INTERNATIONAL STANDARD

ISO 11898

First edition
1993-11-15

Road vehicles — Interchange of digital information — Controller area network (CAN) for high-speed communication

Véhicules routiers — Échange d'information numérique — Gestionnaire de réseau de communication à vitesse élevée (CAN)

Reference number
ISO 11898:1993(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 11898 was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Sub-Committee SC 3, *Electrical and electronic equipment*.

Road vehicles — Interchange of digital information — Controller area network (CAN) for high-speed communication

1 Scope

This International Standard specifies characteristics of setting up an interchange of digital information between Electronic Control Units (ECUs) of road vehicles equipped with the Controller Area Network at transmission rates above 125 kbit/s up to 1 Mbit/s.

The Controller Area Network (CAN) is a serial communication protocol which supports distributed real-time control and multiplexing.

This specification of CAN describes the general architecture of CAN in terms of hierarchical layers according to the ISO reference model for Open Systems Interconnection (OSI) specified in ISO 7498. The data link layer and physical layer are specified according to ISO 8802-2 and ISO 8802-3. This International Standard contains detailed specifications of aspects of CAN belonging to the

a) physical layer;

b) data link layer

— Logical Link Control (LLC) sublayer,

— Medium Access Control (MAC) sublayer.

All other layers of the OSI model do not have counterparts within this specification of CAN protocol but are part of the user's level.

2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this part of ISO 11898. 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 11898 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 7637-3:—[1], *Road vehicles — Electrical disturbance by conduction and coupling — Part 3: Passenger cars and light commercial vehicles with nominal 12 V supply voltage and commercial vehicles with 24 V supply voltage — Electrical transient transmission by capacitive and inductive coupling via lines other than supply lines.*

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

1) To be published.

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.*

3 Definitions and abbreviations

For the purposes of this International Standard, the following definitions apply.

3.1 Data link layer definitions

3.1.1 bit rate: Number of bits per time during transmission, independent of bit representation.

3.1.2 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 stuff bit — into the outgoing bit stream. Receivers destuff the frame, i.e. the inverse procedure is carried out.

3.1.3 bus: Topology of a communication network, where all nodes are reached by passive links which allow transmission in both directions.

3.1.4 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".

3.1.5 contention-based arbitration: Carrier Sense Multiple Access (CSMA) arbitration procedure where simultaneous access of multiple nodes results in a contention. One frame will survive the contention uncorrupted.

3.1.6 frame: Data link protocol data unit specifying the arrangement and meaning of bits or bit fields in the sequence of transfer across the transmission medium.

3.1.7 multicast: Addressing where a single frame is addressed to a group of nodes simultaneously. Broadcast is a special case of multicast, whereby a single frame is addressed to all nodes simultaneously.

3.1.8 multi-master: System partitioned into several nodes where every node may temporarily control the action of other nodes.

3.1.9 node: Any assembly, linked to a communication line, capable of communicating across the network according to a communication protocol specification.

3.1.10 non-return-to-zero: Method of representing binary signals. Within one and the same bit time, the signal level does not change, i.e. a stream of bits having the same logical value provides no edges.

3.1.11 priority: Attribute to a frame controlling its ranking during arbitration. A high priority increases the probability that a frame wins the arbitration process.

3.1.12 protocol: Formal set of conventions or rules for the exchange of information between nodes, including the specification of frame administration, frame transfer and physical layer.

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

3.1.14 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 up to the maximum number of ECUs are connected to the bus line.

3.2.2 differential internal capacitance, C_{diff} (of an ECU): Capacitance 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} (of an ECU): Resistance 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}: value

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

with the voltages V_{CAN_L} and V_{CAN_H} denoting the voltages of CAN_L and CAN_H relative to ground of each individual ECU.

3.2.5 internal capacitance, C_{in} (of an ECU): Capacitance 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} (of an ECU): Sum of all asynchronous delay times occurring on the transmitting and receiving path 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} (of an ECU): Resistance 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 electrical loads on the bus line.

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

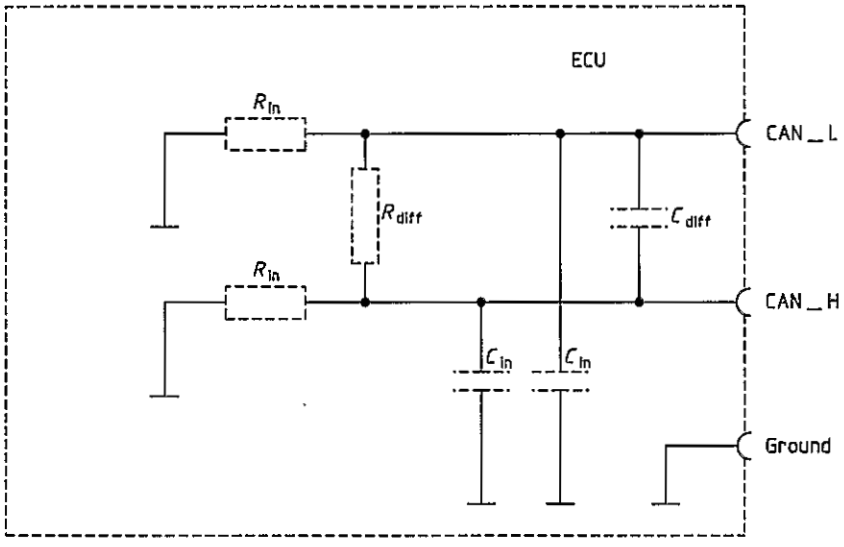

Figure 1 — Definitions of internal capacitances and internal resistances of ECU

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:

— multi-master 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.

4.1 Frames

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

4.2 Bus access method

When the bus is idle, 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 may 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 or if the added node does not require any additional transmitted data.

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

For dectecting errors, the following measures are provided:

— 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 normallly operating (error-active) receiving node. Such frames are aborted and will be retransmitted according to the implemented recovery procedure (see 6.3.3). The recovery time from detecting an error until the possible start of the next frame is typically 17 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. This means that 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 (6) dominant consecutive bits and violates the rule of bit stuffing and 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 (6) recessive consecutive bits. After transmission, an "error-passive" node will wait some additional time before initiating a further transmission (see suspend transmission in 8.4.5 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 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 OSI model

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

— data link layer,

— physical layer.

This International Standard specifies the data link and the physical layer of CAN (see figure 2).

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

— Logic 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).

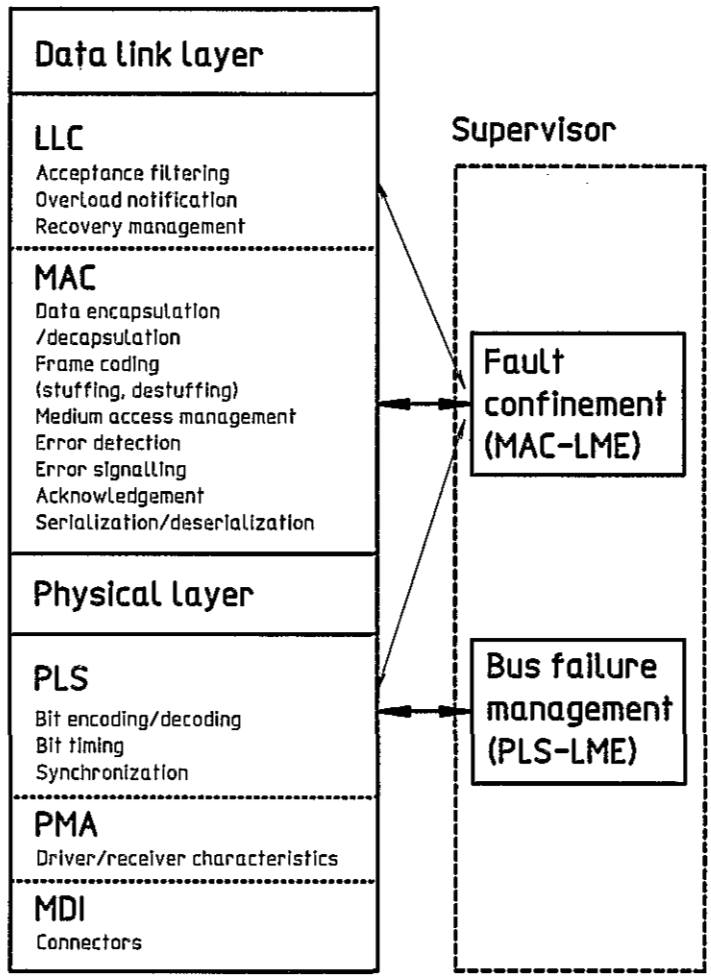

Figure 2 — Layered architecture of CAN

5.2 Protocol specification

Two peer protocol entities communicate with each other by exchanging frame or Protocol Data Units (PDUs). An (N)-layer Protocol Data Unit (NPDU) consists of N-layer specific Protocol Control Information (N-PCI) and (N)-user data. In order to transfer a NPDU it must be passed to a (N-1)-layer entity via a (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 PDU, i.e. a NPDU is directly constructed from the associated NSDU and the layer specific control information N-PCI. Figure 3 illustrates the data link sublayer interactions.

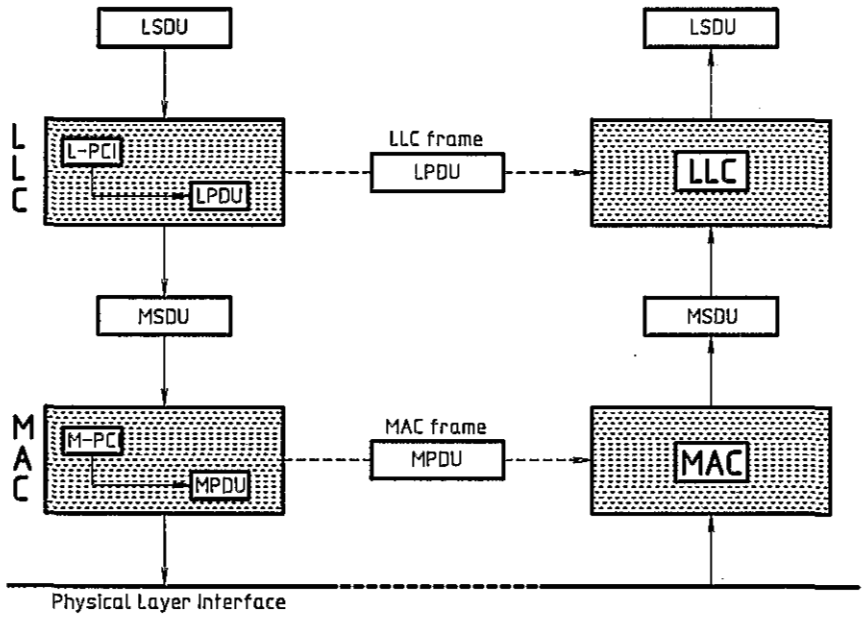

Figure 3 — Protocol layer interactions

5.3 Format description of services

5.3.1 Format description of service primitives

Service primitives are written in the form:

```
service.type(
             [parameter1,...]
            )
```

"service" indicates the name of the service, e.g. L_DATA for data transfer service provided by the LLC sublayer.

"type" indicates the type of the service primitives (see 5.3.2).

"[parameter1,...]" 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:

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.

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).

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 6.1.2. The messages that can be sent between LLC user and LLC sublayer are shown in table 1 a) and b).

Table 1 — Messages between LLC user and LLC sublayer

a) Message sent from LLC user to LLC sublayer	
User to LLC message	**Meaning**
Reset_Request	Request to set the node into an initial state
b) Messages 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 state "bus off".

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

6 Description of 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 Services of LLC sublayer

6.1.1 LLC sublayer

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

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.

Unacknowledged remote data request service

This service provides means by which a LLC user 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:

a) 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.

b) 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 the user.

6.1.2 Service primitive specification

This subclause describes in detail the LLC service primitives and their associated parameters. The complete list of LLC service primitives is given in table 2.

Table 2 — 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	Confirm data transfer
Unacknowledged Remote Data Request Service	
L_Remote.request	Request for remote data request
L_Remote.indication	Indication of remote data request
L_Remote.confirm	Confirmation remote data request

The parameters that are associated with the different LLC service primitives are listed in table 3.

Table 3 — 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

6.1.2.1 L_DATA.request

a) **Function**

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

b) **Semantics of 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 a LLC data frame by use of the data transfer service provided by the MAC sublayer (see table 5).

6.1.2.2 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 a LSDU.

b) **Semantics of L_DATA.indication primitive**

The primitive shall provide parameters as follows:

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

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

c) **Effect on receipt**

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

6.1.2.3 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 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 on receipt of this primitive by the LLC user is unspecified.

6.1.2.4 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 a LSDU.

b) **Semantics of 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 table 5).

6.1.2.5 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 transmission of a LSDU.

b) **Semantics of 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 on receipt of this primitive by the LLC user is unspecified.

6.1.2.6 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 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 on receipt of this primitive by the LLC user is unspecified.

6.2 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.

6.2.1 Specification of LLC data frame

A 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

Identifier

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

DLC field

The number of bytes in the data field is indicated by the Data Length Code. 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. Values other than those specified in table 4 may not be used.

Table 4 — 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

Data field

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

6.2.2 Specification of LLC remote frame

A 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 6.2.1). 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.

6.3 Functions of LLC sublayer

The LLC sublayer provides the following functions:

a) frame acceptance filtering,

b) overload notification,

c) recovery management.

6.3.1 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 frame acceptance filtering whether the frame is relevant for it or not.

6.3.2 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.

6.3.3 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

— (MAC-) acknowledged transfer of LLC frames,

— transmission of overload frames.

The interface service data from or to the LLC sublayer is described in 6.1.

8 Description of MAC sublayer

8.1 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 related to

— encapsulation/decapsulation of the transmit/receive data,

— error detection and signalling,

— management of the transmit/receive medium access.

8.2 Services of 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:

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.

Acknowledged remote data request

This service provides means by which a 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's user.

Overload frame transfer

This service provides means by which a 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.

The service primitives the MAC sublayer provides to the LLC sublayer are given in table 5.

Table 5 — 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

8.2.1 MA_DATA.request

a) **Function**

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

b) **Semantics of 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.

8.2.2 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 a MSDU.

b) **Semantics of 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 a MSDU is indicated to the LLC sublayer only if it has been received correctly.

c) **Effect on receipt**

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

8.2.3 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 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 the arbitration.

c) **Effect on receipt**

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

8.2.4 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 a MSDU.

b) **Semantics of 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.

8.2.5 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 a MSDU.

b) **Semantics of MA_REMOTE.indication primitive**

The primitive shall provide parameters as follows:

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

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

c) **Effect of receipt**

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

8.2.6 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 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 the arbitration.

c) **Effect on receipt**

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

8.2.7 MA_OVLD.request

a) **Function**

The MA_OVLD.request primitive is passed from the LLC sublayer to the MAC sublayer to request transmission of a MAC overload frame (see 8.4.4). The overload frame is a fixed format frame and is completely constructed in the MAC sublayer.

b) **Semantics of 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.

8.2.8 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 8.4.4).

b) **Semantics of MA_OVLD.indication primitive**

The primitive does not provide any parameters:

```
MA_OVLD.indication(
                  )
```

c) **Effect on receipt**

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

8.2.9 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 the overload frame correctly.

b) **Semantics of 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 on receipt of this primitive by the LLC sublayer is unspecified.

8.3 Functional model of 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 part. The functions of both transmit and receive parts are described below.

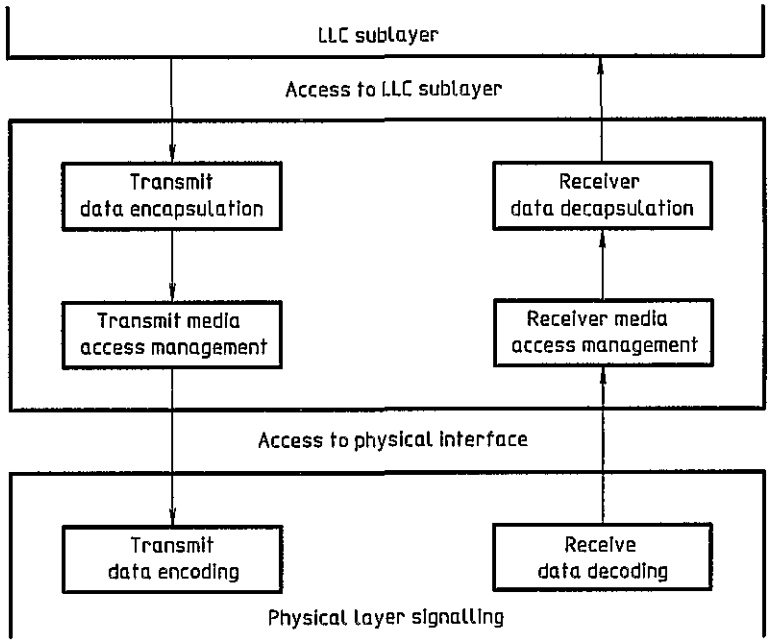

Figure 6 — Media access control functions

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, reserved 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 receive 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.

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) Removing the MAC specific information from the received frame.

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

8.4 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.

8.4.1 Specification of MAC data frame

A MAC data frame is composed of seven different bit fields:

— 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

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 8.4.5). All nodes have to synchronize to the leading edge caused by Start of Frame of the node starting transmission first.

Arbitration field

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 is "0" in a MAC data frame.

Control field

consists of six bits. It includes two bits reserved for future expansion followed by Data Length Code (see 6.2.1). Receivers accept "0" and "1" bits as reserved bits in all combinations. Until the function of the reserved bits are defined, the transmitter will only send "0" bits.

Data field

is equivalent to the LLC data field (see 6.2.1).

CRC field

contains the CRC sequence followed by a CRC delimiter.

CRC sequence

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 polynominal 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 polynominal 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) may 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.

CRC delimiter

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

ACK field

The ACK field is two bits long and contains the ACK slot and the ACK delimiter. In the ACK field, transmitting node sends two "recessive" bits.

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.

ACK delimiter

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

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

8.4.2 Specification of 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:

— Start of Frame (SOF);

— arbitration field;

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

— CRC field;

— ACK field;

— End of Frame (EOF).

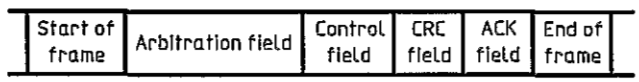

Figure 8 — MAC remote frame

Arbitration field

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 8.4.1).

8.4.3 Specification of error frame

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.

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 detect an error condition too and on their part start transmission of an error flag. So the sequence of "dominant" bits, which may actually 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 then they 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 over any activity on the bus line. Therefore, "error-passive" receivers always have to wait for six (6) subsequent equal bits after detecting an error condition, until they have completed their error flag.

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. Afterwards, it starts transmitting seven more "recessive" bits.

8.4.4 Specification of 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 the LLC sublayer to indicate an internal overload situation (see 8.11).

b) Reactive overload frame

   The transmission of the reactive overload frame will be initated by the MAC sublayer upon certain error conditions (see 8.11).

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.

Overload flag

consists of six "dominant" bits.

The overload flag's form destroys the fixed form of the intermission field (see 8.4.5). As a consequence, all other nodes also detect an overload condition and on their part start transmission of an overload flag.

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 of 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.

8.4.5 Specification of interframe space

Data frames and remote frames are separated from preceding frames whatever type they are (data frame, remote frame, error frame, overload frame) by a bit field called 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.

Interframe space

contains the bit fields Intermission and Bus Idle, and Suspend Transmission for "error-passive" nodes which have been transmitter of the previous frames (see figures 9 and 10).

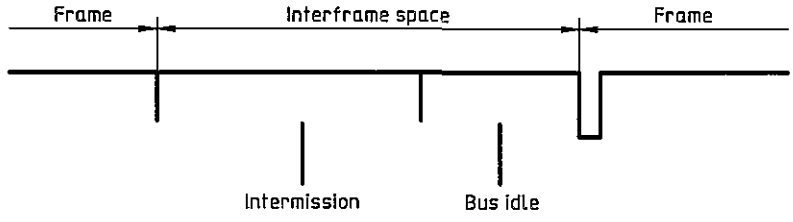

Figure 9 — Interframe space for nodes which are not "error-passive" or have been receiver of previous frame

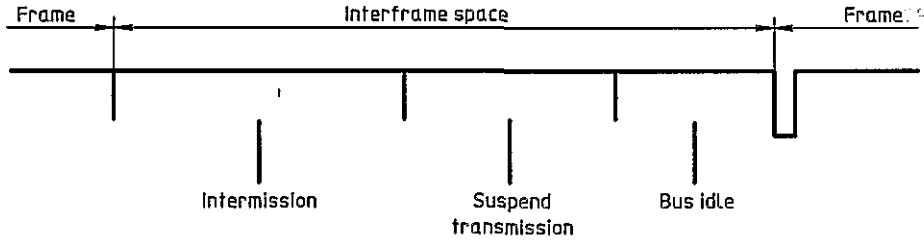

Figure 10 — Interframe space for "error-passive" nodes which have been transmitter of previous frame

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.

Bus idle

The period of bus idle may be of arbitrary length. The bus is recognized to be idle and any node can access the bus in order to transmit. A frame which is pending for transmission during the transmission 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.

Suspend transmission

After an "error-passive" node having 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 a receiver of this frame.

8.5 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 actual transmitted bit stream (see figure 11).

Destuffed bit stream	100000abc	011111abc
stuffed bit stream	1000001abc	0111110abc
	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.

8.6 Order of 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).

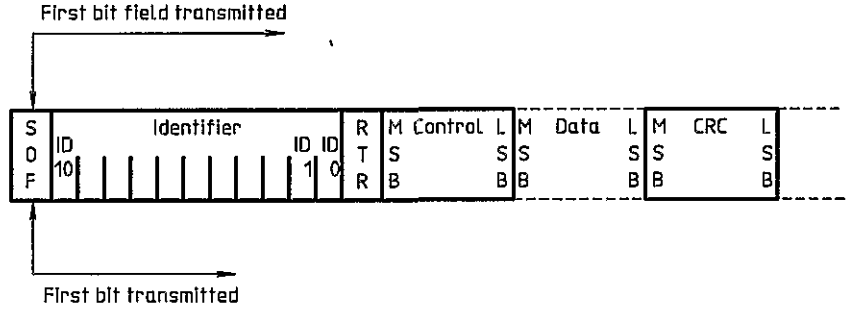

Figure 12 — Order of bit transmission

8.7 Frame validation

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

Transmitter

The frame is valid for a transmitter, if there is no error until the end of End of Frame. If a frame is corrupted, recovery is processed as described in 6.3.3.

Receiver

The frame is valid for receivers, if there is no error until the last but one bit of End of Frame.

8.8 Medium access method

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

8.8.1 Multimaster

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

8.8.2 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.

8.8.3 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 transmitter of the current frame or has been 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 in coincidence, that node transmitting the frame with the highest priority at this time will become the only bus master. The mechanism to resolve the resulting bus access conflict is contention-based arbitration.

8.8.4 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 8.8.3. A MAC error frame is transmitted as specified in 8.10. A MAC overload frame is transmitted as specified in 8.11.

8.8.5 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.

8.8.6 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 the identifier of which 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 higher priority than the remote frame. This is achieved by assigning according values to the RTR bits.

8.8.7 Collision resolution

Apart from the principle that transmission may be initiated only when the bus is free, there exist further principles for the resolution of collision:

— Within one system each information must be assigned by 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 remote frames with identical identifiers and different data length codes leads to unresolvable collisions.

8.9 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):

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 sending a passive error flag and detecting a "dominant" bit does not interpret this as a bit error.

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.

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 does not equal the received one.

Form error

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

Exception: a receiver monitoring a "dominant" bit at the last bit of End of Frame does not interpret this as a form error.

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.

8.10 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.

8.11 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 End of Frame by a receiver.

A 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 a) 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 shall comply with all specifications and values given in this International Standard.

10 Description of physical layer

The physical layer is an electrical circuit realization that connects an ECU to a bus. The total number of ECUs will be limited by the electric loads on the bus line. The physical layer specified is for high speed applications (up to 1 Mbit/s).

10.1 Functional model of 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.

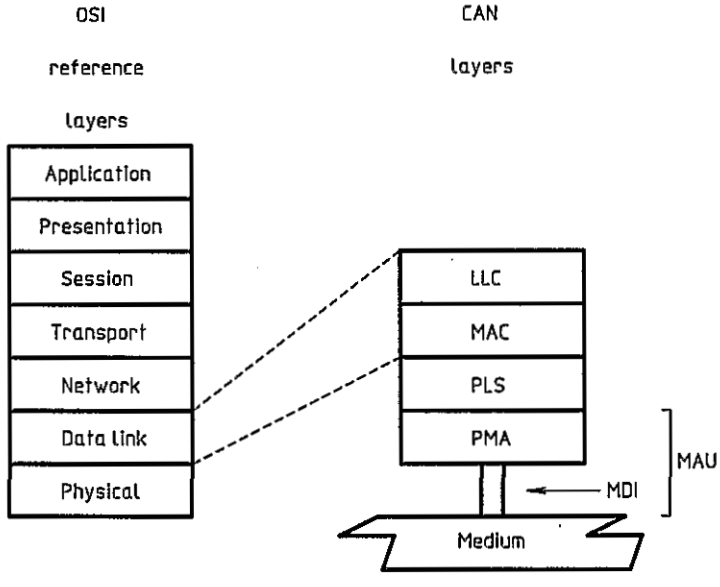

Figure 13 — Model of physical layer architecture

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 Physical Medium Attachment (PMA) and Medium Dependent Interface (MDI).

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

c) Medium Dependent Interface (MDI) encompasses the mechanical and electrical interface between the physical medium and the MAU.

10.2 Services of 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 transmission of a "dominant" or "recessive" bit. The primitive provides the following parameter:

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

The OUTPUT_UNIT parameter may take on 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 take on 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 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).

Nominal bit rate

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

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).

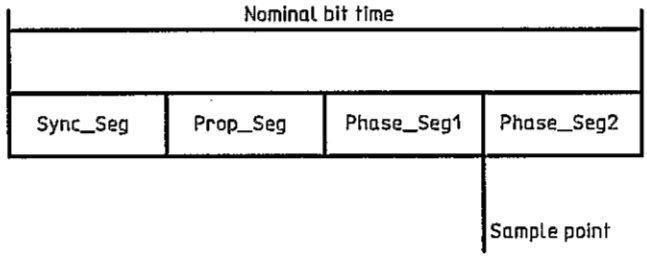

Figure 14 — Partition of bit time

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.

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 (see definition 3.2.6).

PHASE_SEG1, PHASE_SEG2

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

Sample point

The sample point is the point of 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.

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 bit time

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 at least from 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.

Length of time segments

— Sync_Seg is one time quantum long.

— Prop_Seg is programmable to be 1, 2, 3, ..., 8 or more time quanta long.

— Phase_Seg1 is programmable to be 1, 2, 3, ..., 8 or more 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 long.

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

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

a) Only one synchronization within one bit time is allowed.

b) An edge will be used for synchronization only if the value detected at the previous sample point (previous read bus value) differs from the bus value immediately after the edge.

c) Hard synchronization is performed during bus idle whenever there is a "recessive" to "dominant" edge.

d) All other "recessive" to "dominant" edges (and optionally "dominant" to "recessive" edges in case of low bit rates) fulfilling rules a) and b) 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.

Resynchronization jump width

As a result of resynchronization, Phase_Seg1 may be lengthened or Phase_Seg2 may be shortened. The amount of lengthening and shortening the phase buffer segments has an upper limit given by the resynchronization jump width. The resynchronization jump width shall be programmable between 1 and minimum(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 re-synchronizing 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.

Phase error of synchronization edge

The phase error, e, 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

Resynchronization leads to a shortening or lengthening of the bit time such that the position of the sample point is correct, 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 resynchronization 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

10.4.1.1 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.

10.4.1.2 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).

10.4.1.3 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

10.4.2.1 Input 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 High-Speed Medium Access Unit (HS-MAU)

The following description is valid for a two-wire differential bus.

10.5.1 Physical medium attachment sublayer specification

10.5.1.1 Functional description

As shown in figure 15, the bus line is terminated at each end with a load resistor denoted by R_L. These resistors suppress reflections. The locating of R_L within an ECU should be avoided because the bus lines lose termination if one of these ECUs is disconnected from the bus line.

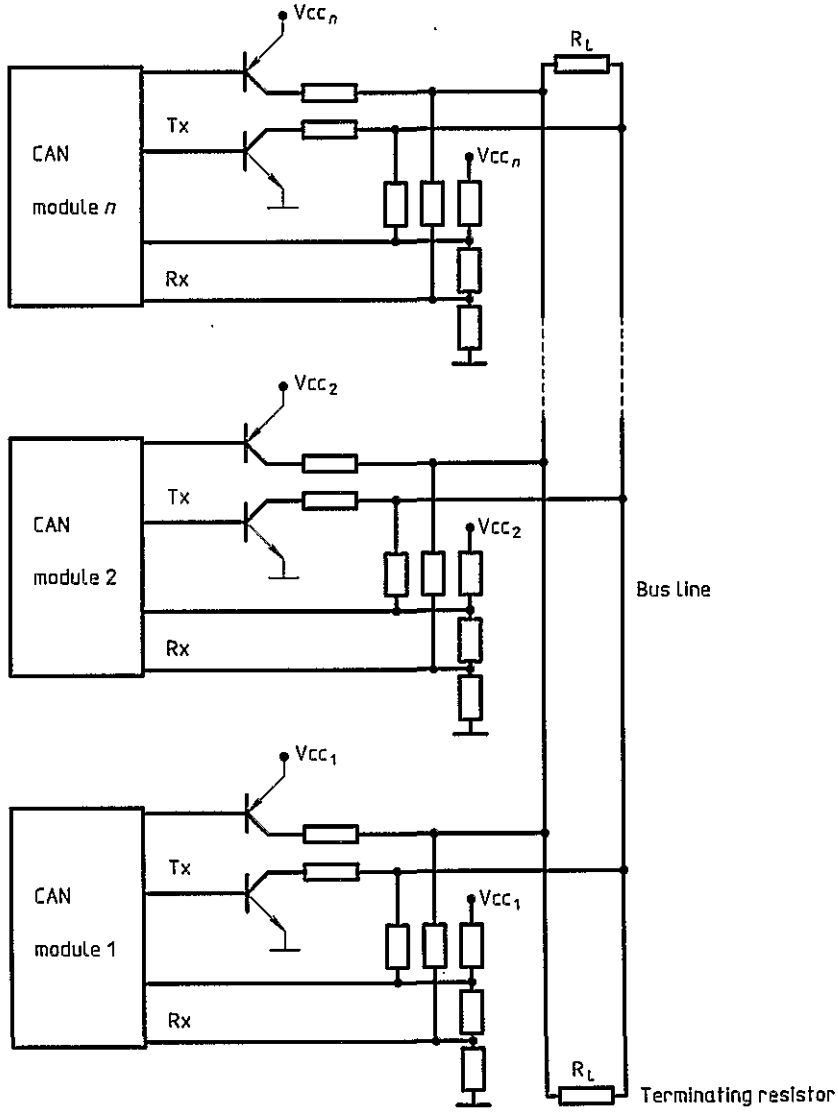

Figure 15 — Suggested electrical interconnection

The bus is in the "recessive" state if the transistor pairs of all ECUs are switched off. In this case the mean bus voltage is generated by each ECU voltage source which has a high internal resistance. Figure 15 shows the resistor network that defines the reference for the receive operation.

A "dominant" bit is sent to the bus line if the transistor pairs of at least one unit are switched on. This induces a current flow through the terminating resistors, and consequently a differential voltage between the two wires of the bus line.

The "dominant" and "recessive" states are detected by a resistor network that transforms the differential voltages of the bus line to the corresponding "recessive" and "dominant" voltage levels at the comparator input of the receiving circuitry.

10.5.1.2 Electrical specification

All data given in tables 6 to 11 are independent of a specific physical layer implementation. The parameters specified in these tables shall be fulfilled throughout the operating temperature range of every ECU. The parameters are chosen such that a maximum number of 30 ECUs may be connected to the common bus line.

10.5.1.2.1 Bus levels

The bus line can have one of the two logical states: "recessive" or "dominant" (see figure 16).

In the "recessive" state, V_{CAN_H} and V_{CAN_L} are fixed to mean voltage level. V_{diff} is approximately zero. The "recessive" state is transmitted during bus idle or a "recessive" bit.

The "dominant" state is represented by a differential voltage greater than a minimum threshold. The "dominant" state overwrites the "recessive" state, and is transmitted during a "dominant" bit.

Bus levels during arbitration

During arbitration, various ECUs may simultaneously transmit a "dominant" bit. In this case V_{diff} exceeds the V_{diff} seen during a single operation. Single operations means that the bus line is driven by one ECU only.

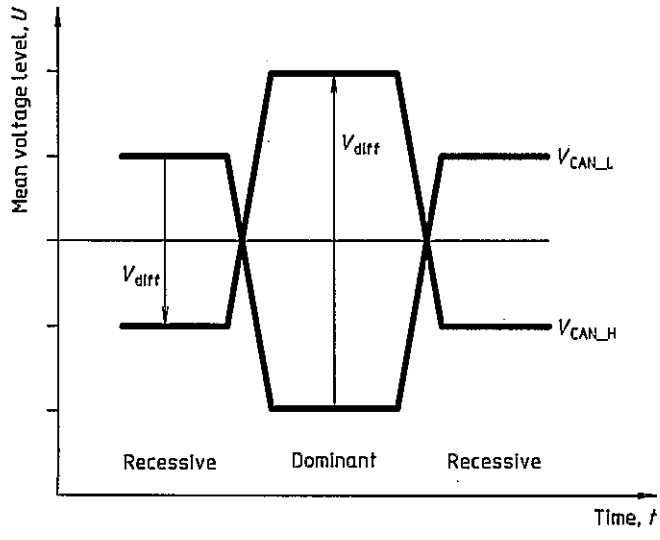

Figure 16 — Physical bit representation

10.5.1.2.2 Electronic control unit (ECU)

The parameters given in table 6 shall be tested at the CAN_L and CAN_H pins of each ECU, with the ECU disconnected from the bus line (see 10.5.4).

Table 6 — Maximum ratings of V_{CAN_L} and V_{CAN_H} of ECU

Nominal battery voltage V	Notation	Voltage min. V	Voltage max. V
12	V_{CAN_H}	−3,0	16,0
	V_{CAN_L}	−3,0	16,0
24	V_{CAN_H}	−3,0	32,0
	V_{CAN_L}	−3,0	32,0

Notes to the ratings:

— undisturbed operation does not have to be guaranteed;

— no destruction of transmitter circuit; no time limit.

Table 7 — DC parameters for recessive state of ECU disconnected from bus line

Parameter	Notation	Unit	Value min.	Value nom.	Value max.	Condition
Output bus voltage	V_{CAN_H}	V	2,0	2,5	3,0	no load
	V_{CAN_L}	V	2,0	2,5	3,0	
Differential output bus voltage	V_{diff}	mV	−500	0	50	no load
Differential internal resistance	R_{diff}	kΩ	10		100	no load [1]
Internal resistor[2]	R_{in}	kΩ	5		50	
Differential input voltage[3]	V_{diff}	V	−1,0		0,5	[4] [5]

1) The load is connected between CAN_H and CAN_L. For an ECU without integrated terminating resistor (normal use), this resistor is a 60Ω resistor. For ECUs with an integrated terminating resistor, this is a 120Ω resistor. In this case, R_L is seen between CAN_H and CAN_L instead of R_{diff}.

2) R_{in} of CAN_H and CAN_L should have almost the same value. The deviation shall be less than 3 % relative to each other.

3) The threshold for receiving the "dominant" and "recessive" bits ensures a noise immunity of 0,3 V and 0,5 V respectively. The lower value for the "dominant" state is motivated by the fact that a lower load resistance between CAN_H and CAN_L is seen (the capacitance of the supply voltage source is the reason that the internal resistance of the transmitter of the "dominant" bit is connected in parallel to the bus load resistance).

4) Threshold for receiving a "recessive" bit.

5) Reception shall be ensured within the common mode voltage range specified in tables 10 and 11 respectively.

Table 8 — DC parameters for dominant state of ECU disconnected from bus line

Parameter	Notation	Unit	Value			Condition
			min.	nom.	max.	
Output bus voltage	V_{CAN_H}	V	2,75	3,5	4,5	load 60 Ω [1]
	V_{CAN_L}	V	0,5	1,5	2,25	
Differential output voltage	V_{diff}	V	1,5	2,0	3,0	load 60 Ω [1]
Differential input voltage[2]	V_{diff}	V	0,9		5,0	load 60 Ω [1][3]

1) The load is connected between CAN_H and CAN_L. For an ECU without integrated terminating resistor (normal use), this resistor is a 60 Ω resistor. For ECUs with an integrated terminating resistor, this is a 120 Ω resistor. In this case, R_L is seen between CAN_H and CAN_L instead of R_{diff}.

2) *The threshold for receiving the "dominant" and "recessive" bits ensures a noise immunity of 0,3 V and 0,5 V respectively.* The lower value for the "dominant" state is motivated by the fact that a lower load resistance between CAN_H and CAN_L is seen (the capacitance of the supply voltage source is the reason that the internal resistance of the transmitter of the "dominant" bit is connected in parallel to the bus load resistance).

3) Threshold for receiving a "dominant" bit. Reception shall be ensured within the common mode voltage range specified in tables 10 and 11 respectively.

Table 9 — AC parameters of ECU disconnected from bus line

Parameter	Notation	Unit	Value			Conditions
			min.	nom.	max.	
Bit time	t_B	µs	1			[1]
Internal capacitance[2]	C_{in}	pF		20		[3]
Differential internal capacitance[2]	C_{diff}	pF		10		1 Mbit/s

1) The min. bit time corresponds to a max. bit rate of 1 Mbit/s. The lower end of the bit rate depends on the protocol IC.

2) In addition to the internal capacitance restriction, a bus connection should also have as low an inductance as possible. This is particularly important for high bit rates. The min. values of C_{in} and C_{diff} may be zero. The max. tolerable values are determined by the bit timing and the network topology parameters l and d (see table 15, note 3). Proper functionality is guaranteed if occurring cable-reflected waves do not suppress the dominant differential voltage levels below $V_{diff} = 0,9$ V and do not increase the recessive differential voltage level above $V_{diff} = 0,5$ V at each individual ECU (see tables 7 and 8).

3) 1 Mbit/s for CAN_H and CAN_L relative to HF ground.

Internal delay time

The internal delay time of an ECU, t_{ECU}, is defined as the sum of all asynchronous delays that occur along the transmission and reception path, relative to the bit timing logic unit of the protocol IC of individual ECUs. For more details see figure 17.

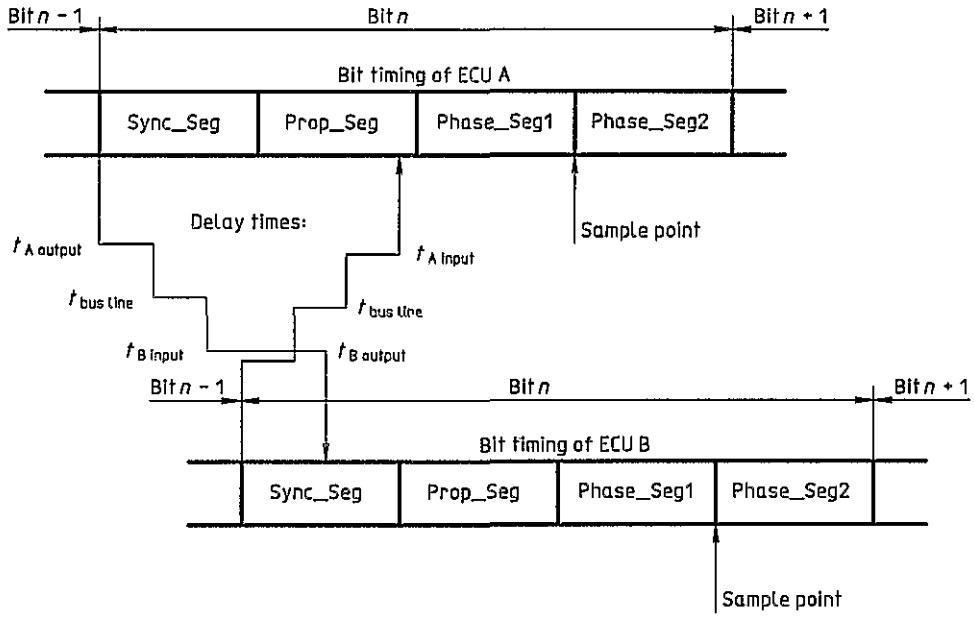

Figure 17 — Time relationship between ECU bit timing logic A and B during arbitration, bit timing of two ECUs, and delay times

In connection with figure 17,

— the sum of output and input ECU delays is critical relative to the bit timing logic. The important characteristic parameter of an ECU is:

$$t_{ECU} = t_{output} + t_{input}$$

— for proper arbitration, the following conditions shall be met:

$$t_{ECU\,A} + t_{ECU\,B} + 2t_{bus\ line} \leqslant t_{SYNC\_SEG} + t_{PROP\_SEG}$$

i.e. the leading transmitting bit timing logic with respect to synchronization of ECU A must be able to know the correct bus level of bit n at the sampling point. The tolerable values of t_{ECU} strongly depend on the required bit rate and line length of the bus and of the possible bit timing as shown by the arbitration condition;

— the acceptable quartz tolerances of the protocol ICs and the potential for incorrect synchronization is determined by PHASE_SEG1 and 2.

10.5.1.2.3 Common mode voltages

The parameters specified in table 10 apply when all ECUs (between 2 and 30) are connected to a correctly terminated bus line.

Table 10 — Bus voltage parameters for recessive state

Parameter	Notation	Unit	Value			Condition
			min.	nom.	max.	
Common mode bus voltage	V_{CAN_H}	V		2,5	7,0	Measured with respect to the individual ground of each ECU.
	V_{CAN_L}	V	−2,0	2,5		
Differential bus voltage[1]	V_{diff}	mV	−120	0	12	Measured at each ECU connected to the bus line.

1) The differential bus voltage is determined by the output behaviour of all ECUs during the recessive state. Therefore V_{diff} is approximately zero (see table 7). The min. value is determined by the requirement that a single transmitter must be able to represent a dominant bit by a min. value of $V_{diff} = 1,2$ V.

Table 11 — Bus voltage parameters for dominant state

Parameter	Notation	Unit	Value			Condition
			min.	nom.	max.	
Common mode bus voltage[1]	V_{CAN_H}	V		3,5	7,0	Measured with respect to the individual ground of each ECU.
	V_{CAN_L}	V	−2,0	1,5		
Differential bus voltage[2]	V_{diff}	V	1,2	2,0	3,0	Measured at each ECU connected to the bus line.

1) The min. value of V_{CAN_H} is determined by the min. value of V_{CAN_L} plus the min. value of V_{diff}. The max. value of V_{CAN_L} is determined by the max. value of V_{CAN_H} minus the min. value of V_{diff}.

2) The bus load increases as ECUs are added to the network, by R_{diff}. Consequently, V_{diff} decreases. The min. value of V_{diff} determines the number of ECUs allowed on the bus. The max. value of V_{diff} is defined by the upper limit during arbitration. This max. value of V_{diff} must not exceed 3 V.

10.5.1.2.4 Illustration of voltage range

Load conditions are defined within the tables 7 to 11. Figures 18 to 21 illustrate the valid voltage ranges of V_{CAN_H} and V_{CAN_L}.

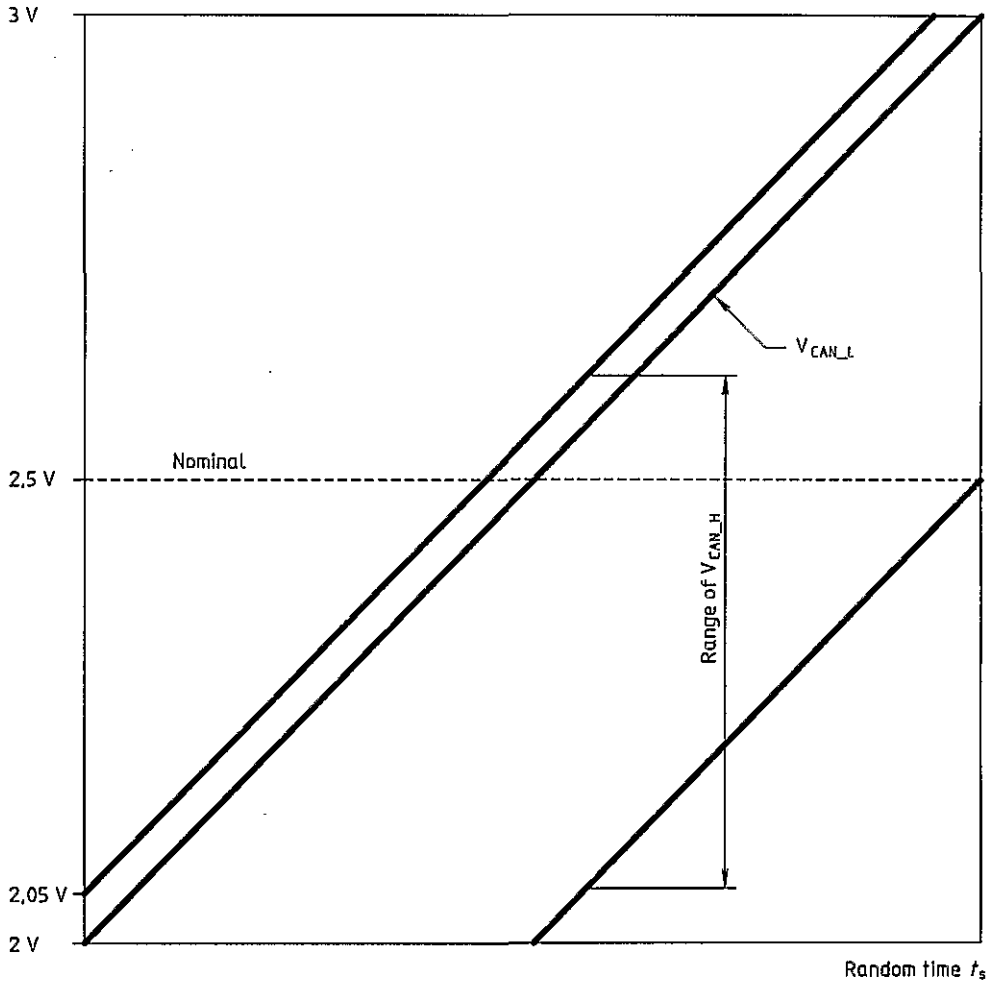

Figure 18 — Valid voltage range of V_{CAN_H} during recessive state of ECU disconnected from bus line, if V_{CAN_L} varies from min. to max. voltage level

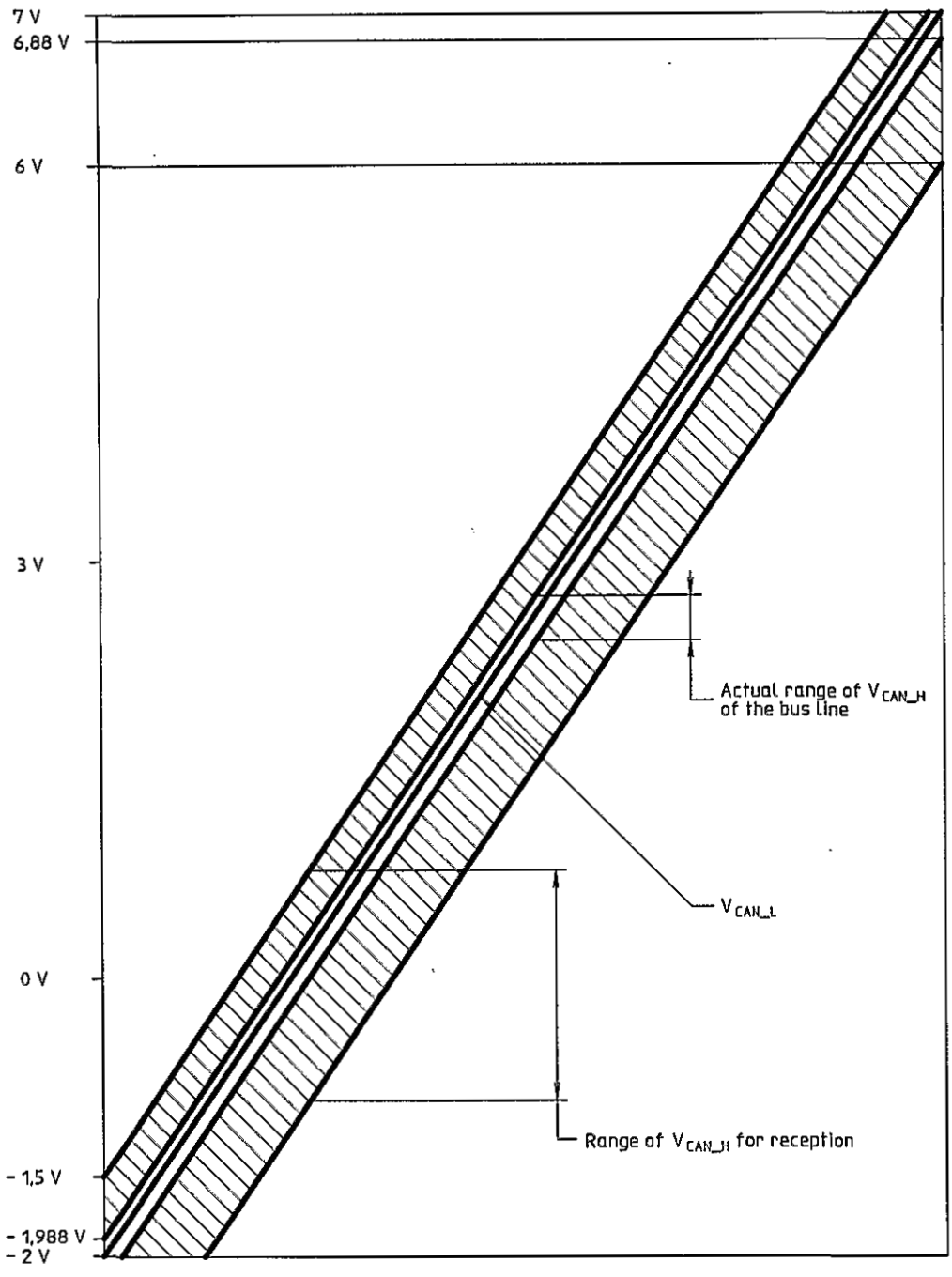

Figure 19 — Valid voltage range of V_{CAN_H} for monitoring recessive bus state, and for disconnected ECU, *if V_{CAN_L} varies from min. to max. common mode range of bus line*

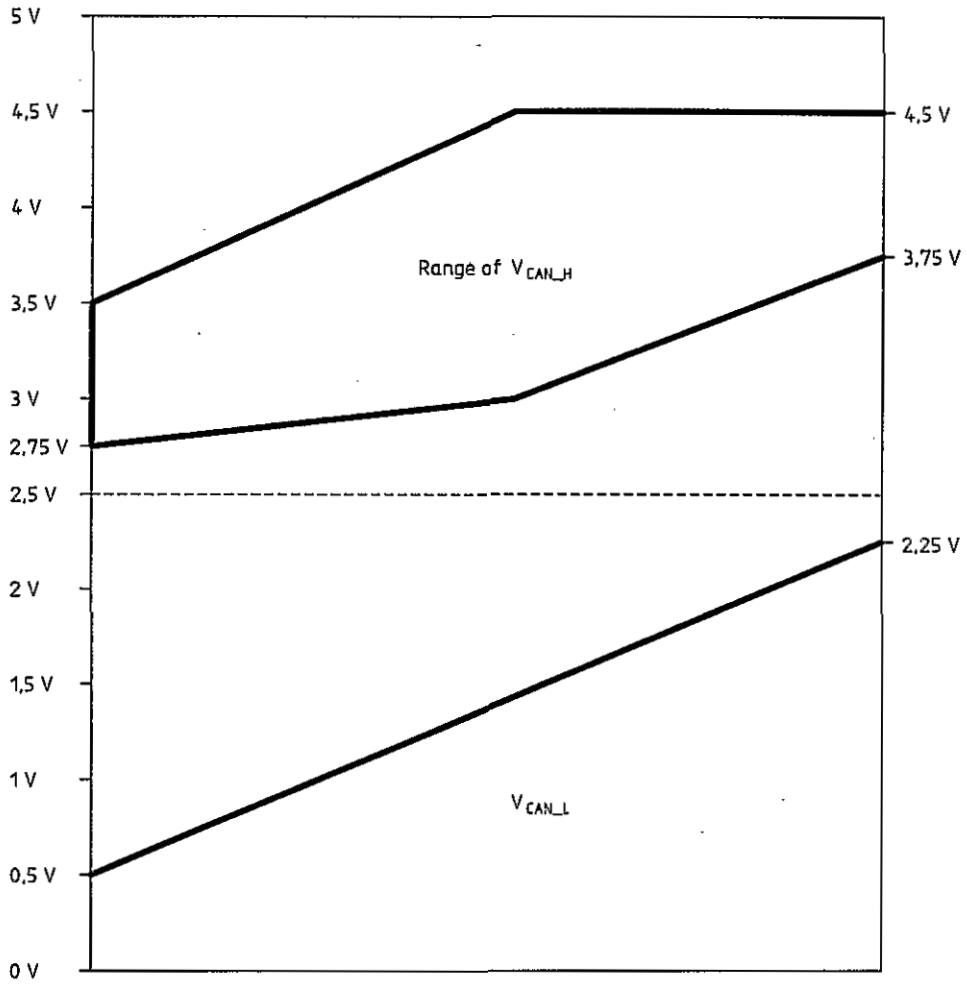

Figure 20 — Valid voltage range of V_{CAN_H} during dominant state of ECU which is disconnected from bus line, if V_{CAN_L} varies from min. to max. voltage level

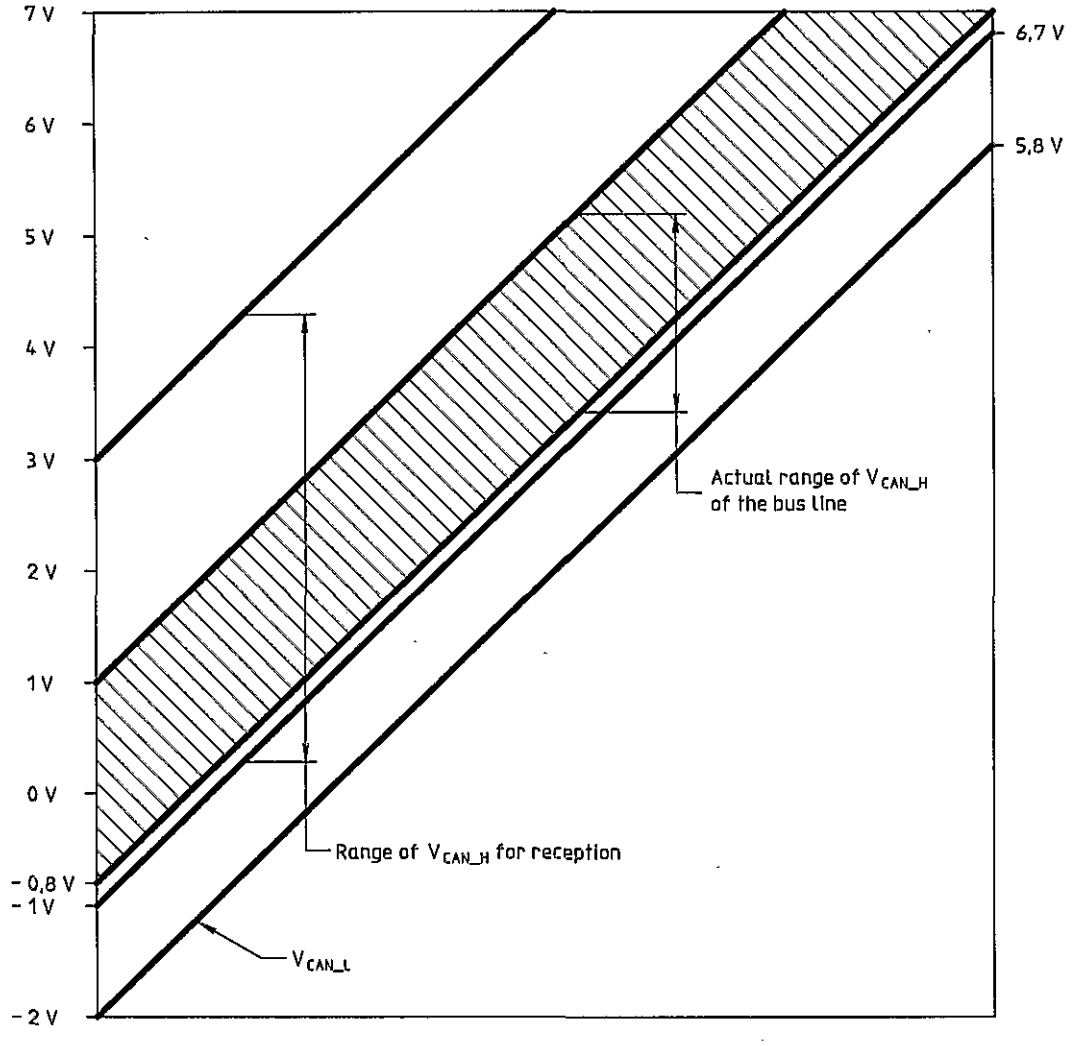

Figure 21 — Valid voltage range of V_{CAN_H} for monitoring dominant bus state, and for disconnected ECU, if V_{CAN_L} varies from min. to max. common mode range of bus line

10.5.1.2.5 Disturbance by coupling

The tolerated disturbances of CAN_H and CAN_L by coupling are defined in accordance with ISO 7637-3:1990, test pulses 3a and 3b.

10.5.2 Medium Dependent Interface (MDI) specification: Connector parameters

A connector used to plug ECUs to the bus line shall meet the requirements specified in table 12. 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.	nom.	max.
Voltage	V_{BAT} = 12 V	U	V			16
	V_{BAT} = 24 V	U	V			32
Current		I	mA	0	25	80
Peak current[1)]		Ip	mA			500
Impedance		Zc	Ω		120	
Transmission frequency		f	MHz	25		
Transmission resistance[2)]		R_T	mΩ		70	

1) Time restriction: $101t_B$.

2) The differential voltage of the bus line seen by the receiving ECU depends on the line resistance between this and the transmitting ECU. Therefore the transmission resistance of the signal wires is limited by the bus level parameters at each ECU.

10.5.3 Physical medium specification

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

Cables for the bus line shall be in accordance with the specifications of table 13.

Table 13 — Physical media parameters of a pair of wires (shielded or unshielded)

Parameter	Notation	Unit	Value			Conditions
			min.	nom.	max.	
Impedance	Z	Ω	108	120	132	Measured between two signal wires.
Length-related resistance	r	mΩ/m		70		1)
Specific line delay		ns/m		5		2)

1) The differential voltage on the bus line seen by a receiving ECU depends on the line resistance between it and the transmitting ECU. Therefore, the total resistance of the signal wires is limited by the bus level parameters of each ECU.

2) The min. delay between two points of the bus line may be zero. The max. value is determined by the bit time and the delay times of the transmitting and receiving circuitry.

10.5.3.1 Termination resistor

The termination resistor used shall comply with the limits specified in table 14.

Table 14 — Termination resistor

Notation	Unit	Value			Conditions
		min.	nom.	max.	
R_c	Ω	118	120	130	Min. power dissipation: 220 mW.

10.5.3.2 Topology

The wiring topology of a CAN network should be as close as possible to a single line structure in order to avoid cable-reflected waves. In practice short stubs, as shown in figure 22 and table 15, are necessary to connect ECUs to the bus line successfully.

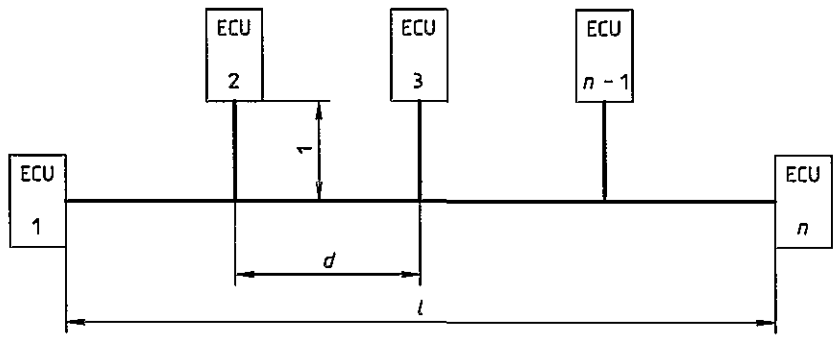

Figure 22 — Wiring network topology

Table 15 — Network topology parameters

Parameter	Notation	Unit	Value			Conditions
			min.	nom.	max.	
Bus length	L	m	0		40	
Cable stub length[1)]	l	m	0		0,3	Bit rate: 1Mbit/s [2)]
Node distance	d	m	0,1		40	

1) Dependent on the topology, the Baud rate, and the slew rate deviations from 120 Ω may be possible. It is, however, necessary to check the applicability of other resistor values in each case.

2) At bit rates lower than 1 Mbit/s the bus length may be lengthened significantly. Depending on *l*, the bit rate and internal capacitances of the individual ECUs, other network topologies with changed lengths *l* and *d* may be used. In this case the influence of occurring cable resonator waves on the bit representation on the bus line should be carefully checked by measurements of V_{diff} at each ECU (see also table 8, note 3).

10.5.4 Conformance tests

Figures 23 to 29 and the formulae shown indicate the principles how the electrical parameters specified in this subclause may be verified.

10.5.4.1 Recessive output of ECUs

Recessive output of the ECUs, V_{CAN_L} and V_{CAN_H}, (see figure 23) are measured unloaded while the bus is idle. V_{diff} is determined by:

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

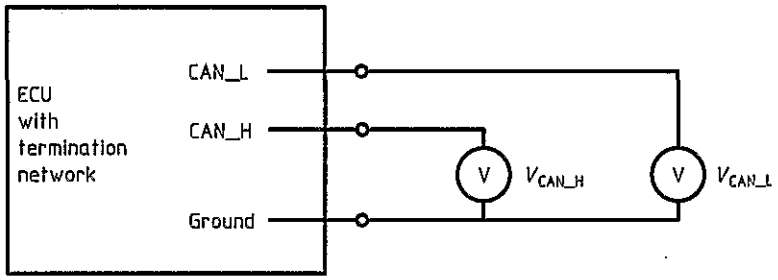

Figure 23 — Measurements of V_{CAN_L} and V_{CAN_H} during bus idle state

10.5.4.2 Internal resistance of CAN_L and CAN_H

The ground-related internal termination resistance of CAN_L and CAN_H (R_{in_L} and R_{in_H}) is measured as shown in figure 24, with the ECU protocol IC set to bus idle.

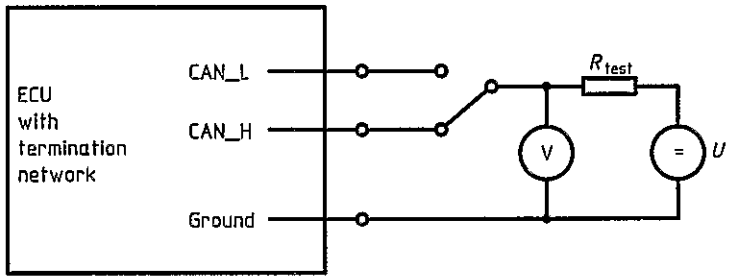

Figure 24 — Measurement of R_{in} while ECU protocol IC is set to bus idle

R_{in_L} and R_{in_H} are determined for $U = 0$ V and $U = 5$ V respectively, with $R_{test} = 5$ kΩ.

R_{in_L} and R_{in_H} are calculated by:

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

where V_{CAN_L} and V_{CAN_H} are the open circuit voltages according to figure 23.

When the termination resistor is integrated within the ECU, CAN_L and CAN_H shall be shorted. The measured value of R_{in_ECU} represents the combined resistance of RH and R_{in_L}. In this case, the following relation is valid:

$$\frac{R_{in\_H,\,min.} \cdot R_{in\_L,\,min.}}{R_{in\_H,\,min.} + R_{in\_L,\,min.}} < R_{in\_ECU} < \frac{R_{in\_H,\,max.} \cdot R_{in\_L,\,max.}}{R_{in\_H,\,max.} + R_{in\_L,\,max.}}$$

10.5.4.3 Internal differential resistor

The measurements of R_{diff} while the ECU protocol IC is set to idle shall be taken as shown in figure 25.

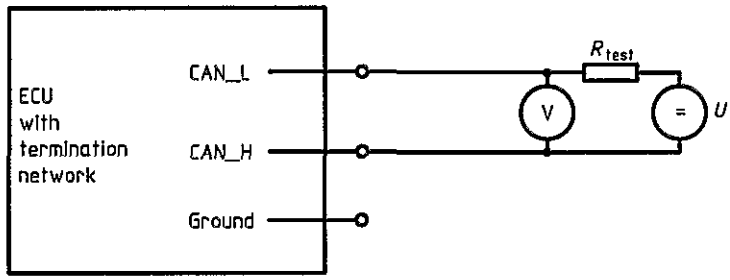

Figure 25 — Measurement of R_{diff} while ECU protocol IC is set to bus idle

R_{diff} is determined for $U = 5$ V and $R_{test} = 10$ kΩ during bus idle as:

$$R_{diff} = R_{test} \frac{V_{diff} - V}{V - U}$$

where V_{diff} is the differential open circuit voltage as in 10.5.4.1.

When the termination resistor is integrated within the ECU, then $R_{test} = 120$ Ω. In this case the measurement of R_{diff} is just the measurement of RL.

10.5.4.4 Recessive input threshold of ECU

The input threshold for recessive bit detection of an ECU shall be measured as shown in figure 26.

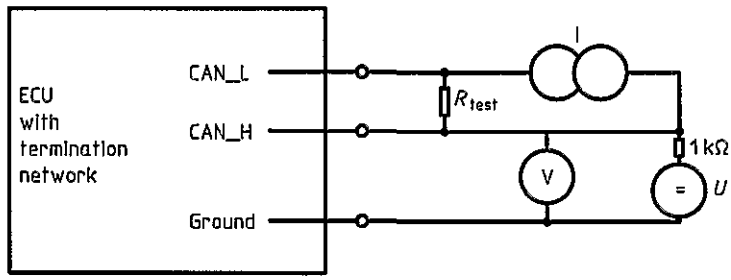

Figure 26 — Testing of input threshold for recessive bit detection

The current I is adjusted to a value which induces the upper threshold of 0,5 V, with $R_{test} = 60$ Ω, for detecting a recessive bit during the recessive state. Alternatively U is set to two values that produce $V = -2$ V and $V = 6{,}5$ V, during bus idle. Under these conditions the ECU shall not stop transmitting the frame. This indicates that every transmitted recessive bit is still detected as recessive by the protocol IC of the ECU tested. The level of dominant bits is nearly independent of U.

When the termination resistor is integrated within the ECU, R_{test} is set to 120 Ω. The conditions for I and U remain unchanged.

10.5.4.5 Dominant output of ECU

The dominant output voltages V_{CAN_H} and V_{CAN_L} shall be taken as shown in figure 27 while the ECU is transmitting a dominant bit.

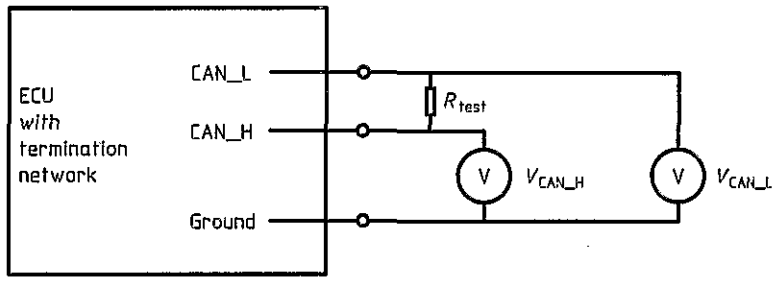

Figure 27 — Measurement of V_{CAN_H} and V_{CAN_L} while ECU transmits dominant bit

V_{CAN_H} and V_{CAN_L} are measured during dominant bit transmission. R_{test} is set to 60 Ω. The corresponding value of V_{diff} is given by:

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

When the termination resistor is integrated within the ECU, R_{test} is set to 120 Ω. The other specifications remain unchanged.

10.5.4.6 Dominant input threshold of ECU

The testing of the input threshold of an ECU to detect a dominant bit shall be undertaken as shown in figure 28.

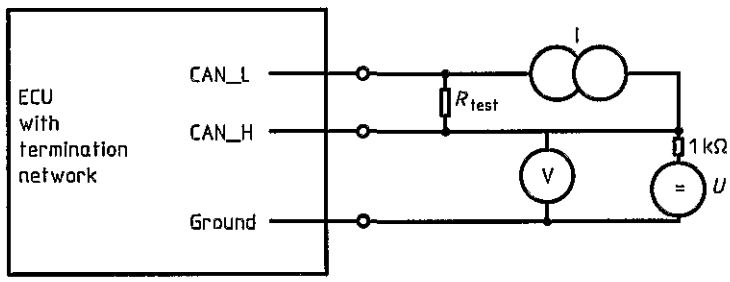

Figure 28 — Testing input threshold for dominant bit detection

The current I is adjusted to a value which induces the upper threshold of 0,9 V with R_{test} = 60 Ω, required to detect a dominant bit during recessive state. U is set to two values alternately that produce V = 2 V and V = 6,1 V during bus idle. Under these conditions the ECU shall stop transmitting the frame. This indicates that each recessive bit transmitted is detected as dominant by the protocol IC of the ECU. The level of dominant bits is nearly independent of U.

When the termination resistor is integrated within the ECU R_{test} is set to 120 Ω. The conditions for I and U remain unchanged.

10.5.4.7 Internal delay time

The measurement of the internal delay time t_{ECU} shall be undertaken as shown in figure 29, i.e. during partly overwriting of the first recessive identifier (see hatched area) by a dominant level until the arbitration is lost.

The test unit shown in figure 29 synchronizes itself to the start of frame bit transmitted by the protocol IC of the ECU. Upon detection of the first recessive identifier bit, the test unit partly overwrites this recessive bit for a duration of t_{overw} by a dominant level (see hatched area in figure 29). The duration of overwriting is increased until the protocol IC of the ECU loses arbitration and stops transmission. When this occurs, the part of the bit time available for delay time compensation t_{avail} is just exhausted (see also figure 14).

The internal delay time t_{ECU} is calculated as:

$$t_{ECU} = t_{avail} - t_{overw}$$

where t_{avail} is the bit time unit of the protocol IC as specified in 10.3.1.1, and t_{overw} is known from the test unit. The dominant and recessive voltage detection levels are set in the test unit to the corresponding threshold voltages for reception. I.e. the dominant overwriting level shall be 0,9 V, and the recessive level shall be 0,5 V. This ensures a uniquely specified relation between voltage levels and internal delay time.

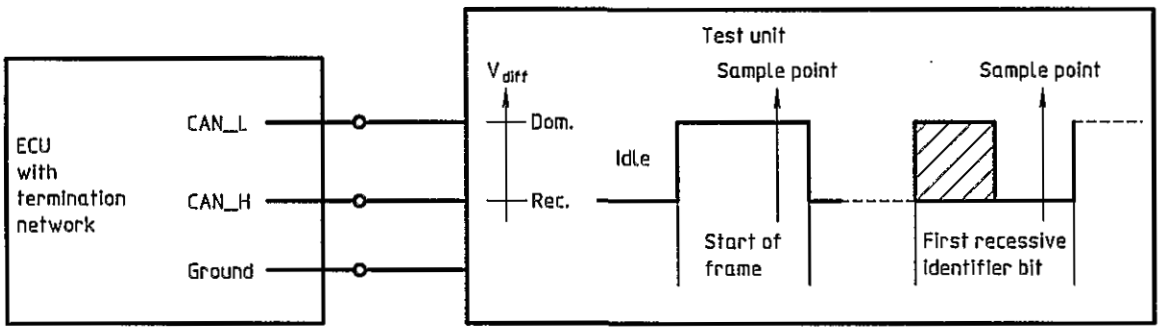

Figure 29 — Internal delay time measurement

11 Description of supervisor

11.1 Fault confinement

11.1.1 Objectives

The objective of fault confinement is to preserve a high availability of data transmission system also in case of a defect of a node. Therefore the fault confinement strategies have to prove reliable on:

— distinction between temporary errors and permanent failures;

— localization and switching-off of faulty nodes.

11.1.2 Strategies

Any node is supplied with a transmit error counter and a receive error counter. The first of them registers the number of errors during the transmission, and the other registers errors during the reception of frames.

If frames are sent or received correctly, the counters are decreased. In case of errors the counters are increased more than they are decreased in case of absence of errors. The ratio in which the counters are increased/decreased depends on the acceptable ratio of invalid/valid frames on the bus. At any time levels of the error counters reflect the relative frequency of previous disturbances.

Depending on predetermined counter values, the behaviour of nodes in respect to errors is modified. I.e. this ranges from a prohibition of sending error flags in order to cancel frames, up to switching-off of nodes which often would send invalid frames.

11.1.3 Fault confinement interface specification

See figure 30.

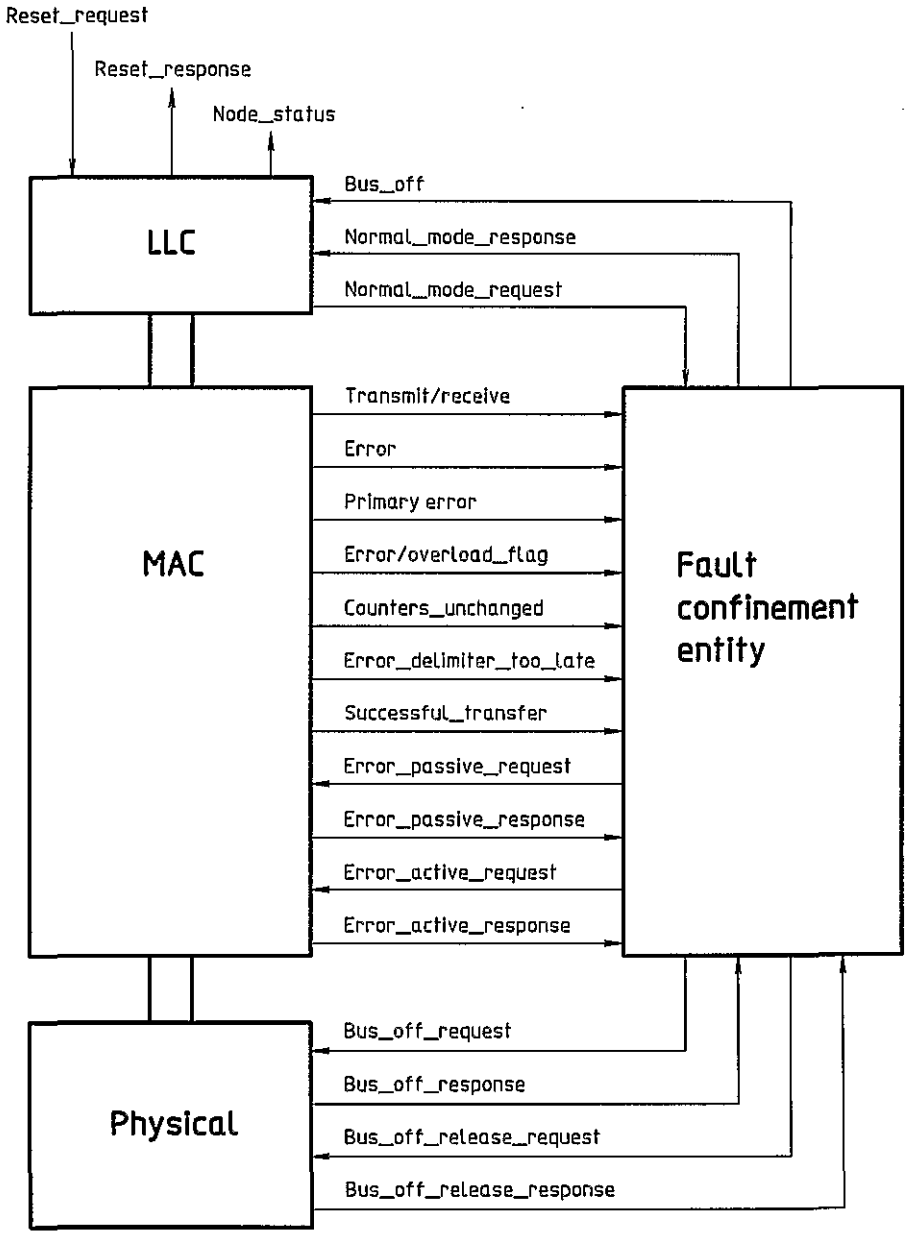

Figure 30 — Fault confinement interface

11.1.3.1 LLC sublayer/fault confinement entity interface

The messages interchanged between the Fault Confinement Entity (FCE) and the LLC sublayer are given in table 16.

Table 16 — Messages between LLC and FCE

a) LLC to FCE messages	
LLC to FCE message	**Meaning**
Normal_Mode_Request	Resets FCE to initial state.[1)]
b) FCE to LLC message	
FCE to LLC message	**Meaning**
Normal_Mode_Response	Response to the Normal_Mode_Request.
Bus_Off	Indicates that the node is in the bus off state.
1) The values of the transmit error counter and the receive error counter are set to zero when the FCE passes into its initial state.	

11.1.3.2 MAC sublayer/fault confinement entity interface

The messages exchanges between the fault confinement entity and the MAC sublayer are given in table 17.

Table 17 — Messages between MAC and FCE

a) MAC to FCE messages	
MAC to FCE messages	**Meaning**
Transmit/receive	Indicates the node's current transfer mode.
Error	Indicates that the MAC sublayer has detected an error (bit error, stuff error, CRC error, form error, acknowledgement error).
Primary_error	Signals that the MAC sublayer has detected a "dominant" bit after sending an error flag (indicates that the MAC sublayer has detected a "primary" error and not an error that is caused by the error flag of another node).
Error/overload flag	Indicates that the MAC sublayer is sending an error flag or an overload flag.
Counters_unchanged	Indicates that the FCE counters remain unchanged (due to special cases, see 11.1.4, rule 3).
Error_delimiter_too_late	Indicates that the MAC sublayer is waiting too long for error delimiter. This signal is set each time after a sequence of eight consecutive dominant bits have been received after sending an error flag.
Successful_transfer	Indicates that transmission/reception was successfully completed.
Error_passive_response	Indicates that the node was set into the "error passive" state.
Error_active response	Indicates that the node was set into the "error active" state again.
b) FCE to MAC messages	
FCE to MAC messages	**Meaning**
Error_passive_request	Request to set the node into the "error passive" state.
Error_active_request	Request to set the node into the "error active" state.

11.1.3.3 Physical layer/fault confinement entity interface

The messages that are exchanged between the Fault Confinement Entity (FCE) and the Physical Layer (PL) are given in table 18.

Table 18 — Messages between FCE and PL

a) FCE to PL message	
FCE to PL messages	**Meaning**
Bus_off_request	Request to switch off the node from the bus.
Bus_off_release_request	Request to set the node into the normal transmit/receive node.
b) PL to FCE message	
PL to FCE messages	**Meaning**
Bus_off_response	Response to bus_off_request.
Bus_off_release_response	Response to bus_off_release_request.

11.1.4 Rules of fault confinement

11.1.4.1 With respect to fault confinement, a node may be in one of the three states, depending on the error counter levels (see 4.12 to 4.14):

— "error-active",
— "error-passive",
— "bus off".

The error counters are modified according to the following rules (more than one rule may apply during a given frame transfer).

a) When a receiver detects an error, the receive error counter will be increased by 1, except when the detected error was a bit error during the sending of an active error flag or an overload flag.

b) When a receiver detects a "dominant" bit as the first bit after sending an error flag, the receive error counter will be increased by 8.

c) When a transmitter sends an error flag, the transmit error counter is increased by 8.

   Exception 1: If the transmitter is "error-passive" and detects an acknowledgement error because of not detecting a "dominant" ACK and does not detect a "dominant" bit while sending its passive error flag.

   Exception 2: If the transmitter sends an error flag because a stuff error occurred during arbitration, and should have been "recessive", and has been sent as "recessive" but monitored as "dominant".

   In exceptions 1 and 2 the transmit error counter remains unchanged.

d) If a transmitter detects a bit error while sending an active error flag or an overload flag, the transmit error counter is increased by 8.

e) If a receiver detects a bit error while sending an active error flag or an overload flag, the receive error counter is increased by 8.

f) Any node tolerates up to 7 consecutive "dominant" bits after sending an active error flag, passive error flag or overload flag. After detecting the fourteenth consecutive "dominant" bit (in case of an active error flag or an overload flag) or after detecting the eighth consecutive "dominant" bit following a passive error flag, and

after each sequence of additional eighth consecutive "dominant" bits, every transmitter increases its transmit error counter by 8 and every receiver increases its receive counter by 8.

g) After the successful transmission of a frame (getting ACK and no error has been detected until End of Frame is finished), the transmit error counter is decreased by 1 unless it was already 0.

h) After the successful reception of a frame (reception without error up to the ACK slot and the successful sending of the ACK bit), the receive error counter is decreased by 1, if it was between 1 and 127. If the receive error counter was 0, it stays 0, and if it was greater than 127, then it will be set to a value between 119 and 127.

11.1.4.2 If during system start-up, only one node is online and if this node transmits some frame, it will get no aknowledgement, detect an error and repeat the frame. It can become "error-passive" but not "bus off" because of this.

A node which is switched off or "bus off" has to run through a start-up routine in order to:

a) synchronize with already available nodes before starting to transmit. Synchronization is achieved when 11 "recessive" bits equivalent to:

— ACK delimiter + End of Frame + Intermission

or

— Error/overload delimiter + Intermission have been detected;

b) wait for other nodes without becoming "bus off" if there is no other node available at the moment.

11.1.5 "Error-active", "error-passive"

If the transmit error counter or the receive error counter of a node exceeds 127 (carry condition in case of a 7-bit receive error counter) then the supervisor requests the MAC sublayer to set the corresponding node into the "error-passive" state.

An error condition letting a node become "error-passive" causes the node to send an active error flag.

An "error-passive" node becomes "error-active" again when both the transmit error counter and the receive error counter are less than or equal to 127 (see figure 31).

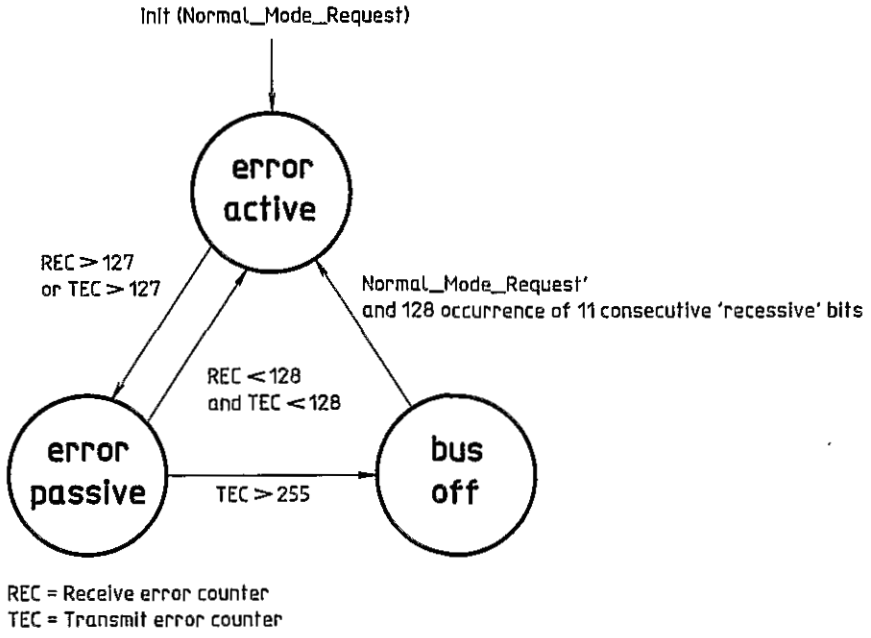

Figure 31 — Node status transition diagram

11.1.6 "Bus off" management

If the transmit error counter of a node is greater than 255 (carry condition in case of a 8-bit transmit error counter) then the supervisor requests the physical layer to set the node into the "bus off" state.

A "bus off" state is not allowed to have any influence on the bus. It shall not send any frames nor send acknowledgement, error frames, overload frames. Whether such a node receives frames from the bus depends on the implementation.

A node which is "bus off" is permitted to become "error-active" (no longer "bus off") with its error counters both set to 0 after having monitored 128 occurrences of 11 consecutive "recessive" bits on the bus (see figure 31).

11.2 Bus failure management

During normal operation, several bus failures may occur that may influence the bus operation. These failures and the resulting behaviour of the network are specified in table 19. The possible open circuit and short circuit failures are shown in figure 32.

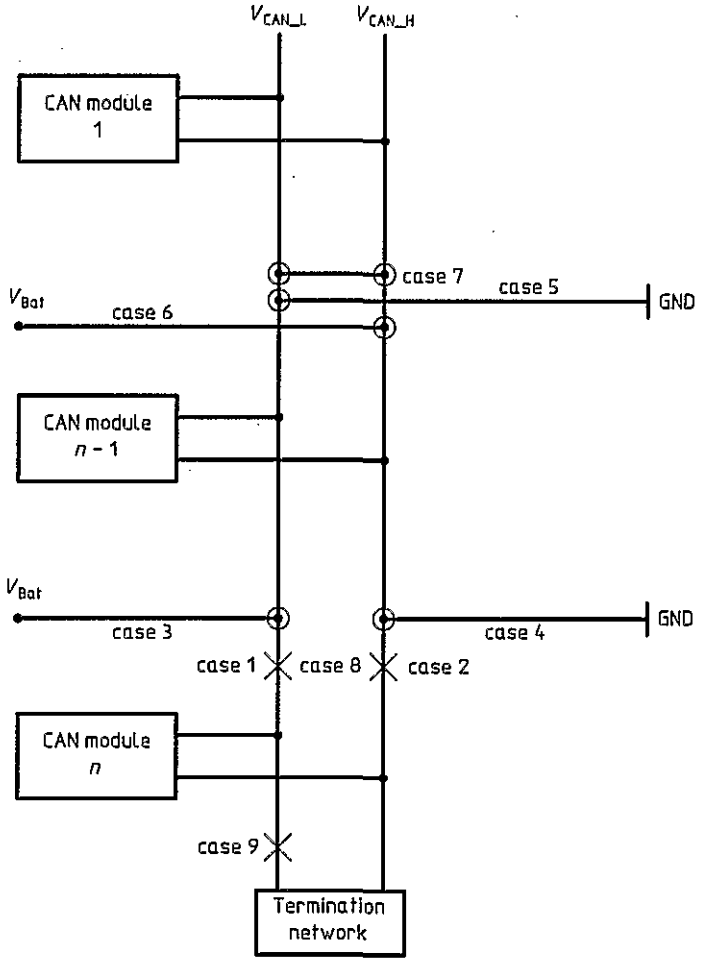

Figure 32 — Possible failures of bus line

Table 19 — Bus failure detection

Description of bus failures		Behaviour of network[1]	Quality of specification[2]
One node becomes disconnected from the bus		The remaining nodes continue communicating.	Recommended
One node loses power		The remaining nodes continue communicating with reduced signal-to-noise ratio.	Recommended
One node loses ground		The remaining nodes continue communicating with reduced signal-to-noise ratio.	Recommended
Loss of the shield connection at any node[3]		All nodes continue communicating.	Recommended
Open and short failures[4]			
	1 CAN_H interrupted	All nodes continue communicating with reduced signal-to-noise ratio.	Recommended
	2 CAN_L interrupted		
	3 CAN_H shorted to battery voltage		
	4 CAN_L shorted to ground		
	5 CAN_H shorted to ground		
	6 CAN_L shorted to battery voltage		
	7 CAN_L wire shorted to CAN_H wire	All nodes continue communicating with reduced signal-to-noise ratio.	Optional
	8 CAN_H and CAN_L wires interrupted at the same location	No operation within the complete system. Nodes within the resulting subsystem that contains the termination network continue communicating.	Recommended
	9 Loss of one connection to termination network	All nodes continue communicating with reduced signal-to-noise ratio.	Recommended

1) The example in figure 32 excludes all fault tolerant modes.

2) The quality of specification is defined as follows.

Mandatory:	If the respective failure occurs the network shall behave as described in the second column of the table.
Recommended:	If the respective failure occurs the network behaviour should be as described in the second column of the table. To exclude this specified functionality is the manufacturer's choice.
Optional:	If the respective failure occurs the network behaviour may be as described in the second column of the table. To include this fuller specified functionality is the manufacturer's choice.

3) This failure should be considered only if a shielded cable is used. In this case the loss of shield connection at one node may cause common mode voltage induced between the shield and the two signal wires.

4) The numbers 1 to 9 refer to cases 1 to 9 in figure 32.

UDC 629.113.018.7:681.327.8

Descriptors: road vehicles, control devices, numerical control, data processing, information interchange, data transmission, local area networks, physical layer, data link layer, communication procedure.

Price based on 58 pages