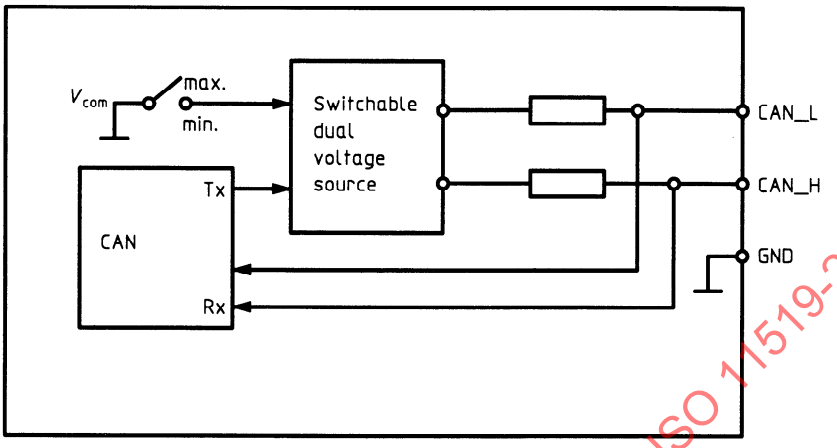


Figure 24 — Structure of the test transmitter

Table 14 — Test voltages

Parameter	V_{com}			
	U_L V		U_H V	
	max.	min.	max.	min.
Tx_rec	6,15	– 0,85	5,85	– 1,15
Tx_dom	5,85	– 1,15	6,15	– 0,85

10.5.4.5 Input capacitances

In order to determine the input capacitances C_{in} and C_{diff}, it is necessary to perform two measurements:

Measurement of C_{busin} (see figure 25 and table 15),

Measurement of C_{in} (see figure 26).

During the measurements a dominant bit shall not be transmitted. The value of the ECU input capacitance is in the range of a few pF. Therefore, the measurement equipment itself shall have a negligible capacitance or its capacitance shall be compensated by an exact measurement.

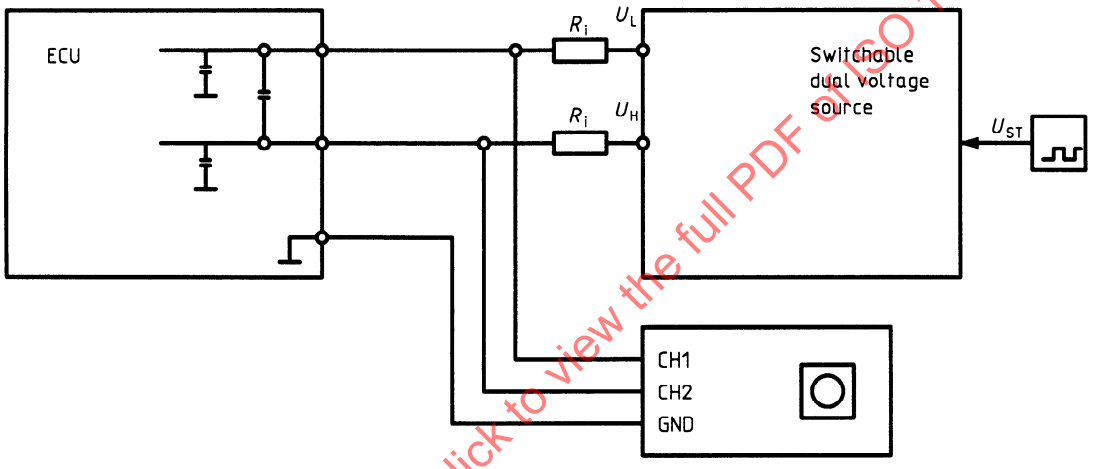


Figure 25 — Measurement of the input capacitance C_{busin} during the recessive state

Table 15 — Test voltages

Parameter	Test voltage at U_{ST} V	
	recessive	dominant
U_L	5	0
U_H	0	5