next â†’ â†• prev CAN (Controller Area Network) protocol CAN stands for Controller Area Network protocol. It is a protocol that was developed by Robert Bosch in around 1986. The CAN protocol is a standard designed to allow the microcontroller and other devices to communicate with each other without any host computer. The feature that makes the CAN protocol unique among other communication protocols is the broadcast type of bus. Here, broadcast means that the information is transmitted to all the nodes. The node can be a sensor, microcontroller, or a gateway that allows the computer to communicate over the network through the USB cable or ethernet port. The CAN is a message-based protocol, which means that message carries the message identifier, and based on the identifier, priority is decided. There is no need for node identification in the CAN network, so it becomes very easy to insert or delete it from the network. It is a serial half-duplex and asynchronous type of communication protocol. The CAN is a two-wired communication protocol as the CAN network is connected through the two-wired bus. The wires are twisted pair having 120â„¦ characteristics impedance connected at each end. Initially, it was mainly designed for communication within the vehicles, but it is now used in many other contexts. Like UDS, and KWP 2000, CAN also be used for the on-board diagnostics. Why CAN? The need for a centralized standard communication protocol came because of the increase in the number of electronic devices. For example, there can be more than 7 TCU for various subsystems such as dashboard, transmission control, engine control unit, and many more in a modern vehicle. If all the nodes are connected one-to-one, then the speed of the communication would be very high, but the complexity and cost of the wires would be very high. In the above example, a single dashboard requires 8 connectors, so to overcome this issue, CAN was introduced as a centralized solution that requires two wires, i.e., CAN high and CAN low. The solution of using CAN protocol is quite efficient due to its message prioritization, and flexible as a node can be inserted or removed without affecting the network. Applications of CAN protocol Initially, CAN protocol was designed to target the communication issue that occurs within the vehicles. But later on, due to the features it offers, it is used in various other fields. The following are the applications of CAN protocol: Automotive (passenger vehicles, trucks, buses)

Electronic equipment for aviation and navigation

Industrial automation and mechanical control

Elevator and escalators

Building automation

Medical instruments and equipment

Marine, medical, industrial, medical CAN layered architecture As we know that the OSI model partitions the communication system into 7 different layers. But the CAN layered architecture consists of two layers, i.e., data-link layer and physical layer. Let's understand both the layers. Data-link layer: This layer is responsible for node to node data transfer. It allows you to establish and

terminate the connection. It is also responsible for detecting and correcting the errors that may occur at the physical layer. Data-link layer is subdivided into two sub-layers: MAC: MAC stands for Media Access Control. It defines how devices in a network gain access to the medium. It provides Encapsulation and Decapsulation of data, Error detection, and signaling. LLC: LLC stands for Logical link control. It is responsible for frame acceptance filtering, overload notification, and recovery management.

Physical layer: The physical layer is responsible for the transmission of raw data. It defines the specifications for the parameters such as voltage level, timing, data rates, and connector. CAN specifications define CAN protocol and CAN physical layer, which are defined in the CAN standard ISO 11898. ISO 11898 has three parts: ISO 11898-1: This part contains the specification of the Data-link layer and physical signal link.

ISO 11898-2: This part comes under CAN physical layer for high speed CAN. The high- speed CAN allows data rate upto 1 Mbps used in the power train and the charges area of the vehicle.

ISO 11898-3: This part also comes under CAN physical layer for low-speed CAN. It allows data rate upto 125 kbps, and the low speed CAN is used where the speed of communication is not a critical factor. CiA DS-102: The full form of CiA is CAN in Automation, which defines the specifications for the CAN connector. As far as the implementation is concerned, the CAN controller and CAN transceiver are implemented in the software with the help of the application, operating system, and network management functions. CAN Framing Let's understand the structure of the CAN frame. SOF: SOF stands for the start of frame, which indicates that the new frame is entered in a network. It is of 1 bit.

SOF stands for the start of frame, which indicates that the new frame is entered in a network. It is of 1 bit. Identifier: A standard data format defined under the CAN 2.0 A specification uses an 11-bit message identifier for arbitration. Basically, this message identifier sets the priority of the data frame.

A standard data format defined under the CAN 2.0 A specification uses an 11-bit message identifier for arbitration. Basically, this message identifier sets the priority of the data frame. RTR: RTR stands for Remote Transmission Request, which defines the frame type, whether it is a data frame or a remote frame. It is of 1-bit.

RTR stands for Remote Transmission Request, which defines the frame type, whether it is a data frame or a remote frame. It is of 1-bit. Control field: It has user-defined functions. IDE: An IDE bit in a control field stands for identifier extension. A dominant IDE bit defines the 11-bit standard identifier, whereas recessive IDE bit defines the 29-bit extended identifier. DLC: DLC stands for Data Length Code, which defines the data length in a data field. It is of 4 bits. Data field: The data field can contain upto 8 bytes.

It has user-defined functions. CRC field: The data frame also contains a cyclic redundancy check field of 15 bit, which is used to detect the corruption if it occurs during the transmission time. The sender will compute the CRC before sending the data frame, and the receiver also computes the CRC and then compares the computed CRC with the CRC received from the sender. If the CRC does not match, then the receiver will generate the error.

The data frame also contains a cyclic redundancy check field of 15 bit, which is used to detect the corruption if it occurs during the transmission time. The sender will compute the CRC before sending the data frame, and the receiver also computes the CRC and then compares the computed CRC with the CRC received from the sender. If the CRC does not match, then the receiver will generate the error. ACK field: This is the receiver's acknowledgment. In other protocols, a separate packet for an acknowledgment is sent after receiving all the packets, but in case of CAN protocol, no separate packet is sent for an acknowledgment.

This is the receiver's acknowledgment. In other protocols, a separate packet for an acknowledgment is sent after receiving all the packets, but in case of CAN protocol, no separate packet is sent for an acknowledgment. EOF: EOF stands for end of frame. It contains 7 consecutive recessive bits known End of frame. Now we will see how data is transmitted through the CAN network. A CAN network consists of multiple of CAN nodes. In the above case, we have considered three CAN nodes, and named them as node A, node B, and node C. CAN node consists of three elements which are given below: Host

A host is a microcontroller or microprocessor which is running some application to do a specific job. A host decides what the received message means and what message it should send next.

A host is a microcontroller or microprocessor which is running some application to do a specific job. A host decides what the received message means and what message it should send next. CAN Controller

CAN controller deals with the communication functions described by the CAN protocol. It also triggers the transmission, or the reception of the CAN messages.

CAN controller deals with the communication functions described by the CAN protocol. It also triggers the transmission, or the reception of the CAN messages. CAN Transceiver

CAN transceiver is responsible for the transmission or the reception of the data on the CAN bus. It converts the data signal into the stream of data collected from the CAN bus that the CAN controller can understand. In the above diagram, unshielded twisted pair cable is used to transmit or receive the data. It is also known as CAN bus, and CAN bus consists of two lines, i.e., CAN low line and CAN high line, which are also known as CANH and CANL, respectively. The transmission occurs due to the differential voltage applied to these lines. The CAN uses twisted pair cable and differential voltage because of its environment. For example, in a car, motor, ignition system, and many other devices can cause data loss and data corruption due to noise. The twisting of the two lines also reduces the magnetic field. The bus is terminated with 120â„¦ resistance at each end. CAN Characteristics With the help of differential voltage, we will determine how 0 and 1 are transmitted through the CAN bus. The above figure is the voltage graph that shows the voltage level of CAN low and CAN high. In CAN terminology, logic 1 is said to be recessive while logic 0 is dominant. When CAN high line and CAN low line are applied with 2.5 volts, then the actual differential voltage would be zero volt. A zero volt on CAN bus is read by the CAN transceiver as a recessive or logic 1. A zero volt on CAN bus is an ideal state of the bus. When CAN high line is pulled up to 3.5 volt and the CAN low line is pulled down to 1.5 volt, then the bus's actual differential voltage would be 2 volts. It is treated as a dominant bit or logic 0 by the CAN transceiver. If the bus state is reached to dominant or logic 0 then it would become impossible to move to the recessive state by any other node. Key points learnt from the CAN characteristics Logic 1 is a recessive state. To transmit 1 on CAN bus, both CAN high and CAN low should be applied with 2.5V.

Logic 0 is a dominant state. To transmit 0 on CAN bus, CAN high should be applied at 3.5V and CAN low should be applied at 1.5V.

The ideal state of the bus is recessive.

If the node reaches the dominant state, it cannot move back to the recessive state by any other node. CAN bus logic

From the above scenario, we get to know that the dominant state overwrites the recessive state. When the node sends the dominant and the recessive bit simultaneously, then the bus remains dominant. The recessive level occurs only when all the nodes send the recessive bit. Such logic is known as AND logic, and physically it is implemented as an open collector circuit. CAN Communication Principle As we know that the message is sent based on the priority set in the arbitration field. For the standard frame, the message identifier is 11 bit, while for the extended frame, the message identifier is 29 bit. It allows the system designer to design the message identifier at the design itself. The smaller the message identifier, the higher, would be the message priority. Let's understand how arbitration works through a flow chart. The sender wants to send the message and waiting for the CAN bus to become idle. If the CAN bus is idle, then the sender sends the SOF or the dominant bit for the bus access. Then, it sends the message identifier bit in the most significant bit. If the node detects the dominant bit on the bus while it has transmitted the recessive bit, it means that the node has lost the arbitration and stops transmitting further bits. The sender will wait and resend the message once the bus is free. CAN Arbitration Example If we consider three nodes, i.e., Node 1, Node 2, and Node 3, the message identifiers of these nodes are 0x7F3, 0x6B3, and 0x6D9, respectively. The transmission of all the three nodes with the most significant bit is shown in the above diagram. 11th bit: As all the three bits of nodes are recessive, so bus bit will also remain recessive. 10th bit: All the nodes have 10th bit as recessive, so the bus will also remain recessive. 9th bit: Node 1 has recessive bit while other nodes have a dominant bit, so the bus will also remain dominant. In this case, node 1 has lost the arbitration, so it stops sending bits. 8th bit: Both node 2 and node 3 are sending recessive bit, so that the bus state will remain recessive. 7th bit: The node 2 is sending dominant bit while node 3 has sent recessive bit, so that the bus state will remain dominant. In this case, the node 3 has lost the arbitration, so it stops sending the message while the node 2 has won the arbitration means that it will continue to hold the bus until the message is received. Next Topic HTTP Status Codes

â†• prev next â†’

Reference

From Neurons to Neighborhoods : The Science of Early Childhood Development

Decoding the Ethics Code: A Practical Guide for Psychologists