Basics of the CAN Bus

Version: 18.08.2016

General introduction

Interfaces are used to transfer information between the individual components of a system.

In a bus system, all components are connected to a common data line via short stitch lines.

This minimizes the cost of cabling and makes it easy to connect additional components.

However, the flow of data must be controlled by an access procedure (protocol) when all components use a common bus line. As far as possible, components from different manufacturers should also work together.

The Controller Area Network (CAN) connects several equal components (node) via a 2-wire bus plus additional ground line. The CAN protocol was developed by Bosch in 1983 for use in motor vehicles and was first presented to the public in 1986.

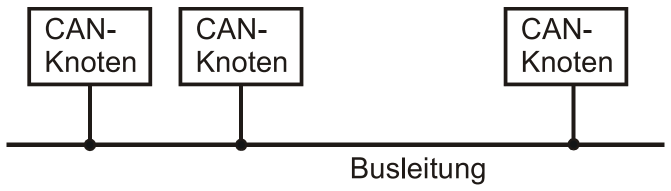

Figure 1: Controller Area Network (CAN)

Due to its high level of interference safety, low costs and real-time capability, CAN is used not only in the automotive industry, but also in many other industries (e.g. commercial vehicles, mobile machinery, railways, medical technology, industrial automation, elevators, and machine bus systems).

The organization "CAN in Automation" (CiA) is dedicated to the further development of the CAN protocol and the specification of the CANopen application protocol as well as the CANopen profiles.

Physical description of the CAN interface

The physical CAN transmission is standardized in ISO 11898-2 (high-speed) and ISO 11898-3 (low-speed). Transceivers from various manufacturers, such as the PCA82C250 from NXP, are available to implement this specification.

The electrical interference safety is achieved, among other things, by the fact that one bit is mapped on two lines simultaneously with a counter-intuitive potential change. We are also talking about a differential signal.

Thus, a redundantly inverted transmission of the logical signal is performed on a second line.

Faults scattered into the line act on both lines in the same direction. However, since the two differential lines always have opposite levels, the difference between the levels is largely maintained even in the event of disturbances. This is called common mode rejection ratio (CMRR). The CAN-High and CAN-Low lines contain the inverted and non-inverted serial data signals.

By designing as an open collector (PNP to V_{CC} at CAN-H and NPN on GND at CAN-L), several participants on the bus can also be connected in parallel without creating electrical short circuits in the event of a conflict.

The state with two different levels on CAN-H and CAN-L is called the dominant state (level difference: 2.0 volts nominal); the state with two equal levels is called recessive (level difference: 0.0 volts nominal).

The dominant state corresponds to a logical zero: If a node places a logical zero on the bus, it may overwrite the state of a logical one of another node. The coupling of the nodes over the bus line is a logical and link (Wired-And).

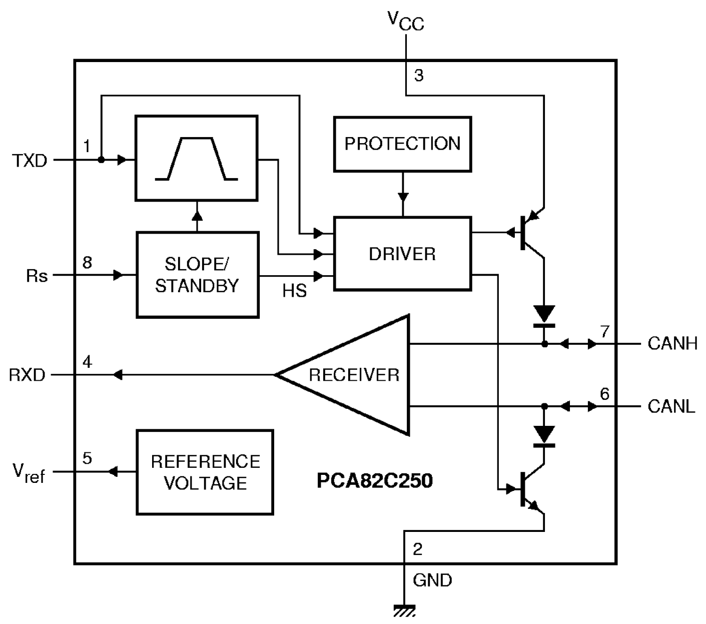

Figure 2: Transceiver PCA82C250 of NXP

logical level	state	CAN-H	CAN-L	difference level
0	dominant	Transistor durchgeschaltet (zieht Pegel auf VCC)	Transistor durchgeschaltet (zieht Pegel auf GND)	2 Volts
1 or high-impedance	rezessiv or floating	Transistor locked	Transistor locked	0 Volts

Table 1: display of logical levels in CAN

Another measure to increase interference safety is NRZ coding, i.e. there is not a flank change in every bit. In order to avoid that the participants lose the synchronization to the transmitter, one bit of the other polarity is inserted by the sender after five bits of the same "polarity" (recessive or dominant). The recipients automatically remove these stuff bits so that the sent bit sequence and the one forwarded to the host controller are the same.

Plug assignment

As a CAN connector, the 9-pin Sub-D connector proposed by CAN in Automation (CiA) has established itself in many applications. Both "female" and "male" connectors are used simultaneously in the nodes. Thus, additional nodes can be integrated into the bus line without interruption. ("Daisy-chain"-Wiring)

At least one 3-pin cable with CAN-High, CAN-Low and Ground is required for the transmission of CAN signals. The use of a shielded cable is not mandatory.For larger cable lengths, twisting of the line pair and shielding are recommended.

CiA has also recommended pin assignment for other connectors (CiA 303-1): for example, for the 5-pin M12 round connector and 4- and 5-pin "open style" connectors without going into the dimensions. Another useful thing is the 10-pin "multipole connector" defined by CiA, where pin 10 is reserved and not allowed to be used.

PIN	Signal	Description
1		reserved
2	CAN-L	negated CAN-Signal (Dominant Low)
3	CAN-GND	ground
4		reserved
5	CAN-SHLD	shielding (optional)
6	GND	Devices ground (optional)
7	CAN-H	positive CAN Signal (Dominant High)
8		reserved
9	VCC	Supply voltage (optional)

Table 2: CAN assignment of the D-SUB 9 plug connector

Bit rate and line lengths

In principle, the CAN network can transmit bit rates up to 1 Mbps. All CAN nodes must be able to process the message at the same time. The maximum cable length is therefore dependent on the bit rate. The table shows recommended bit rates and the corresponding maximum cable length.

The achievable length also depends on the cable used, the network topology and the sampling time.

Bitrate	Cable length
10 kbits/s	6,7 km
20 kbits/s	3,3 km
50 kbits/s	1,0 km
125 kbits/s	500 m
250 kbits/s	250 m
500 kbits/s	125 m
1 Mbits/s	25 m

Table 3: Relationship between CAN bit rate and maximum cable length

Bus scheduling (completion resistance)

The bus scheduling takes place on the CAN bus with a line topology with 120 ohms at both ends of the network. Scheduling is also recommended for short lines with low bit rates. Without scheduling, there are reflections. In practice, for short lines, a scheduling at one end is sufficient, but ideally the bus is scheduled at both ends (and only there) with 120 ohms each.

Principle of data exchange in the CAN network

When data transmission in a CAN bus, no nodes are addressed, but the contents of a message (e.g. speed or motor temperature) are indicated by a unique identifier. In addition to content flag, the identifier also sets the priority of the message.

With the following acceptance check, all stations determine whether the received data is relevant to them or not after the message has been received correctly using the identifier (ID). The content-related addressing achieves a high degree of flexibility: stations can be added to the existing CAN network very easily.

All messages are heard by all participants ("broadcast") and forwarded to the host controller for processing, depending on the ID acceptance filtering. Measures that are required by several ECUs as information can be distributed over the CAN network in such a way that not every control unit needs its own sensor.

Collision detection

Each participant may send messages without a special request from another participant (e.g. Master). As with Ethernet, multiple subscribers may send at the same time. The message with the lowest identifier (the lowest binary value) receives permission to send.

The process for collision checking via the identifier is called "bitwise arbitration". According to the "wired-and-mechanism", in which the dominant state (logically 0) overwrites the recessive state (logically 1), all those nodes lose the competition for the bus allocation, which send recessively but detect a dominant bit on the bus. All "losers" automatically become recipients of the message with the highest priority and do not try to send again until the bus becomes free.

Simultaneous bus access by multiple nodes must always result in a unique transmission permission, so the identifiers must be clearly assigned, i.e. they must not be used by two participants at the same time. By the method of bitwise arbitration via the identifiers of the messages to be transmitted, each collision is clearly resolved after a predictable time: For messages in the basic format (11-bit ID) there are a maximum of 13 bit times (29-bit ID), in the extended format it is a maximum of 33 bit times. The above-mentioned stuffbits are not taken into account.

Layers of CAN software and CAN hardware

The individual tasks of CAN communication are carried out according to the IOS/OSI reference model in "layers".

- Physical Layer: This layer describes the physical properties, such as signal level, transmission speed, sampling time, plug, cable, etc. It is partially realized in the CAN controller and in the CAN transceiver.

- Data Link Layer: This is the actual CAN protocol with its message formats (data telegrams, remote request telegram, error telegram and overload telegram) as well as the error behavior ("fault confinement").
- The higher protocols: the layers above are usually not individually identified and are usually implemented in software on the host controller. In some industries, these higher protocols are standardized (for example, CANopen, DeviceNet, SAE J1939). The automotive industry has standardized a transport protocol in ISO 15675 internationally, with which one can segment long messages with more than 8 bytes on the transmitter side and reassemble them on the receiver side.

Building a CAN message

A message is packaged in a form specific to the CAN bus. This packaging is called a "frame".

A frame consists of seven characteristic fields:

- Start-of-frame bit
- Arbitration field (CAN identifier plus RTR bit)
- Control panel (contains the data length code
- Data field (0 to 8 bytes)
- CRC field (contains a 15-bit checksum and an end marker)
- Acknowledge field (ACK bit plus end marker)
- End-of-frame

There are two frame formats that differ mainly in the length of the identifier:

- Base format (11-bit identifier)
- Extendet format (29-bit identifier)

The types of frames are different:

- Data Frame (message is sent without a special prompt)
- Remote Transmission Request (RTR) Frame (message is requested— the recipient who "owns" the message with the requested identifier returns the corresponding data frame)

Base frame according to ISO 11898-1 (formerly also known as CAN 2.0A):

Start	Identifier	RTR	IDE	r0	DLC	DATA	CRC	ACK	EOF+IFS
1 Bit	11 Bit	1 Bit	1 Bit	1 Bit	4 Bit	0...8 Byte	16 Bit	2 Bit	10 Bit

Extended frame according to ISO 11898-1 (formerly known as CAN2.0B):

Start 1 Bit	Identifier 11 Bit	SRR 1 Bit	IDE 1 Bit	Identifier 18 Bit	RTR 1 Bit	r1 1 Bit	r0 1 Bit	DLC 4 Bit	DATA 0...8 Byte	CRC 16 Bit	ACK 2 Bit	EOF+IFS 10 Bit

- Start: dominant, is used for synchronization,
- Identifier: information for the recipient and priority information for bus scarring,
- RTR: distinguishes between data (dominant) and RTR telegram (recessive),
- IDE: Identifier Extension,
- r0, r1: reserved,
- DLC: contains the length information of the following data field,
- DATA: contains the data of the telegram,
- CRC: contains the CRC checksum and the CRC end detection for the previous bit sequence,
- ACK: contains a confirmation from other participants when the message is received correctly,
- EOF: indicates the end of the data telegram (seven recessive bits),
- IFS: 3-bit-long space between CAN frames,
- SRR: replaces the RTR bit in the extended frame,
- IDE: indicates that another 18 bits will follow,

Error detection in the CAN network

The CAN protocol can detect and signal errors itself. To detect errors, the CAN protocol implements three mechanisms at the message level:

1. Cyclic Redundancy Check (CRC)

The CRC ensures the information of the frame by adding redundant test bits on the send side. On the receiving side, these test bits are recalculated from the received bits and compared with the received test bits. A CRC error occurs in the event of nonconformance.

2. Frame-check

This mechanism checks the structure of the transferred framework. The errors detected by Frame-Check are called format errors.

3. ACK-error

Of all receivers, the received frames are acknowledged by positive acknowledgment (the recessive bit of the transmitter is "overwritten" by dominant bits of the receivers). If no acknowledgement is detected on the transmitter (ACK error), this indicates a transmission error that may only be detected by the receivers, a falsification of the ACK field, or non-existent receivers.

In addition, the CAN protocol implements two bit-level error detection mechanisms.

1. Monitoring

Each node that sends simultaneously observes the bus level. It detects differences between the sent and received bits. This allows you to safely detect all global errors and bit errors that occur locally at the transmitter.

2. Bit-stuffing

At the bit level, the encoding of the individual bits is checked. The CAN protocol uses non-return-to-zero (NRZ) encoding, which ensures maximum bit encoding efficiency. The synchronization flanks are generated according to the method of bit-stuffing by inserting a stuff bit with complementary value into the bit stream after five consecutive equivalent bits, which the receivers automatically remove. If one or more errors are detected by at least one node using the mechanisms described above, the ongoing transmission is aborted by sending an "error flag". This prevents the transmitted message from being accepted by other stations and thus ensures second-level data consistency. After canceling the transmission of a bad message, the sender automatically starts to resend its message (Automatic Repeat Request).

If an error occurs several times in a row, this will automatically shut down the node.

Effective transfer rate for data bytes

Despite the self-access of a CAN node to the bus line, a node of the highest priority can be used to specify the effective transmission rate. A message in the base format with eight data bytes requires a maximum of 130 bits. A maximum number of 19 stuff bits and 3 space bits is assumed:
1 Start bit
+11 Identifier bits
+ 1 RTR bit
+ 6 Control bits
+ 64 Data bits
+ 15 CRC bits
+ 19 (maximum) Stuff bits
+ 1 CRC delimiter
+ 1 ACK slot
+ 1 ACK delimiter
+ 7 EOF bits
+ 3 IFS (Inter Frame Space) bits
= 130 bits

A maximum of 154 bits must be transferred in the extended format.

The net data rate is the number of data bits divided by the total number of bits times the transfer rate.

The following table lists the effective transfer rate for different data bytes. The variable number of stuff bits is not taken into account:

	effective transmission rate in kBit/s at 1000 kBit/s	
Number of data bytes per frame	with Standard ID	with Extended ID
1	145	107
2	254	193
3	338	264
4	405	323
5	460	374
6	505	417
7	432	455
8	576	489

Table 4: effective transmission rate

The GSV-3CAN measuring amplifier, which transmits a measurement value of three data bytes per frame, can transmit a maximum of 338*1024/24 = 14421 readings per second over the CAN network at 1 Mbps.

In the packed format of the GSV-3CAN, on the other hand, a frame with 8 data bytes is used, which contains four readings of two bytes each.

This increases the number to 576*1024/64*4 = 36864 readings per second, which can be recorded via the CAN network.

Measuring amplifier with CAN for strain gauges

The measuring amplifiers GSV-2, GSV-3, GSV-6 and GSV-8 are available for the evaluation of the signals of strain gauges (DMS).

Product	Description
GSV-2LS	Circuit board; RS232 and CAN, CANopen Protocol;
GSV-2AS	IP66 Alu housing, RS232 and CAN, CANopen Protocol;
GSV-2FSD	Front panel installation; Display, RS232 and CAN, CANopen protocol;

Product	Description
GSV-2TSD-DI	Desktop device with battery; Display, RS232, USB and CAN, CANopen protocol;
GSV-3CAN	IP66 Alu housing, CAN
GSV-6CPU	Printed circuit board, 3.3V, UART and CAN, each without transceiver
GSV-8DS	Desktop 8-channel amplifier; UART, USB, CAN with CANopen protocol
GSV-8AS	IP66 Alu housing, 8-channel amplifier; UART, USB, CAN with CANopen protocol

Table 5: Measuring amplifier for strain gauges with CAN

Related links

http://www.can-cia.org/	CAN in Automation (CiA) is the international user and manufacturer association for the Controller Area Network (CAN).