CAN

What is CAN?

* The controller area network (CAN Bus) is a message based protocol to allow ECU to communicate.
* Each ECU is called Node.
* Communication could be started by any node.
* The CAN communication is asynchronous.
* The communication medium where this communication is happening or where the data is transmitted is called as Bus.
* It is highly immune to noise as it is dual twisted pair.

Why CAN?

* Automotive domain needed a way for all ECU’s to communicate the data among themselves.
* Communication could be started by any node.
* Highly immune to electrical noise.
* Short latency period.

Why CAN as Dual twisted Pair?

* For Noise Cancelation.
* Two similar wires placed closely in an electrical circuit generate equal and opposite magnetic fields. The two magnetic fields cancel or nullify each other as well as external magnetic fields.
* Thus when two wires are twisted, it enhances this cancellation effect and provides self shielding for the pair of wires.

Why CAN is having 120 ohms at each end?

* To minimize the reflection reference, to reduce noise. To ensure that reflection does not cause communication failure, the transmission line must be terminated with 120 Ohms.

CAN Features:

* CAN Speed
* CAN Data
* CAN Bus
* CAN Logic
* CAN Controller and Transceivers
* CAN Network Topology

CAN Speed:

* CAN Should transfer data in form of frames.
* CAN frames can go upto 1mbps speed with the bus length increases above the maximum speed decreases.
* As it is serial bus we transfer bits per second.

CAN Data:

* The Bus contains frame:
* The frame has 2 parts: PCI & Data
* PCI – Protocol Controlled Information
* Maximum Data Sent per frame is 8 bytes.

CAN Bus:

* CAN Bus contains two pairs/wires which are twisted called as twisted pair.
* Contains CAN High and CAN Low.
* Terminated on both ends of wire with 120 ohm resistor.
* Logic 1 = Recessive bit.
* Logic 0 = Dominant bit.

CAN Controller and Transceiver:

* CAN transceiver is used to convert TTL for microcontrollers and vice versa.

CAN Network Topology:

* It contains Bus topology.
* If one ECU Contains baud rate 200kbps, then all ECU have 200kbps.

CAN Properties:

* Wired
* Serial
* Asynchronous
* Broadcast (Multicast)
* Message Based Protocol
* Message Filtering
* Half Duplex
* Acknowledge Method
* CSMA/CA Protocol
* Temporary and Permanent Node Failures.

CAN Frames:

* Data Frame
* Remote Frame
* Error Frame
* Overload Frame

Data Frame:

* Data Frame is a Frame Containing node data for transmission.
* Carries data from transmitter node to all the receiver node.
* Two types of data frame
* Standard Frame – 11 Bit msg ID -211 of msgs.
* Extended Frame – 29 Bit msg ID –(base id + 18 bit extension)

Remote Frame:

* It is a frame requesting the transmission of a specific identifier.
* Remote frame requests for data frame.
* A node wants a particular data frame, so it sends the remote frame with same message id on the bus.
* This remote frame is followed by the data frame on the bus transmitted by the node who owns that data frame.
* Hence remote frame sending node got the data it requested for.

Error Frame:

* Error frame is a frame transmitted by any node detecting an errors.
* Error frame signals an error condition in the data frame being transmitted currently.
* If a node detects an error in the data frame being transmitted on the bus currently, then it destroys that data frame and signals all other nodes by transmitting a error frame.
* The error can be detected by any of the receiver node or the transmitter node.

Overload Frame:

* Overload frame is transmitted by a node when it is overloaded and needs some time to process the data frame received previously.
* It is a frame to inject a delay between data or remote frame.
* Overload frame format is same as that of an error frame.
* Strategy is that since the overloaded node needs some time, so it buys that time by keeping the bus busy with its overload frame preventing other nodes to start transmitting data frames.
* Dummy frame is sent to bus, so that other frames can’t transmit to bus.
* The dummy frame transmission is known as overload frame.
* The maximum of 3 consecutive overload frames can be transmitted by each node after a data frame.

Explain Can\_H and Can\_L:

* Can\_H and Can\_L have different voltage levels that are interpreted by each controller.
* Can\_H usually measures from 2.5v to 3.75v and Can\_L measures from 2.5v to 1.25v
* When both lines read 2.5v the signal is called recessive i.e., 1
* When Can\_h -> 3.75v and Can\_L -> 1.25v the signal is called dominant i.e., 0

Extended Frame Format:

* CAN is a message based protocol.
* We know with standard frame format the msg ID being 11 bits we have maximum of 512 frames per network.
* In real time scenario, this number is not always sufficient.
* So a extended frame format was proposed where the message ID is 29 bit long.
* In a network both standard and Extended CAN frames can co-exist.
* These frames are differentiated by a new bit called as IDE.
* SRR -> substitute remote request.
* RTR should co-inside with SRR.
* IDE -> will tell the controller whether it is standard frame or Extended Frame.
* IDE -> 1 -> Extended frame
* IDE -> 0 -> Standard frame
* R1 -> Remote
* R0 -> Data
* 11 bit ID -> Base ID
* 18 bit ID -> Extended ID
* SRR -> In place of RTR bit in standard frame format, always recessive and single bit.
* IDE -> Identifier Extension Indicator Bit to identify standard or Extended frame.

Diff between standard CAN and Extended CAN

* Standard frame format supports a length of 11 bits, we can have maximum of 512 frames per network.
* Extended frame format supports a length of 29 bits.
* In a network both standard and Extended CAN frames can co-exist.
* IDE can tell difference between standard and Extended frame.

Bit Stuffing:

* Bit stuffing occurs after 5 consecutive bits of the same value.
* If the bits are all 1’s or 0’s an additional bit of the opposite value is inserted into the data stream.
* The purpose of bit stuffing is to maintain synchronization between different nodes on the CAN bus.
* The CAN bus protocol relies on edge transitions to synchronize the internal clocks of the different node.
* By adding bit stuffing after every 5 consecutive bits of the same value, it ensures that there are enough edge transitions for the nodes to stay synchronized.

Bus arbitration:

* If the multiple nodes wants to transmit frame simultaneously on the bus, the conflict is resolved by as BUS arbitration.
* Node that transmits message with lowest ID (Highest priority) wins arbitration continues to transmit.
* SOF, Msg ID, RTR participate in bus arbitration.
* Nodes that lost arbitration will start new arbitration as soon as bus is free for access again

Types of CAN Error:

Bit Error:

* When the transmitter node monitors a different bit value than the value it transmitted in that bit, it is called as a bit error.
* Bp = Bm
* Bit(puts) = Bit(monitor)
* If it is not equal -> bit error

Acknowledgement Error:

* When the transmitter node monitors a recessive bit value in Ack slot, then it means it did not get a Ack. This is Ack Error.

CRC Error:

* When the receiver node has not given ack but still monitors a dominant bit in ack slot, then it is a CRC Error.

Stuff Error:

* When a receiver at least 6 consecutive bits of same polarity received, then it means the bit stuffing rule is violated. This is stuff error.

Form Error:

* When the fixed form of the CAN data frame (CD + AD + EOF) is altered then form Error occurs.