CAN Protocol

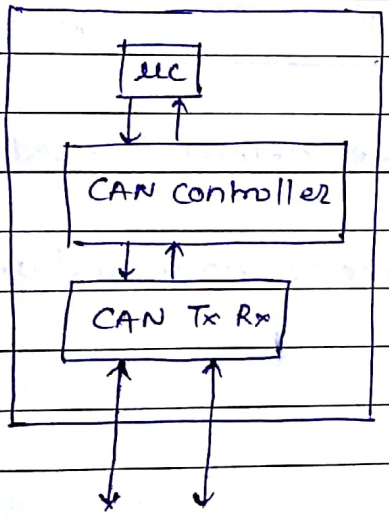
Date: / /

Controller Area Network - designed by Robert Bosch.

About CAN protocol -

- CAN was 1st created for automotive use, so its common application is "In-Vehicle Electronic Networking".
- CAN bus acts as Central Networking System of the CAR.
- CAN protocol is two wire protocol.
- CAN is a multimaster serial bus standard in Automotive Industry for connecting Electonics control units. (ECU's) also known as Nodes.
- Every ECU contains 3 IC's - uC, CAN controller, CAN transreceiver

ECU/ Node



- CAN protocol is designed in such way that it will never face data integrity problem.

data integrity - very weak or strong signal causes cumpt the data.

(If we use the UART protocol inside engine. due to engine sound it will face data integrety problem.

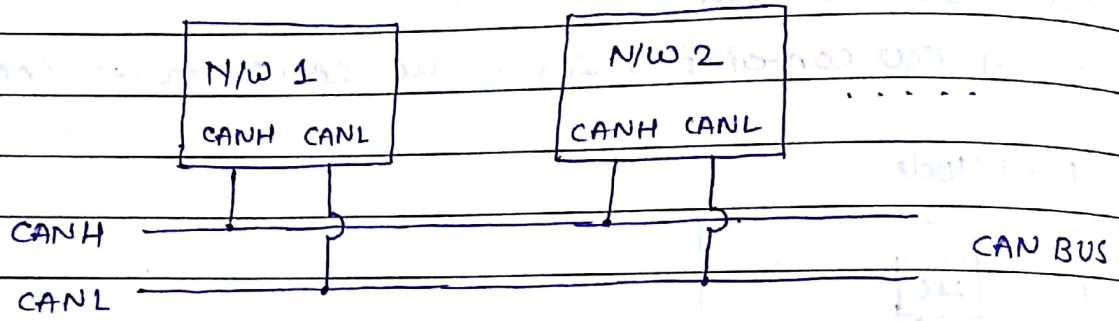
- CAN protocol is also used for On-Board Diagnostic system (OBD system).

Date: / /

- purpoes to replace the complex wiring harness with two-wire-BUS.

features of CAN protocol:

1. The CAN Bus was developed by Robert Bosch
2. Two wire Bus
- CANH (CAN High voltage)
- CANL (CAN LOW voltage)



3. IN CAN protocol Differential Voltage method is used to transmit '1' data bit.
(Differential voltage is less sensitive to noise and any external Interference.)

4. CAN bus is:
- Broadcast
- Multi master
- Half duplex
- Asynchronous Serial communication.

5. CAN Bus speed and distance-
- 1Mbps at N/w length below 40 meters
- Decreasing the bit rate below, longer network distance
Eg. 500 meters at 125 Kbps (Kilobit per second)

Cable length vs Signalling Rate

Bus length (m)	signalling rate (Mbps)
40	1
100	0.5

Date: / /

Buslength	Signalling Rate (mbps)
200	0.25
500	0.10
1000	0.05.

6 Max. Nodes on CAN Bus:

- practically upto 30 Nodes.

(When more than the 30 Nodes are used on a bus it is recommended that a transreceiver with a high bus input impedence.)

Bus Impedance = ?

To increase the I/P bus impedance increase the current and decrease the voltage. ⟹ how?

7. CAN bus is available in two versions

- Standard CAN (Version 2.0A)

11 bit Node identifier

- Extended CAN (Version 2.0B)

29 bit node identifier.

- have difference in msg frame & no. of nodes.

these identifiers are user defined. we cannot assign two same identifiers to different nodes.

MAX. bytes that can be transmitted by node to another node is 8 bytes.

CAN protocol is not designed to send the massive data.

Using CAN FD we can transfer the data more than 8 bytes.

8. To achive Design transparency and implementation flexibility CAN has been subdevided into different layers

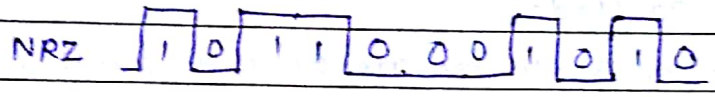
- Application layer (etc)
- Data Link layer
- physical layer

} CAN protocol works.

CAN FD → flexible Data rate (upto 64 bytes)

8. CAN Data Format -

- The message is transmitted serially onto the bus using a non-return-to-zero (NRZ) format
- As the name suggest, there is no transition between same polarity bit.



9. CAN standard supports several topologies. commonly used topologies are:

- line / Bus Topology
- Star topology
- Hybrid Topology

10. Byte order -

MSB to LSB.

Date: / /

CAN Applications:

- CAR, trucks, buses, Industrial Vehicles, ships, planes, Industrial Automation.
- Marine
- Military
- Medical
- Agricultural, etc..

CAN in Automotive Industry - Applications.

- power window / Electric window & mirrors.
- CAR lightning system
- Engine control unit
- Airbag system
- Anti-Lock Breaking system.
- Automatic climat control system
- Keyless Entry system
- power steering.

etc---

Bus setup -

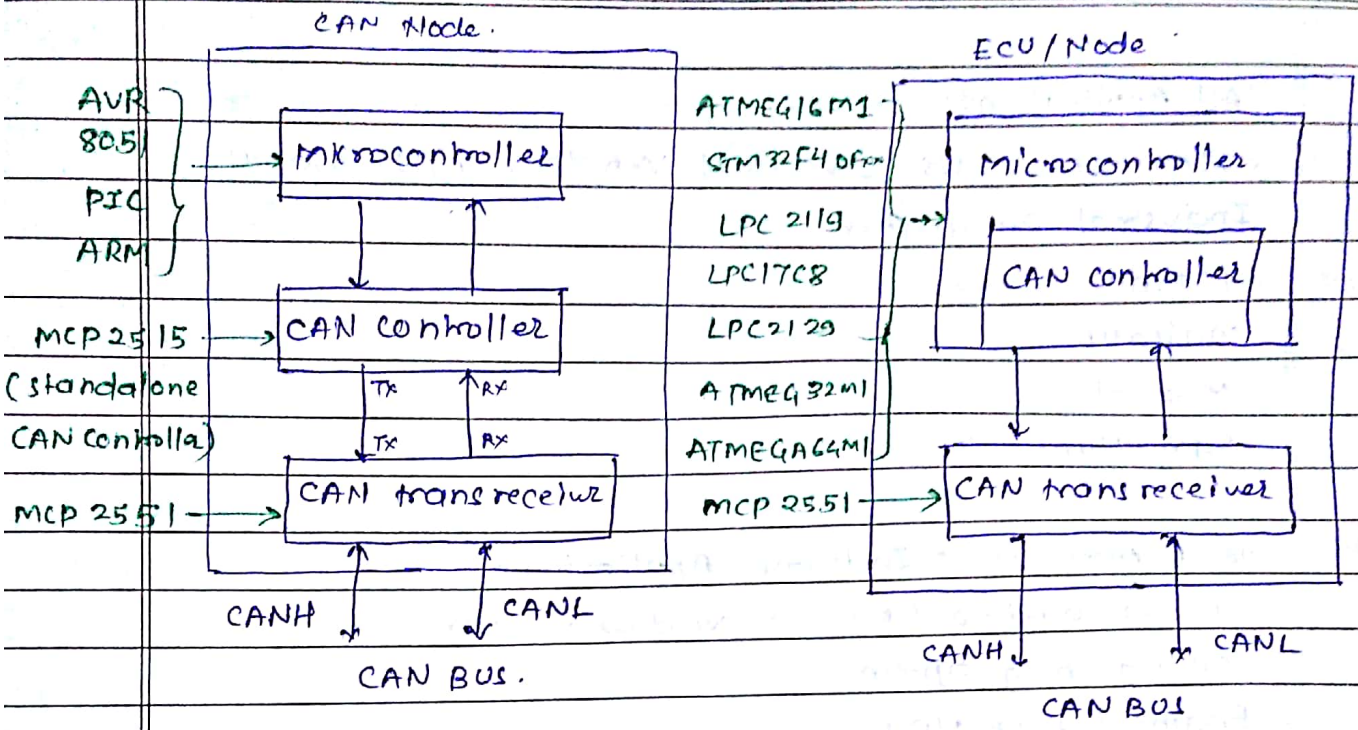
About ECU / Node

minumum requirment of the ECU / Node -

1. Microcontroller
2. CAN controller
3. CAN transreceiver

Extra Components can be available in ECU -

- sensors
- switches / Buttons.
- Bulbs / LED's, motors,



No protocol used betn CAN controller and CAN transreceiver.

- MCP2515 have total 128 Registers.
- CAN controller MCP2515 std. number is ISO 11898-1 and CAN transreceiver have standerd no. ISO 11898-2,3

Role of CAN Controller and CAN transreceiver

① **CAN controller -**

1. Create message frame.
2. Transmit and receive msg frame
3. Arbitration
4. Error Handling.
5. Synchronization.

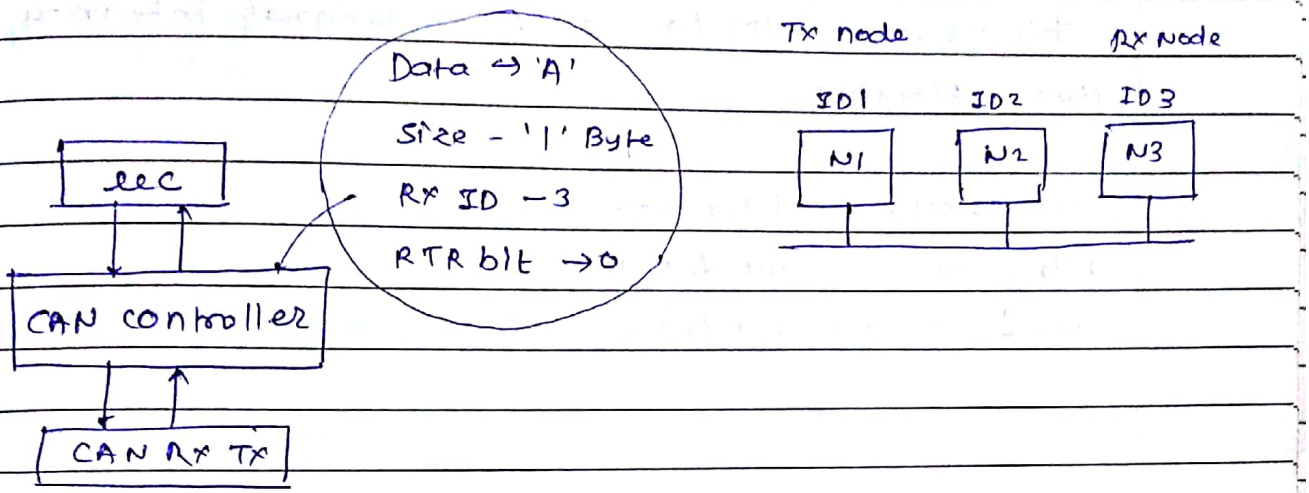
② **CAN transreceiver -**

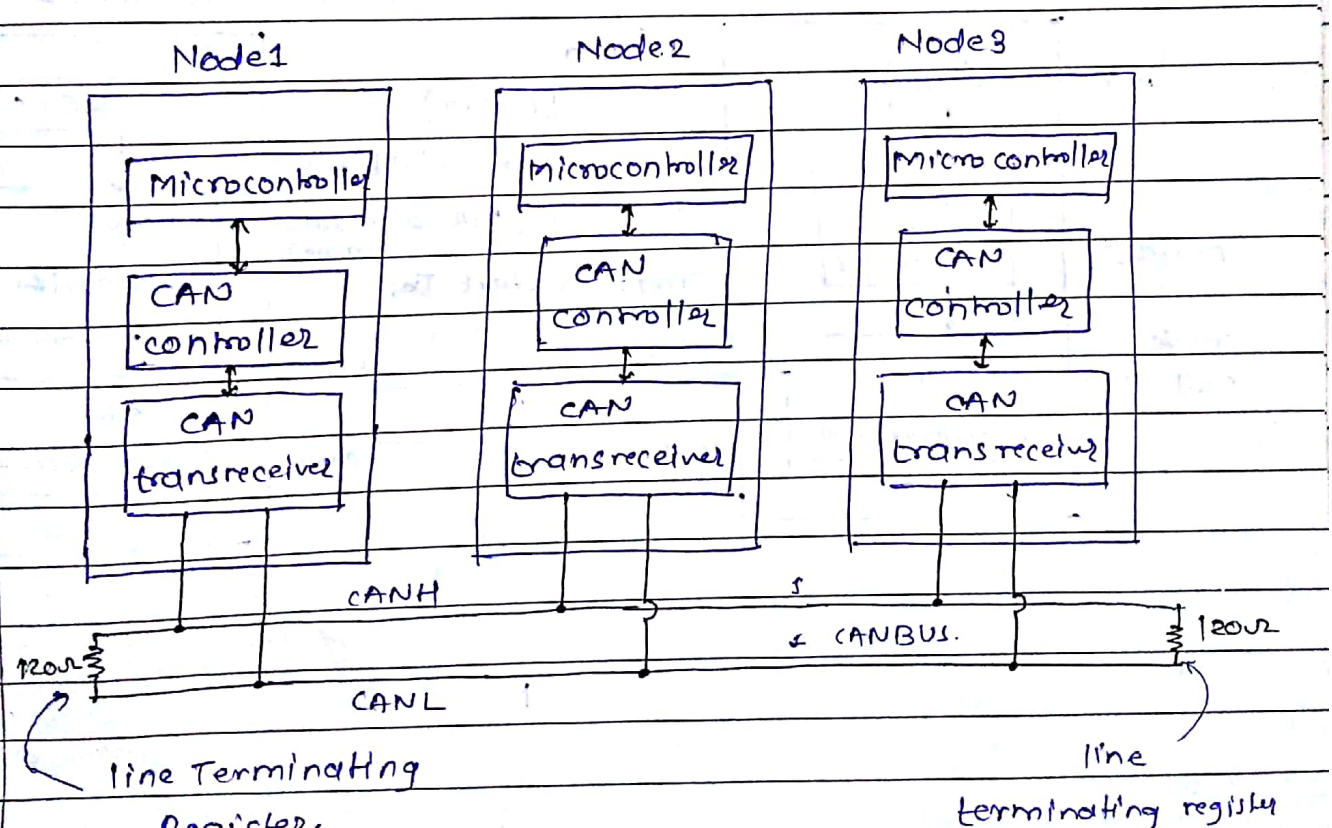
1. Convert Each bit into a CAN voltage level (Voltage conversion)

TTL voltage level ⟷ CAN voltage level & vice versa.

Date: / /

CAN Controller work in Data Link layer and CAN transreceiver work in physical layer.





— "prevent signal Reflection"

Idle state of CAN BUS - 1 1

120Ω value is recommended by bosch.

Date: / /

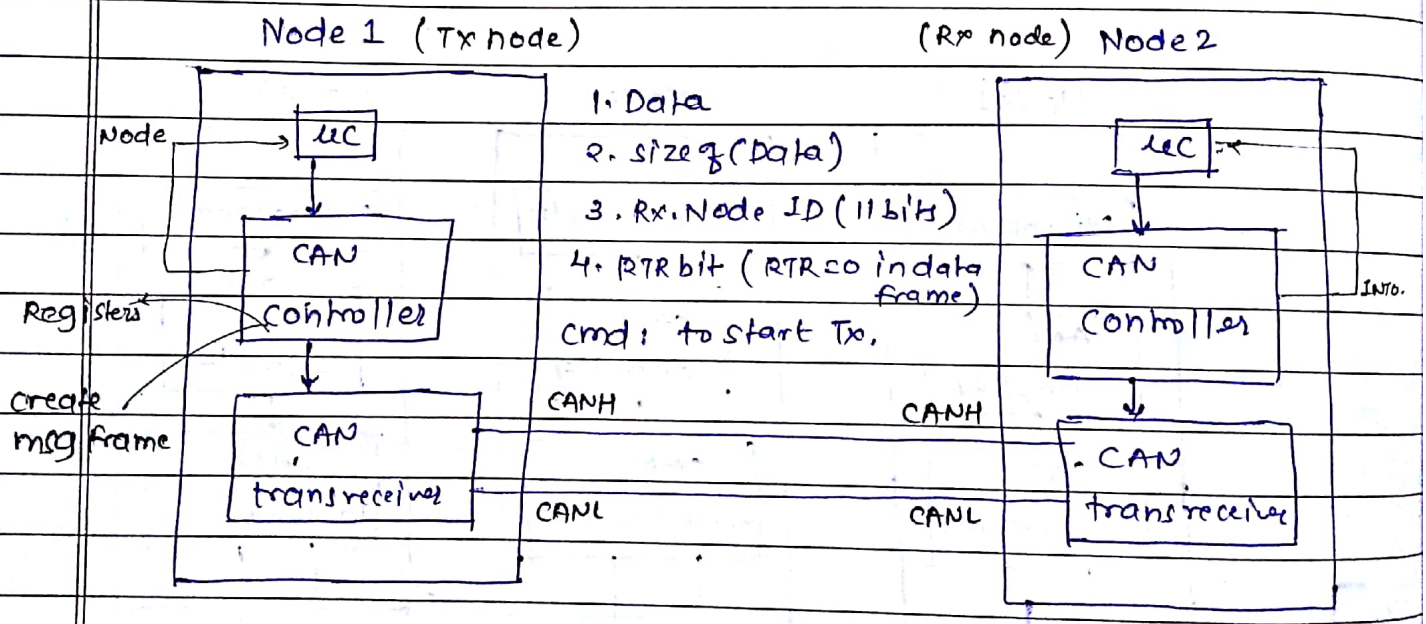
line terminating Register.

- The CAN Bus should be properly terminated at both ends with resistors that match the impedence of the bus
- This resistors helps to reduce the siagnal integrity issues like reflection.

Terminology used for bits -

bits 0 → Dominant Bit

bit 1 → Ressessive Bit.



CAN Voltage levels.

bit 0 (Dominant Bit)

Vdiff > 0.9 V

bit 1 (Recessive Bit)

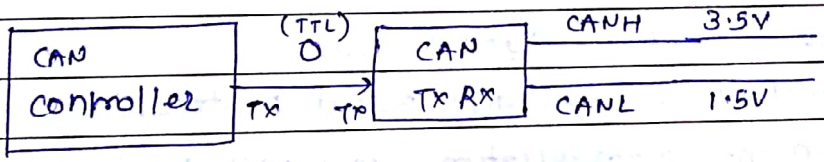
Vdiff < 0.5 V

Vdiff = CANH − CANL

TTL Voltage levels :

bit 0 : 0V to 0.8 V

bit 1 : 2V to 5.5 V



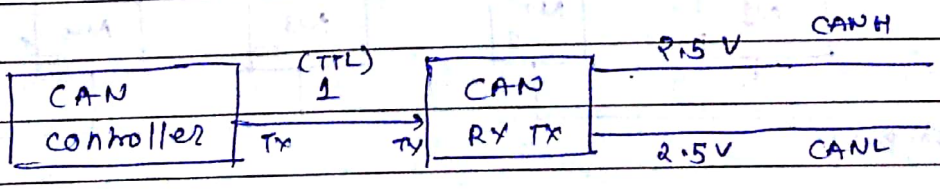
Start of frame — End of frame

0 | 0 | 0 | ...

* for Dominant bit

Vdiff = 3.5 V − 1.5 V = 2 V

= 2 V > 0.9

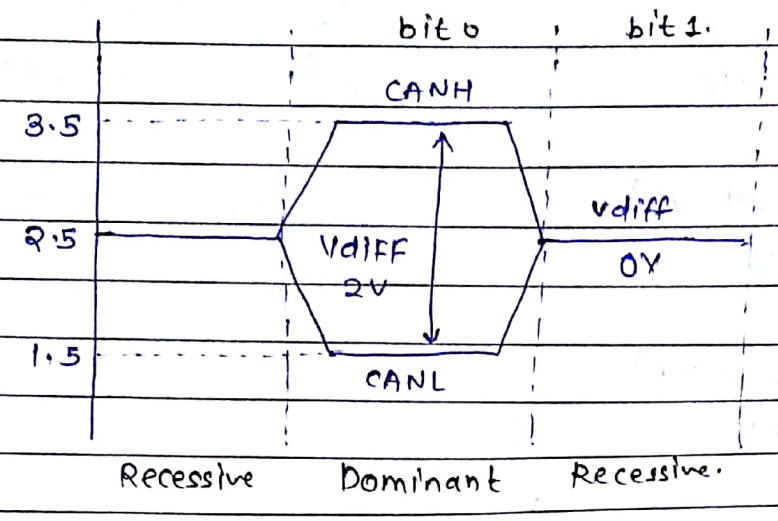


* for Recessive bit

Vdiff = 2.5 V − 2.5 V

= 0 V ✗ 0.5 V

Date: / /



testing tools for CAN

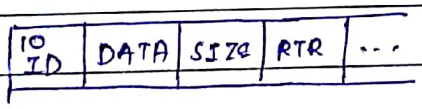
① CANOe

② CAN Analyser.

Q How CAN protocol works ?

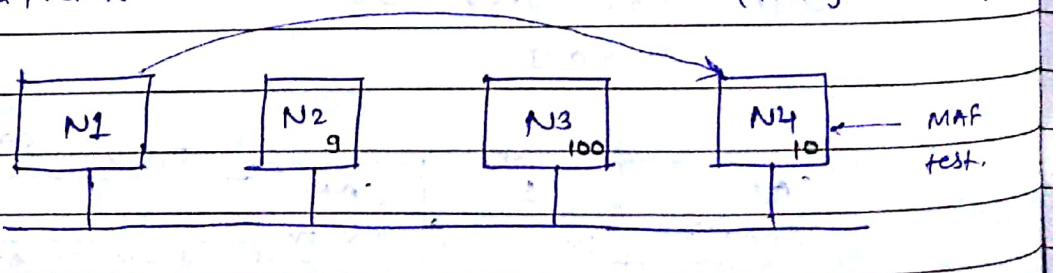
- CAN bus is Broadcast type of bus
- CAN protocol is a "message Based protocol".

It means in the Communication, messages are having priorities instead of nodes.



Message frame or Data frame

Message Acceptance filter (MAF) (In register setting)



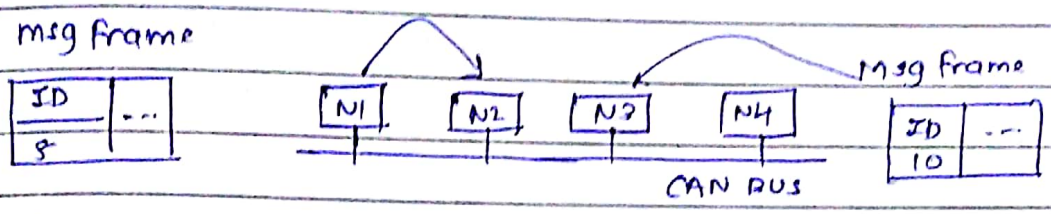
In Broadcast message MAF setting is not required

But otherwise we have to do MAF setting in each & every node.

Date: / /

Identifier and message priority :

- The lower the numerical value of the identifier, the higher the priority of msg.



10th bit 0th

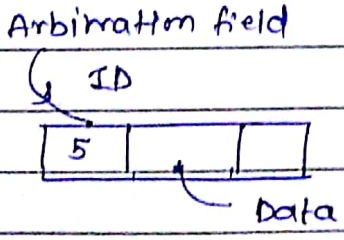
00 ..... 0101 - 5 ✓

00 ..... 1010 - 10

0 & 1 = 0 - wired logic

CAN Protocol Arbitration -

- In CAN protocol arbitration, bit 0 is a dominant bit and 1 is recessive bit
- The Node who writes first 0 that node will get bus control & message frame will get transmitted
- Node who lost arbitration, will wait till the bus become free (idle) & then transmits msg frame from beginning.



During arbitration, each transmitting node monitors the bus state and compares the received bit with transmitted bit. If a dominant bit is received when a recessive bit is transmitted then the node stops transmitting. That means that node lost arbitration. And that node can transmit a frame once the bus become free.

papergrid

Date: / /

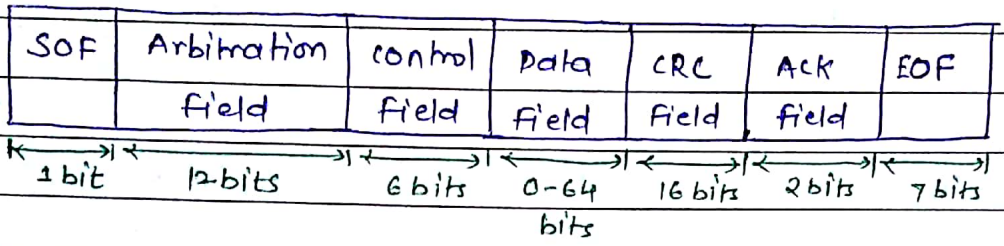
papergrid

Date: / /

Date: / /

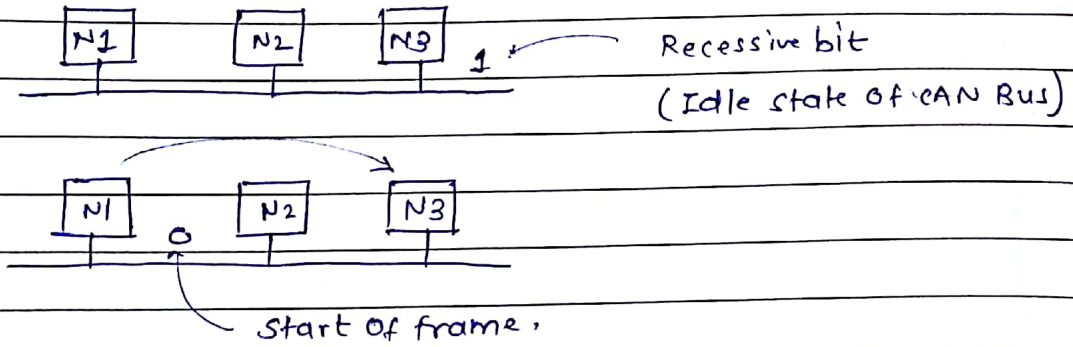
Standard Data frame.

- V2.0A Data frame:



- SOF (start of frame) :

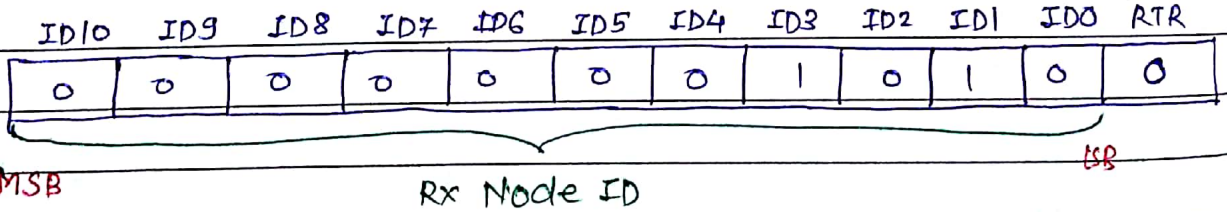
- It consist of a single dominant bit.

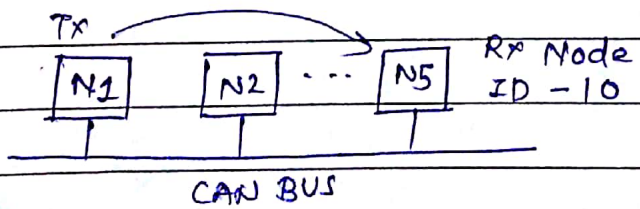


- Arbitration field :

- It consist of Rx Node Identifier and RTR bit

RTR : Remote Transmission Request





RTR	frame
0	Data frame
1	Remote frame.

Date: / /

Imp Node -

Theoritically, Rx Node ID

$2^{11} = 2048$

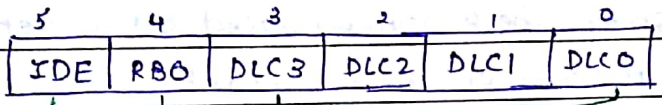
i.e. In the range of 0-2047.

practically,

- The 7 MSB bits (ID10 - ID4) must not be all recessive (all 1)

* Two nodes cannot transmit a frame using same Rx Node ID.

- Control field (6 bits)



Identifier Extension — reserved bit 0 — Data Length Code (0-8 bytes)

IDE	frame
0	Std. data frame
1	Extended Data frame.

RB0 → 0

DLC3	DLC2	DLC1	DLC0	size.
0	0	0	0	0 bytes
0	0	0	1	1 byte
d	d	r	d	2 bytes
d	d	r	r	3 bytes
:				
r	d	d	d	8 bytes.

- Size of data field is transmitted using DLC.

Date: / /

• Data field (0-64 bits / 0-8 bytes)

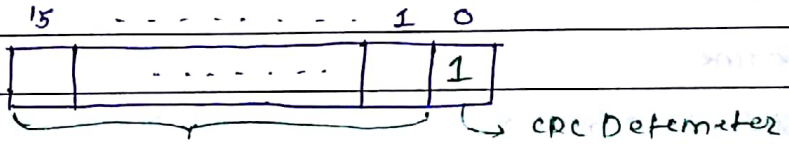
- It consist of the data to be transfered within a data frame
- It can contain data from 0-8 bytes which each contains 8 bits which are transffered MSB first.

• CRC field (Cyclic Redundancy check) (16 bit):

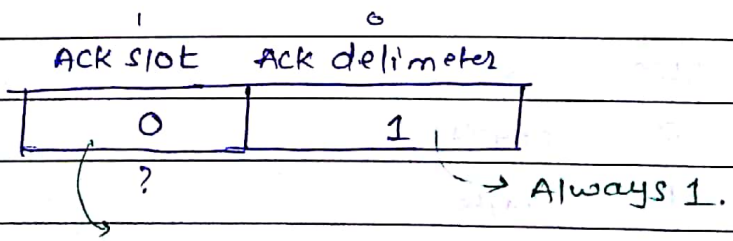
CRC field follows the data field & is used to detect transmission errors

- CRC is also called as Error-detecting-Code.
- CRC check is done by can controller only CAN RXTX only performs voltage conversions.
- CRC is a short check value, based on a remainder of polynomial division of their contents.
- CRC value is 15 bit & 16th bit is Delimeter.

CRC Delimeter - Always '1':



• Acknoledgement field (2bit) :



• EOF (End of Frame) (7 bits)

- Each data frame & remote frame is ended by a flag sequence consisting of 7 recessive bits. (7 times 1)

Date: / /

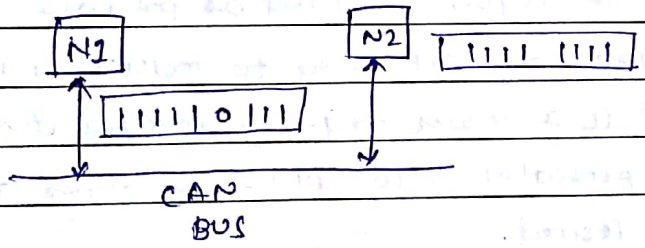
- Transmitter node will write EOF to free the bus (idle the bus)
- This case will not happen in Data field if there is '7' 1's in the Data field then controller add '0' (opposite polarity bit) called as bit stuffing.

imp Q What is Bit stuffing

To ensure enough transitions to maintain synchronization, a bit of opposite polarity is inserted after five consecutive bit of the same polarity

- If 7 1's are written consecutively. It will considerd as EOF.

If the 5 1's are written then can we add one '0' (i.e. opposite polarity bit) to that Data)



Bit stuffing will not affect the EOF it is done in the other field.

- Bit stuffing can't be affected on following fields:
  - CRC Delimiter
  - Ack field
  - EOF.
- All above field sizes are fixed & are not stuffed.

Can we will take care of bit stuffing. in program no need to take ca

Date: / /

Read :

Nominal Bit Time

Nominal Bit Rate

Time quantum

Type of Errors in can protocol.

- Time quantum -

Time quantum is a fixed unit of time derived from the oscillator period.

CAN FD (FD - Flexible data rate)

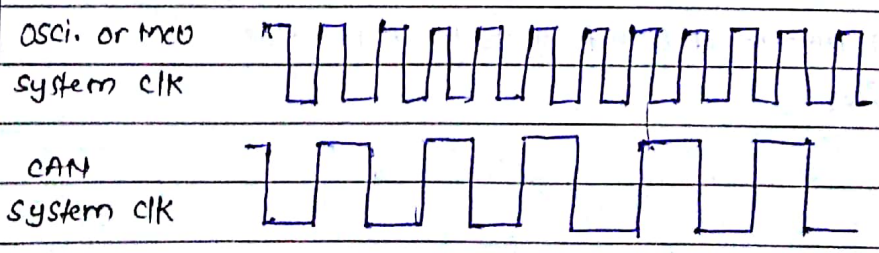
Speed - 12mbps (max)

Data field - 64 bytes.

Difference bet^n CAN & CAN FD ⇒ ?

The ckt which divide the frequency called as prescalar.

- A prescalar is an electronic ckt used to reduce a high freq. electrical signal to a lower freq. by integer division.
- The purpose of the prescalar is to allow the timer to clocked at the rate a user desired.
- In CAN controller it is also called as a baudrate prescalar.
- BRP is used to divide the microcontrollers main internal clock into a suitable clock rate.



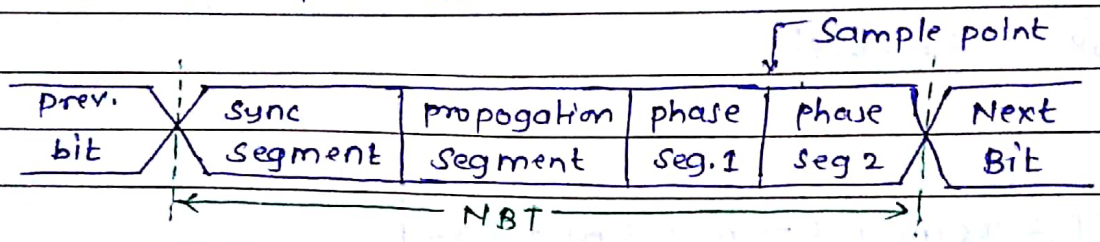
CAN bit period.

Date: / /

Data integrity check is done in CAN ucc.

- Nominal Bit Time

It is time taken to transmit a single bit whether it may be recessive / dominant bit.



When SOF is written on CAN Bus then only other nodes will generate clock pulses.

- Sync. segment
  - The duration of sync. segment is (Always.) 1 TQ.
  - used to synchronise clock pulses of all nodes.

  All nodes sit in idle state after sync. (ie. for other nodes. upto propogation seg.) & the will read the next data simultenously.

- propogation seg - (Node will not read data line)
  - The propo. seg. is need to compensate for the delay in bus lines.
  - propogation seg. duraton 1TQ to 8 TQ.

- phase1 & phase 2
  - Actual bit sampling takes place in this segment.
  - phase seg1 duration : 1TQ to 8 TQ
  - phase seg2 duration : 1TQ to 8 TQ.

* Through program we can tell to ucc (CAN) to take multiple samples but before phase seg. 2. (min. 1 to max3 sample points)
  (Node will wait uptill phase seg1 & read value after that).

NBT = 8 µs (In program)

$NBR = \frac{1}{NBT}$ $\left[F = \frac{1}{T} \right]$

$\therefore NBR = \frac{1}{8\ \mu s}$

NBR = 125 kbps

NBT = $t_{sync} + t_{prop.} + t_{ps1} + t_{ps2}$

t_{sync} = 1 TQ

$t_{prop.}$ = 2TQ

t_{ps1} = 7 TQ

t_{ps2} = 5TQ.

Synchronization Jump width (SJW) = 1 TQ [1-4 TQ]

Total TQs - 1+2+7+5+1 = 16 TQ.

16 TQ (to achive 8 µs)