Controller Area Network (CAN)

Dr. Vivek Kaundal

L&T Technology Services

LTTS
GLOBAL
ENGINEERING
ACADEMY

Agenda

- Introduction to CAN
- CAN Frames
- CAN Error Detection
- CAN SYNCRONIZATION
- CAN Controller
- CAN Physical Layer
- Questions

Version

Version	Reviewed by	Approved by	Remarks
1.0	Akkaladevi Venkat Ramana	Dr. KNS Acharya	

Embed Syllabus of the Couse on this slide

Learning Outcome of the Course

At the completion of the Course participants will exhibit competencies / Skills of:

1. Understanding the CAN protocol in detail.
2. Understand the complete frame of CAN protocol.
3. Understand the CAN physical layer and further design the same frame in Automotive tools.

Pre-Requites for the Course

1. Basic domain knowledge of Automotive Electronics

2. Basic knowledge of protocols

Introduction to Controller Area Network (CAN)

Foundation of CAN

- The foundations of the CAN concept were laid, at a given period, on the basis of a state of the market and technology.
- For a long time, many companies developed their own solutions for In- vehicle communication.
- Nearly all of these solutions are known as ‘proprietary’.
- Because of the different interests of each of the companies involved, this has led to a significant fragmentation of the market.

History

- Controller Area Network (CAN) was initially created by German automotive system supplier Robert Bosch in the mid-1980s for automotive applications as a method for enabling robust serial communication

- The goal was to make automobiles more reliable, safe and fuel-efficient while decreasing wiring harness weight and complexity

History

- CAN was first developed by Robert Bosch GmbH, Germany in 1986 when they were requested to develop a communication system between three ECUs (electronic control units) in vehicles designed by Mercedes.

- They found that point to point communications was no longer suitable in this situation. The need for a multi-master communication system became imperative.

- The first CAN silicon was then fabricated in 1987 by Intel

History

- 1983 Start of the Bosch internal project to develop an in-vehicle network
- 1986 Official introduction of CAN protocol
- 1987 First CAN controller chips from Intel and Philips Semiconductors
- 1991 Bosch’s CAN specification published

History

- 1991 CAN Kingdom CAN-based higher-layer protocol introduced by Kvaser
- 1992 CAN in Automation (CiA) international users and manufacturers group established
- 1992 CAN Application Layer (CAL) protocol published by CiA
- 1992 First cars from Mercedes-Benz used CAN network

History

- 1993 ISO 11898 standard published
- 1994 1st international CAN Conference (iCC) organized by CiA
- 1994 DeviceNet protocol introduction by Allen-Bradley
- 1995 ISO 11898 amendment (extended frame format) published.
- 1995 CANopen protocol published by CiA
- 2000 Development of the time-triggered communication protocol for CAN (TTCAN)

Concept of CAN

- CAN was designed to operate in environments subject to a high level of pollution, primarily due to electromagnetic disturbance
- CAN achieves transmission reliability by an efficient error detection mechanism
- CAN has multi-master functionality to increase the possibility of providing fast recovery from errors after their detection
- Bitwise arbitration principle provides a ‘non-destructive’ arbitration method, when several stations attempt to send a message simultaneously

Introduction

- Controller Area Network (CAN) is a serial network
- A network among sensors, microcontrollers and actuators
- Now popular in industrial automation and agricultural applications
- Two-wire system, well suited for high speed applications using short messages
- Offers communication rate up to 1 Mbits/sec allowing real-time control

Introduction

- CAN is also used in many other industrial automation and control applications
- Has an OSI session layer known as Time Triggered CAN (TTCAN)
- Has a sub-bus known as the Local Interconnect Network (LIN)
- CAN is documented in ISO 11898 (for applications up to 1Mbps) and ISO 11519 (for applications up to 125 Kbps)

Application

CAN is used in a wide area of applications

Automotive

Building automation

Medical

Factory automation

CAN

Railway

Maritime

Machine control

Application

Car Body components
Typically low-speed CAN Bus

Power Train components
Typically high-speed CAN Bus

CAN Overview

In the beginning...

- the different control systems (and their sensors) to exchange information.
- a cable network with a length of up to several miles and many connectors was required.
- growing problems concerning material cost, production time and reliability.

CAN Overview

Solution...

- point-to-point wiring is replaced by one serial bus connecting all control systems.
- → CAN

CAN Overview

CAN basic concept

- In the CAN protocol, the bus nodes do not have a specific address. Instead, the address information is contained in the identifiers of the transmitted messages, indicating the message content and the priority of the message.

- The number of nodes may be changed dynamically without disturbing the communication of the other nodes.

- Multicasting and Broadcasting is supported by CAN.

CAN Overview

CAN basic concept...

CAN basic concept..

- There are two bus states, called "dominant" and "recessive".

- The bus logic uses a "Wired-AND" mechanism, that is, "dominant bits“ (equivalent to the logic level "Zero") overwrite the "recessive" bits (equivalent to the logic level "One").

CAN Overview

CAN basic concept...

- Only if all nodes transmit recessive bits (ones), the Bus is in the recessive state.

CAN Overview

CAN basic concept...

➲ As soon as one node transmits a dominant bit (zero), the bus is in the dominant state.

Features

Bus-access by message priority

- Carrier Sense Multiple Access / Collision Resolution

Bus access conflicts resolved by arbitration

- Bit-wise
- Non-destructive
- Allows for guaranteed latency time

Message identifier

- CAN has no node addresses

Features

Extensive ERROR checking

- Five different checks
- Every connected node participate

Data consistency secured

- A message is accepted by all nodes or none

Different Bus Management Methods

- Bit-wise arbitration
- Master/Slave
- Daisy Chain
- TDMA

Features

A Higher Layer Protocol is always required

- CAN is only a low level specification

The capability of CAN is restricted by the Higher Layer Protocol chosen

- Market segment
- Real-time requirements
- Product Administration requirements
- etc

Layer Structure

All
People
Seem
To
Need
Data
Processing

Higher Layer Protocol...

Layer Structure

Higher Layer Protocol...

7: Application Layer → Interface between the data communication environment and the application

6: Presentation Layer
5: Session Layer
4: Transport Layer
3: Network Layer

→ Not explicitly specified in CAN

2: Data Link Layer →

Logical Link Control (LLC)
Acceptance filtering, overload notification and recovery management
Medium Access Control (MAC)
Data encapsulation (de-capsulation), frame coding(stuffing/de-stuffing), medium access management, error detection, error signaling, acknowledgement, and serialization (de-serialization).

1: Physical Layer →

Physical Signalling
Bit Encoding / Decoding, Bit Timing, Frame / Bit Synchronisation
Physical Medium Attachment
Driver / Receiver Characteristics
Transmission Medium
Cable, Connectors

CAN Frames

Type of Frames

Data Frame

Standard Format (CAN 2.0A): 11-bit Identifier

2^{11} identifiers possible

Extended Format (CAN 2.0B): 29-bit Identifier

2^{29} identifiers possible

Data Frame

- marks the start of any CAN frame
- it is always a dominant bit
- provides a falling edge for hard synchronization of transmitter and receivers

Frames

Data Frame

- contains the Identifier (11 bit for CAN 2.0A) which is used for arbitration
- Identifier determines frame priority: low identifier = high priority
- the highest seven bits of the identifier must not be all recessive
- Remote Transmission Request (RTR) bit is always dominant in a Data Frame

Data Frame

- contains the Identifier (29 bit for CAN 2.0B) which is used for arbitration
- Identifier determines frame priority: low identifier = high priority
- the highest seven bits of the identifier must not be all recessive
- Remote Transmission Request (RTR) bit is always dominant in a Data Frame
- Substitute Remote Request (SRR) bit is always recessive in a Data Frame

Frames

Data Frame

- Identifier Extension (IDE) bit is dominant for Standard Frames and recessive for Extended Frames
- r0 bit is not used (“reserved for future extensions”)
- Data Length Code (DLC, 4 bits) indicates number of data bytes in Data Field;
  may take values ranging from 0 to 8, other values are not allowed

Frames

Data Frame

0 Data Bytes	e.g. to indicate an event
1 Data Byte	low net data rate on the CAN bus
8 Data Bytes	high net data rate on the CAN bus

- contains the actual information which is transmitted
- number of data bytes may range from 0 to 8 <u>in units of bytes</u>
- number of data bytes is given in the Data Length Code (DLC)
- transmission starts with the first data byte (byte 0), MSB first

Frames

Data Frame

- contains the 15-bit Cyclic Redundancy Check (CRC) code
- use polynomial generator $x^{15} + x^{14} + x^{10} + x^{8} + x^{7} + x^{4} + x^{3} + 1$
- CRC is a complex, but fast and effective error detection method
- the CRC Field Delimiter (DEL) marks the end of the CRC field
- the CRC Field Delimiter is always recessive

Frames

Data Frame

- contains the Acknowledgement (ACK Slot) bit
- the Acknowledgement bit can be dominant (TX) or recessive (RX)
- the ACK Field Delimiter (DEL) marks the end of the ACK field
- the ACK Field Delimiter is always recessive

Frames

Data Frame

- consists of seven consecutive recessive bits
- marks the end of the Data Frame
- follows the Acknowledge (ACK) field
- is followed by the Interframe Space (IFS)

Acknowledgement procedure

A frame must be acknowledged by at least one other node. Otherwise ACK-Error

Remote Frame

- Used to request transmission of a specific Data Frame
- Similar to a Data Frame, but without Data Field
- Remote Transmission Request (RTR) bit is recessive
- Same identifier as the Data Frame which is requested
- Note: When Remote Frame is transmitted at the same time as corresponding Data Frame, Data Frame wins arbitration because of dominant RTR bit

How is used Request Frame

➲ If a node wishes to request the data from the source, it sends a Remote Frame with an identifier that matches the identifier of the required Data Frame. The appropriate data source node will then send a Data Frame as a response to this remote request.

Inter frame Space (IFS)

- consists of three consecutive recessive bits
- separates two consecutive frames from each other
- no transmission is allowed during the Inter frame Space (IFS)
- is needed by controllers to copy received frames from their Rx buffers
- ACK Field Delimiter + EOF + IFS = 11 consecutive recessive bits
- might be followed by “Bus Idle” sequence of indefinite length

Overload Frame

- Unit sends Overload Frame when at present it cannot receive frames any more due to high workload
- Transmission of an Overload Frame is started during the first two bits of the Inter frame Space (IFS) of the preceding frame
- Other units react immediately by also transmitting Overload Frames
  ⇒ Overload Flags overlap, resulting in up to 12 consecutive dominant bits
- Implemented in very few (mostly older) controllers, though controllers must still be able to interpret correctly Overload Frames they receive
- Overload Frames do not influence the error counters (TEC and REC)

Active Error Frame

- Sender and receivers reject erroneous frame completely and do not process it any further
- Error flag actively violates the bit stuffing rule.
- After the Error Delimiter bus activity returns to normal and the interrupted node attempts to re send the aborted message.

Passive Error Frame

- can still receive frames like a unit in error active state
- has to wait after transmission of a Data Frame for 8 recessive bit cycles on the bus (Suspend Transmission Field) until it is permitted to transmit another Data Frame
- can go back to error active state for TEC <= 127 AND REC <= 127

Quiz time

- **What is the Standard CAN Frame Formats?**

Quiz time

- **What are the two types of frame format of CAN?**

Quiz time

- **What are the two types of frame format of CAN?**

Solution:

- Base frame format: with 11 identifier bits
- Extended frame format: with 29 identifier bits

Quiz time

- **What is the length of the CRC and CRC delimiter?**

Quiz time

- **What is the length of the CRC and CRC delimiter?**
- **Solution:**
  - **CRC= 15**
  - **CRC Delimiter =1**

Quiz time

- **What is Featured of CAN Protocol?**

Quiz time

- **What is Featured of CAN Protocol?**
- Simple and flexibility in Configuration.
- CAN is Message-Based Protocol.
- Message prioritization feature through identifier selection.
- CAN offer Multi-master Communication.
- Error Detection and Fault Confinement feature.
- Retransmission of the corrupted message automatically when the bus is idle.

CAN

Error Detection

Overview

Following error types are detected:

- Bit error — transmitted and received bit are different (except in arbitration, acknowledgement and passive error)
- Bit stuffing error — more than five bits of equal polarity inside of a frame are detected
- CRC error — receive CRC code doesn't match with the calculated code
- ACK error — transmitting node receives no dominant acknowledgement bit (no receiver accepts the transmission message)
- Form error — fixed-form bit field contains one ore more illegal bits
  e. g. violation of end of frame EOF format, CRC- or ACK-delimiter

Error Detection

Cyclic Redundancy Check

- This CRC sequence is transmitted in the CRC Field of the CAN frame.
- The receiving node also calculates the CRC sequence using the same formula and performs a comparison to the received sequence.

- If node B detects a mismatch between the calculated and the received CRC sequence , then a CRC error has occurred.
- Node B discards the message and transmits an Error Frame to request retransmission of the garbled frame.

Error Detection

Acknowledgement procedure

➲ Acknowledge Error has occured and the message has to be repeated. No Error Frame is generated, though.

Error Detection

Frame Check.

If a transmitter detects a dominant bit in one of the four segments:

- CRC Delimiter,
- Acknowledge Delimiter
- End of Frame
- Interframe Space

then a Form Error has occurred and an Error Frame is generated. The message will then be repeated.

Error Detection

Bit Monitoring

Bit Error occurs if a transmitter

- Sends a dominant bit but detects a recessive bit on the bus line or,
- Sends a recessive bit but detects a dominant bit on the bus line.
- An Error Frame is generated and the message is repeated.
- When a dominant bit is detected instead of a recessive bit, no error occurs during the Arbitration Field or the Acknowledge Slot because these fields must be able to be overwritten by a dominant bit in order to achieve arbitration and acknowledge functionality.

Error Detection

Bit Stuffing Check

- If six consecutive bits with the same polarity are detected between Start of Frame and the CRC Delimiter, the bit stuffing rule has been violated.
- A stuff error occurs and an Error Frame is generated. The message is then repeated.

Error Detection

Handling of the error management unit

- The error management unit is used to:
  - cancel erroneous messages at all CAN nodes
  - to disconnect nodes from the CAN bus, if they detect/generate to many errors
- The error management unit uses two error counters:
  - a receive error counter REC
  - a transmit error counter TEC
- The CAN controller may work in three different states:
  - error active state — for each detected error there is generated an active error flag (queue of six consecutive dominant bits)
  - error passive state — for each detected error there is generated an passive error flag (queue of six consecutive recessive bits)
  - bus off state — if to many errors are detected by one node, this node is automatically disconnected from the bus

Status of a CAN Node

The CAN node status levels are switched depend on the values of TEC and REC:

- Error active Transmit Error Counter TEC and Receive Error Counter REC are less than 128 (CAN node status after reset: TEC = REC = 0)
- Error passive TEC or REC is greater than 127 and TEC is less than 255
- Bus off TEC is greater than 255

Status of Node	Error Counter	Generated Error Flag
Error active	*TEC and REC ≤ 127*	*6 dominant bits*
Error passive	*TEC or REC > 127 and ≤ 255*	*6 recessive bits*
Bus off	*TEC > 255*	

Operational States of a CAN Controller

Bus off recovery sequence:

(Reset+Configuration or Configuration) and occurrence of 128 times 11 recessive bits

Steps of CAN Error Handling

- Each CAN node which detects an error - local or global error - sends an error flag to inform all other stations about this error (globalization)
- An error flag is a queue of six consecutive bits

Error active node :

Error condition

Error passive node

Error condition

➔ In the case of one node detects a local error it will send an error flag. Other nodes may detect an error because of this error flag. They send also an error flag.

➔ After receiving/sending an error flag all nodes cancel the message.

➔ Each node which detects an error increments his error counter(s).

➔ The transmitter of a cancelled message retransmits automatically this message.

The Error Counters

- The ERROR MANAGEMENT UNIT serves two ERROR COUNTER

- **Successful transmission** (incl. ACK)

- **Bit Error** detected (e.g. can't write dominant bit)
- **8th consecutive dominant bit following an Error Flag**
- **no Acknowledge bit** (only in Error Active mode)
- **error detected in EOF**

- **Successful reception** (incl. ACK)

- **node detects an error (+1)**
- **error detected in first 6 bits of EOF (+1)**
- **transmit node is sending an Error Flag (+8)**
- **Bit Error** detected **(+8)**
- **8th consecutive dominant bit following an Error Flag (+8)**

BUS OFF

A unit is in bus off state when

- its Transmit Error Counter (TEC) is greater than 255: TEC > 255
- Note: The value of the Receive Error Counter (REC) is of no importance

In bus off state a unit

- is practically disconnected from the bus
- cannot receive and transmit anything any more
- can only leave bus off mode via a hardware reset OR a software reset and subsequent initialization carried through by the host (BOSCH specification: TEC and REC are set to zero and the unit must receive 128 times a field of 11 consecutive recessive bits)

The Warning Level for a unit is set when

- its Transmit Error Counter (TEC) is greater than 96: TEC > 96 AND / OR
- its Receive Error Counter (REC) is greater than 96: REC > 96

When a unit reaches the Warning Level

- it is indicated to the controller CPU that the unit is “heavily disturbed”
- an “Error Warning Flag” is set
- optionally, an interrupt might result from this
- the “Error Warning Flag” can be cleared for TEC <= 96 AND REC <= 96

Practical use of the Warning Level

- by constantly checking the “Error Warning Flag”, it can be determined whether a unit gets near the threshold to the error passive state
- unfortunately, by checking this flag, one cannot determine whether a unit is still in error active state or already in error passive state

Error Detection

Receive Error Counter REC

The probability for not discovering an error is

$$4.7 * 10^{-11}$$

Example 1*

A CAN bus is used	365 days / year 8 hours / day
with a transmission speed of	500 kBit / sec
and errors arise every	0.7 seconds

⇒ in 1.000 years, only one error remains undiscovered

Example 2**

A CAN bus in a car is runing at	500 kBit / sec
with an average bus load of	15 %
an average data frame size of	110 bits
for an average operating time of	4000 hours

⇒ only one error in 100.000 automobiles remains undetected

*Source: CiA

**Source: Kaiser, Schröder: “Maßnahmen zur Sicherung der Daten beim CAN-Bus”

Quiz Time

What is Cyclic Redundancy Check?

Quiz Time

What is Cyclic Redundancy Check?

Solution :

Each message features a 15-bit Cyclic Redundancy Checksum (CRC), and any node that detects a
different CRC in the message than what it has calculated itself will signal a CRC Error.

Quiz Time

What Is The Use Of Bit Stuffing?

Quiz Time

What Is The Use Of Bit Stuffing?

Solution:

Long NRZ messages cause problems in receivers

Clock drift means that if there are no edges, receivers lose track of bits
Periodic edges allow receiver to resynchronize to sender clock

Quiz Time

How to recover from CAN- Busoff?

Quiz Time

How to recover from CAN- Busoff?

Solution: The Bus Off state should only be left by a software reset. After software reset the CAN bus controller has to wait for 128 x 11 recessive bits to transmit a frame. This is because other nodes may pending transmission requests. It is recommended not to start an hardware reset because the wait time rule will not be followed then.

CAN

SYNCRONIZATION

Synchronization

Non-Return-to-Zero NRZ

➲ The CAN protocol uses Non-Return-to-Zero or NRZ bit coding. This means that the signal is constant for one whole bit time and only onetime segment is needed to represent one bit.

➲ Usually, but not always, a "zero" corresponds to a dominant bit, placingthe bus in the dominant state, and a "one" corresponds to a recessive bit, placing the bus in the recessive state.

Problems when using NRZ coding

- long sequences of bits of the same polarity
- no changes in voltage level for a longer time
- no falling edges for synchronization
- synchronization between sender and receiver may be lost

Synchronization

Bit stuffing

- Bit stuffing is used to ensure synchronization of all busnodes
- during the transmission of a message, a maximum of five consecutive bits may have the same polarity.
- the transmitter will insert one additional bit of the opposite polarity into the bit stream before transmitting further bits.
- The receiver also checks the number of bits with the same polarity and removes the stuff bits again from the bit stream. This is called "destuffing".

Bit stuffing ...

- after five consecutive bits of same polarity, insert one bit of reverse polarity
- CRC code is calculated before bit stuffing is done
- bit stuffing is done by the sender directly before transmission
- de-stuffing is done by the receiver directly after reception
- CRC code check is done after de-stuffing the frame
- bit stuffing is applied to part of the frame from SOF to CRC field

Bit stuffing...examples

Example 1

original sequence

stuffed sequence

Example 2

original sequence

stuffed sequence

Example 3

original sequence

stuffed sequence

Synchronization

Bus Synchronization

- CAN handles message transfers synchronously.
- All nodes are synchronized at the beginning of each message with the first falling edge of a frame which belongs to the Start Of Frame bit ➔ Hard Synchronization.

- To ensure correct sampling up to the last bit, the CAN nodes need to re-synchronize throughout the entire frame. This is done on each recessive to dominant edge.

Synchronization

Bit Construction

- 4 Segments, 8-25 Time Quanta (TQ) per bit time
- Time Quanta generated by programmable divide of Oscillator
- CAN Baud Rate (= 1 / Bit Time) programmed by selection of appropriate TQ length + appropriate number of TQ per bit

Bit time

The bit time t_{Bit} is the time it takes to transmit one bit.

Example:	**Data rate:**	**f**	**=**	**500 kBit / s**
➔	**Bit time:**	**t_{Bit}**	**=**	**2 µs**

Time quantum

The bit time is divided up into time quanta t_q.

Length of one time quantum: $t_q = BRP / f_{Sys}$

Allowed number of time quanta: $8 \leq n_q \leq 25$

BRP: Baud Rate Prescaler, f_{Sys}: System clock

Bit time segments

	Sync_Seg	Synchronization Segment
	Prop_Seg	Propagation Time Segment
	Phase_Seg1	Phase Buffer Segment 1
	Phase_Seg2	Phase Buffer Segment 2

Synchronization Segment

Edges in CAN bus level are expected to occur here.

Synchronization Segment has fixed length: $t_{Sync_Seg} = 1\ t_q$

Propagation Time Segment

The Propagation Time Segment compensates for physical delay times within the CAN network.

Length: $1\ t_q \leq t_{Prop_Seg} \leq 8\ t_q$

Sample point

Phase_Seg1 **Phase Buffer Segment 1**

Phase_Seg2 **Phase Buffer Segment 2**

- Phase Buffer Segment 1 may be lengthened during resynchronization.
- Phase Buffer Segment 2 may be shortened during resynchronization.

Sample point

The Phase Buffer Segments surround the sample point.

Lengths:

$$1\,t_q \le t_{Phase_Seg1} \le 8\,t_q$$
$$2\,t_q \le t_{Phase_Seg2} \le 8\,t_q$$

Adjustable controller parameters

TSEG1 Time Segment 1

TSEG1 = Prop_Seg + Phase_Seg1

TSEG2 Time Segment 2

TSEG2 = Phase_Seg2

BRP Baud Rate Prescaler

SJW Synchronization Jump Width

SJW ≤ min (Phase_Seg1, Phase_Seg2) **AND** **SJW ≤ 4**

SJW indicates the maximum number of time quanta t_q by which TSEG1 may be lengthened or TSEG2 may be shortened.

Sample point

The Phase Buffer Segments surround the sample point.

Lengths:

$$1\,t_q \le t_{Phase_Seg1} \le 8\,t_q$$
$$2\,t_q \le t_{Phase_Seg2} \le 8\,t_q$$

Synchronization

Bit lengthening

	Sync_Seg	Synchronization Segment
	Prop_Seg	Propagation Time Segment
	Phase_Seg1	Phase Buffer Segment 1
	Phase_Seg2	Phase Buffer Segment 2

Synchronization

Bit Shortening

	Sync_Seg	Synchronization Segment
	Prop_Seg	Propagation Time Segment
	Phase_Seg1	Phase Buffer Segment 1
	Phase_Seg2	Phase Buffer Segment 2

Sample point

The Phase Buffer Segments surround the sample point.

Lengths:

$$1\,t_q \le t_{Phase_Seg1} \le 8\,t_q$$
$$2\,t_q \le t_{Phase_Seg2} \le 8\,t_q$$

Parameter calculation (1)

1. Define bit rate f_{Bit}

   or $\boxed{\text{bit time} \quad t_{Bit} = 1 / f_{Bit}}$

2. Define number of time quanta n_q per bit time t_{Bit} . Remember that:

$$\boxed{8 \leq n_q \leq 25}$$

Parameter calculation (2)

3. Calculate length of time quantum t_q

$$t_q = t_{Bit} / n_q$$

4. Calculate value of Baud Rate Prescaler (BRP)

$$BRP = t_q * f_{Sys}$$

➔ First Bit Timing parameter: BRP

Parameter calculation (3)

5. **The length of the Synchronization Segment Sync_Seg is fixed:**

$$t_{Sync_Seg} = 1\ t_q$$

6. **To define the length of the Propagation Segment Prop_Seg, measure the delay times in the system.**

Rule of thumb: **The longer the CAN Bus, the longer the Propagation Segment.**

Parameter calculation (4)

7. **Define the length of the Propagation Segment Prop_Seg. Remember that:**

$$1\ t_q \leq t_{Prop_Seg} \leq 14\ t_q$$

Additionally, the following rule applies:

$$3\ t_q \leq t_{Bit} - t_{Sync_Seg} - t_{Prop_Seg} \leq 16\ t_q$$

Synchronization

	Sync_Seg	Synchronization Segment
	Prop_Seg	Propagation Time Segment
	Phase_Seg1	Phase Buffer Segment 1
	Phase_Seg2	Phase Buffer Segment 2

- Prop_Seg >= Delay A_to_B + Delay B_to_A
- Prop_Seg >= 2 • [max(node output delay+ bus line delay + node input delay)]
- Delay A_to_B >= node output delay(A) + bus line delay(A→B) + node input delay(B)

Parameter calculation (5)

8. If the remaining number of time quanta

$$\boxed{(t_{Bit} - t_{Sync_Seg} - t_{Prop_Seg}) / t_q}$$

is an odd number, choose:

$$t_{Phase_Seg2} = t_{Phase_Seg1} + 1\ t_q$$

else choose: $\boxed{t_{Phase_Seg2} = t_{Phase_Seg1}}$

Parameter calculation (6)

9. Calculate values of TSEG1 and TSEG2:

$$TSEG1 = (t_{Prop_Seg} + t_{Phase_Seg1}) / t_q$$

$$TSEG2 = t_{Phase_Seg2} / t_q$$

➔ Second Bit Timing Parameter: TSEG1

Third Bit Timing Parameter: TSEG2

Parameter calculation (7)

10. Choose the Synchronization Jump Width (SJW).
    Remember that:

$$SJW \leq \min(Phase_Seg1, Phase_Seg2)$$
$$SJW \leq 4$$

Rule of thumb: The larger the oscillator tolerance, the larger the Synchronization Jump Width.

➔ **Fourth Bit Timing Parameter: SJW**

Synchronization

Relation between Baud Rate and Bus Length

1MB → **40m**

50KB → **1000m**

Quiz Time

- Why CAN-BUS uses NRZ encoding instead of RZ?

Quiz Time

- Why CAN-BUS uses NRZ encoding instead of RZ?
- Solution:
  - NRZ encoding uses to change Controller digital signal into BUS level.

Quiz Time

- For synchronization of CAN
- Prop_Seg >= .....?

Quiz Time

- For synchronization of CAN
- Prop_Seg >= Delay A_to_B + Delay B_to_A?

Quiz Time

- What is Bit Stuffing?

Quiz Time

- What is Bit Stuffing?
- Solution:
- The idea behind bit stuffing is to provide a guaranteed edge on the signal so the receiver can resynchronize with the transmitter before minor clock discrepancies between the two nodes

CAN
Arbitration

Physical Transmission

- Transmission with Carrier Sense Multiple Access with Collision Avoidance CSMA/CA
- dominante Bits (low) win against recessive Bits (high) on the bus

Arbitration: Example

CAN Controller

CAN Controller

Structure of the CAN Protocol Controller

CAN Controller

Bit Stream Processor
- translates messages into frames and vice versa.
- inserts and extracts stuff bits
- calculates and checks the CRC code
- evaluate at the Sample Point and processes the sampled bus input bit.

Bit Timing Logic
- number of time quanta in the bit time..
- synchronization but occasional
- Bit Timing Logic (configured by TSEG1, TSEG2, and SJW) defines the number of time quanta in the bit time.

The Shift Register
- serializes the messages to be sent and parallelizes received messages
- Its loading and shifting is controlled by the BSP.

Baud Rate Prescaler
- defines the length of the time quantum,
- the basic time unit of the bit time; the

CAN

Physical Layer

Physical Layer

Transceiver PCA 82C250

V_{ref}	reference voltage output
TxD	transmit data input
RxD	receive data output
GND	ground

Rs	slope resistor input
CAN_H	HIGH level CAN voltage input / output
CAN_L	LOW level CAN voltage input / output
V_{cc}	supply voltage

CAN - Physical aspects

- Transmission rate up to 1 MBit/s needs bus driver circuits (transceiver) according to ISO/DIS 11898
- CAN High Speed Transmission (ISO/DIS 11898)
  - Transmission rate 125 Kbit/s up to 1 MBit/s
  - max. bus length depend on transmission rate (e.g. 1 MBit/s max. 40 m bus length)
  - up to 30 nodes

Physical Layer

Transceiver TJA1053/1054

EN	enable input signal
NSTB	**not** standby input signal
NWAKE	**not** wake input signal
TxD	transmit data input
RxD	receive data output
BAT	battery voltage
V_{CC}	supply voltage

NERR	error output pin
CAN_H	high voltage bus line
RTH	termination resistor
CAN_L	low voltage bus line
RTL	termination resistor
INH	inhibit output
GND	ground

CAN Low Speed Transmission (ISO / DIS 11519-2)

- Transmission rate 10 KBit/s up to 125 KBit/s
- max. bus length depend on the distributed capacity of the line
- up to 20 nodes

CAN and EMI

- Due to the differential nature of transmission CAN is insensitive to electromagnetic interference, because both bus lines are affected in the same way which leaves the differential signal unaffected.

- To reduce the sensitivity against electromagnetic interference even more, the bus lines can additionally be shielded. This also reduces the electromagnetic emission of the bus itself, especially at high baud rates.

Double-Wire Bus Line Termination

Connectors for CAN

The CAN connector is defined according to the specifications ISO/DIS 11898 and CiA/DS 102-1 (defines pinning). The optional Pins 6 and 9 may be used to feed a NiPC (Networked industrial Process Control) unit.

Pin	Signal	Description
1	-	Reserved
2	CAN_L	CAN_L bus line dominant low
3	CAN_GND	CAN Ground
4	-	Reserved
5	(CAN_SHLD)	Optional CAN Shield
6	(GND)	Optional Ground
7	CAN_H	CAN_H bus line dominant high
8	-	Reserved
9	(CAN_V+)	Optional CAN external pos. supply

male

female

CAN Cable Termination Resistor

- The CAN bus cable needs at the ends a termination resistor R_T to provide EMC characteristics without corrupting the DC characteristics:

- For basic termination at a serial cable use for termination resistor R_T = 124 Ω with linear CAN bus lines.

- For a split termination concept use for termination resistor $R_T/2$ = 62 Ω with a star architecture.

CAN Cable Termination Resistor cont.

➲ The multiple termination concept is used when network architecture differs from linear structure e.g. the cable drop length of a few meters is needed. For such a topology a multiple termination concept with three branches needs about three times the total termination resistance R_T = 180 Ω .

Daimler-Crysler Patented Star Topology

- Daimler-Crysler have patented a star topology with a common termination. This topology avoids resonance problems, undesired harmonies and intensive to incoming noise. All noise traveling towards the termination point will be absorbed there and never reflected.

Quiz

- **What is voltage Levels in CAN Protocol?**

Quiz

- **What is voltage Levels in CAN Protocol?**
- **Solution:**
  - CANH voltage level is 3.3 Volt
  - CANL Voltage Level is 1.5 Volt

Quiz

- **Why CAN Bus uses differential Cable?**
- **Solution:**

Quiz

- **Why CAN Bus uses differential Cable?**
- **Solution:**
- CAN Protocol used two-wire twisted-pair cables for CAN-Bus architecture to reduce cross-talk and EMI

Quiz

- **Why Can Is Having 120 Ohms At Each End?**

Quiz

- **Why Can Is Having 120 Ohms At Each End?**

Solution:

- To minimize the reflection reference, to reduce noise. To ensure that reflection does not cause communication failure, the transmission line must be terminated.

Quiz

- **What is Can Arbitration?**
- **Solution:**

Quiz

- **What is Can Arbitration?**
- **Solution:**
- CAN Arbitration is nothing but the node trying to take control of the CAN bus

Thank You !

L&T Technology Services