Automotive Networks : A Review

Manoj. V [1], Niresh J [2]

[1] PG Student, Dept of Automobile Engineering, PSG College of Technology, Peelamedu, Coimbatore. India
[2] Asst Professor, Dept of Automobile Engineering, PSG College of Technology, Peelamedu, Coimbatore. India

Abstract—*In recent years, rapid growth in the field of electronics and computer technology which makes the life simpler and faster. This development hits the automobile sector, which makes increases the systems in vehicle like infotainment system, safety system and security system. These systems are integrated to know the status of the vehicle for each and every second, this is done by means of different networking protocols. In this paper, the different network architecture and protocols are discussed and which is best suited for automobile in the current scenario.*

Keywords—*Controller Area network, LIN, MOST, Ethernet, FLEXRAY and network architecture.*

I. INTRODUCTION

A modern car contains a lot of electronic devices such as advanced safety systems, powertrain control, sensors, and means for diagnostics. These subsystems have evolved over time, relying on various communication services provides different networking technologies. Communication is needed among the many circuits and functions of the vehicle. For example, when the driver presses the headlights switch on the dashboard, the headlights react. For this to occur, communication is needed between the dashboard switch and the front of the vehicle. In current vehicle systems this type of communication is handled via a dedicated wire through point-to-point connections. If all possible combinations of switches, sensors, motors, and other electrical devices in fully featured vehicles are accumulated, the resulting number of connections and dedicated wiring is enormous. Networking provides a more efficient method for today's complex in-vehicle communication. The early days of networking involved proprietary serial buses using generic UART (Universal Asynchronous Receiver/Transmitter) or custom devices.

In-vehicle networking, also known as multiplexing, is a method for transferring data among distributed electronic modules via a serial data bus. Without serial networking, inter-module communication requires dedicated, point-to-point wiring resulting in bulky, expensive, complex, and difficult to install wiring harnesses. Applying a serial data bus reduces the number of wires by combining the signals on a single wire through time division multiplexing. Information is sent to individual control modules that control each function, such as anti-lock braking, turn signals, and dashboard displays.

A decreased number of dedicated wires is required for each function, and thus reduces the size of the wiring harness. System cost, weight, reliability, serviceability, and installation are improved. Common sensor data, such as vehicle speed, engine temperature, etc. are available on the network, so data can be shared, thus eliminating the need for redundant sensors. Networking allows greater vehicle content flexibility because functions can be added through software changes. Existing systems require an additional module or additional I/O pins for each function added. Car manufacturers are discovering new features that are enabled by networking. For example, the 1996 Lincoln Continental's Memory Profile System stores each driver's preference for ride firmness, seat positions, steering assist effort, mirror positions, and even radio station presets.

II. TYPES OF VEHICLE NETWORKS

Controller Area Network

The Controller Area Network (CAN, also known as CAN Bus) is a vehicle bus standard designed to allow electronic control units and devices to communicate with each other in applications without a host computer. As an alternative to conventional multi-wire looms, CAN Bus allows various electronic components (such as: electronic control units, microcontrollers, devices, sensors, actuators and other electronic components throughout the vehicle) to communicate on a single or dual-wire network data bus up to 1 Mb/s. The CAN Bus is a message based protocol, designed originally for multiplex electrical wiring within motor vehicles, but also can be used in many other contexts.

Typically the CAN Bus is made up two wires, CAN-H (CAN High) and CAN-L (CAN Low) which connect to all the devices in the network. The signals on the two CAN lines have the same sequence of data, but their amplitudes are opposite. So if a pulse on the CAN-H line goes from 2.5V to 3.75V then the corresponding pulse on the CAN-L line goes from 2.5V to 1.25V (opposite than CAN-H). By sending the data in equal and opposite ways like this allows for greater noise immunity and therefore less chance of the data being corrupted. Status of bit with the value 0 = 2.5V differential voltage = dominant state

Status of bit with the value 1 = 0V differential voltage = recessive state. **C**AN Controller receives the transfer data from the microcomputer integrated in the control unit/device (also known as CAN Node). The CAN controller processes this data and relays it to the CAN transceiver. Also, the CAN controller receives data from the CAN transceiver, processes it and relays it to the microcomputer integrated in the control unit/device (CAN Node)**.**CAN Transceiver is a transmitter and receiver in one. It converts the data which the CAN controller supplies into electrical signals and sends this data over the data bus lines. Also, it receives data and converts this data for the CAN controller. CAN Data Bus Terminal is a resistor (R) typically of 120 ohms. It prevents data sent from being reflected at the ends and returning as an echo. Supplying Data: The CAN Node provides data to the CAN controller for transfer. Sending Data: The CAN transceiver receives data from the CAN controller, converts it into electrical signals and sends them back into the network. Receiving Data: All other CAN Nodes networked with the CAN data bus become receivers. Checking Data: The CAN Node checks whether they require the data they have received for their functions or not. Accepting Data: If the received data is important, it is accepted and processed. If not, the received data is ignored.

There are two different ISO standards for CAN systems that relate to the physical layer: ISO 11898-3 low speed CAN up to 125 kb/s (distance up to 500 m) and ISO 11898-2 high speed CAN up to 1 Mb/s (distance up to 40 m).The CAN system is further divided into two formats for the message frames 2.0A and 2.0B, the two standards differ in the size of the identifiers (ID)Standard CAN (version 2.0A) uses 11 bit identifiers in the arbitration field. Extended CAN (version 2.0B) supports a length of 29 bits for the identifier, made up of the 11 bit identifier (base identifier) and an 18 bit extension.

CAN BUS DATA MESSAGE STRUCTURE

Start Field is the beginning of a message with a dominant bit. So this field marks the start of the data rotocol. A bit with 3.75 V (depending on the used system) is sent over the CAN-H line and a bit with 1.25 V is sent over the CAN-L line, i.e. the differential voltage is 2.5V. Message Identifier defines the level of priority of the data protocol. If, for instance, two CAN Nodes want to send their data protocol simultaneously, the CAN Node with the higher priority takes precedence. The lower the value the higher the priority of the message. As stated earlier, depending on the standard being used, the length of the frames can be in two formats: standard, which uses an 11 bit arbitration ID, and extended, which uses a 29 bit arbitration ID. Control, also known as Check Field

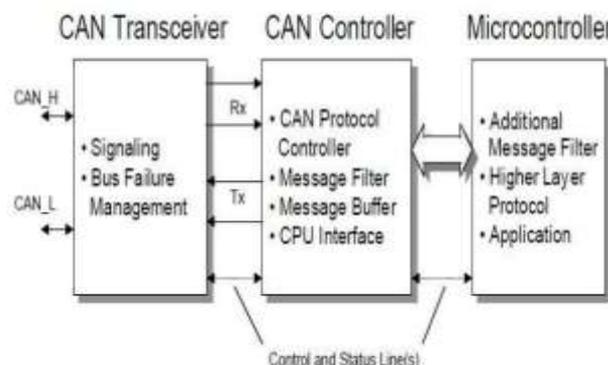displays the number of items of information contained in the data field.

Fig.1: Can architecture

This field allows any receiver to check whether it has received all the information transferred to it. Data Field, in this field the information is transferred to the other CAN Nodes. CRC = Cyclic Redundancy Check, also known as Safety Field contains 15 bit cyclic redundancy check code and a recessive delimiter bit. The CRC field is used for transfer faults detection. ACK = Acknowledge Field, also known as Confirmation Field, in this field the receivers signal to the transmitter that they have correctly received the data protocol. If an error is detected, the receivers notify the transmitter of this immediately. The transmitter then sends the data protocol again. EF = End Field, this field marks the end of the data protocol. This is the last possibility to indicate errors which lead to a repeat transfer. As stated earlier, at the CAN system there is no master that controls when individual CAN Nodes have access to read and write data on the CAN Bus. When a CAN Node is ready to transmit data, it checks to see if the Bus is busy and then simply writes a CAN frame (message) onto the network. The CAN frames that are transmitted do not contain addresses of either the transmitting node or any of the intended receiving node(s). Instead, an arbitration ID that is unique throughout the network labels the frame. All CAN Nodes on the CAN network receive the CAN frame, and depending on the arbitration ID of that transmitted frame, each CAN Node on the network decides whether to accept or ignore the received frame. If multiple CAN Nodes try to transmit a message onto the CAN Bus at the same time, the node with the highest priority (lowest arbitration ID) automatically gets Bus access. Lower-priority CAN Nodes must wait until the Bus becomes available before trying to transmit again. In this way, you can implement CAN networks to ensure deterministic communication among CAN Nodes.

LOCAL INTERCONNECT NETWORK (LIN)

LIN (Local Interconnect Network) is a concept for low cost automotive networks, which complements the

existing portfolio of automotive multiplex networks. LIN will be the enabling factor for the implementation of a hierarchical vehicle network in order to gain further quality enhancement and cost reduction of vehicles. The standardization will reduce the manifold of existing low-end multiplex solutions and will cut the cost of development, production, service, and logistics in vehicle electronics. The LIN standard includes the specification of the transmission protocol, the transmission medium, the interface between development tools, and the interfaces for software programming. LIN promotes the interoperability of network nodes from the viewpoint of hardware and software, and a predictable EMC behaviour.

WORKFLOW

The LIN workflow concept allows for the implementation of a seamless chain of design and development tools and it enhances the speed of development and the reliability of the LIN cluster. The LIN Configuration Language allows for safe sub-contracting of nodes without jeopardizing the LIN system functionality by e.g. message incompatibility or network overload. It is also a powerful tool for debugging of a LIN cluster, including emulation of non-finished nodes. The LIN Node Capability Language, which is a new feature in LIN 2.0, provides a standardized syntax for specification of off-the-shelves slave nodes. This will simplify procurement of standard nodes as well as provide possibilities for tools that automate cluster generation. Thus, true Plug-and-Play with nodes in a cluster will become a reality.

An example of the intended workflow is depicted below:
The slave nodes are connected to the master forming a LIN cluster. The corresponding node capability files are parsed by the system defining tool to generate a LIN description file (LDF) in the system definition process. The LDF is parsed by the System Generator to automatically generate LIN related functions in the desired nodes (the Master and Slave3 in the example shown in the picture above). The LDF is also used by a LIN bus analyzer/emulator tool to allow for cluster debugging.

NODE CONCEPT

The workflow described above generates the complete LIN cluster interaction module and the developer only has to supply the application performing the logic function of a node. Although much of the LIN specifications assumes a software implementation of most functions, alternative realizations are promoted. In the latter case, the LIN documentation structure shall be seen as a description model only: A node in a LIN cluster interfaces to the physical bus wire using a frame transceiver.

The frames are not accessed directly by the application; a signal based interaction layer is added in between. As a complement, a diagnostic interface exist between the application and the frame handler, as depicted below.

MASTER AND SLAVE

A LIN cluster consists of one master task and several slave tasks. A master node1 contains the master task as well as a slave task. All other nodes contain a slave task only. A sample LIN cluster with one master node and two slave nodes is depicted below:
A node may participate in more than one cluster. The term node relates to a single bus interface of a node if the node has multiple LIN bus interfaces. The master task decides when and which frame shall be transferred on the bus. The slave tasks provide the data transported by each frame. Both the master task and the slave task are parts of the Frame handler.

FRAMES

A frame consists of a header (provided by the master task) and a response (provided by a slave task).
The header consists of a break and sync pattern followed by an identifier. The identifier uniquely defines the purpose of the frame. The slave task appointed for providing the response associated with the identifier transmits it, as depicted below. The response consists of a data field and a checksum field. The slave tasks interested in the data associated with the identifier receives the response, verifies the checksum and uses the data transported.

This results in the following desired features:
System flexibility: Nodes can be added to the LIN cluster without requiring hardware or software changes in other slave nodes.
Message routing: The content of a message is defined by the identifier2.
Multicast: Any number of nodes can simultaneously receive and act upon a single frame.
Data transport Two types of data may be transported in a frame; signals or diagnostic messages. Signals are scalar values or byte arrays that are packed into the data field of a frame. A signal is always present at the same position of the data field for all frames with the same identifier. Diagnostic messages are transported in frames with two reserved identifiers. The interpretation of the data field depends on the data field itself as well as the state of the communicating nodes. The master task (in the master node) transmits frame headers based on a schedule table. The schedule table specifies the identifiers for each header and the interval between the start of a frame and the start of the following frame. The master application may use different schedule tables and select among them.

The LIN Specification Package consists of the following specifications:

- The LIN Physical Layer Specification describes the physical layer, including bit rate, clock tolerances, etc.
- The LIN Protocol Specification describes the data link layer of LIN.
- The LIN Diagnostic and Configuration Specification describes the service that can be layered on top of the data link layer to provide for diagnostic messages and node configuration.
- The LIN API Specification describes the interface between the network and the application program, including the diagnostic module.
- The LIN Configuration Language Specification describes the format of the
- LIN description file, which is used to configure the complete network and serve as a common interface between the OEM and the suppliers of the different network nodes, as well as an input to development and analysis tools.
- The LIN Node Capability Language Specification describes a format used to describe off-the-shelf slave nodes that can be used with a Plug-and-Play tool to automatically create LIN description files.

FLEXRAY

FlexRay is an automotive networking standard that was developed by the FlexRay consortium which disbanded in 2009. Members of the FlexRay consortium before its dissolution included BMW, Volkswagen, Daimler and General Motors. The main advantages of FlexRay over CAN are its flexibility, higher maximum data rate (10Mbps) and its deterministic, time triggered, TDMA behaviour. However, FlexRay nodes are more expensive than CAN nodes which can be unappealing for high volume manufacture. It provides constant latency and jitter through clock synchronisation. Its tight latency and time characteristics mean that is is often used as part of 'drive-by-wire' applications where deterministic performance is critical. A similar standard is TTP.

MOST

MOST was developed to primarily support networking of multimedia data. The maximum possible bandwidth as defined by the MOST150 standard is 150 Mb/s, which makes it much more suitable than CAN for multimedia data transmission.While the MOST Cooperation published the MOST specification, it lacks specific details relating to the data link layer (OSI Layer 2), making these details available only on a royalty basis.

ETHERNET

Ethernet is a commonly utilized communication bus, which is the communication technology of choice for much of the Internet due to its cost, speed, and flexibility. A motivating force for Ethernet for use in vehicles is the increased bandwidth that it offers. Legacy technologies such as CAN and MOST were specifically developed for automotive applications and, as such, offer an advantage in that they are tailored with in-vehicle communication in mind. At the time of their inception, the bandwidth levels provided were sufficient for the applications that they supported, i.e., by modern standards, low-bandwidth control applications, but this is no longer the case. Ethernet has already superseded CAN bus connections for interfacing with diagnostic equipment due to its increased bandwidth, example of the time taken to flash the firmware of a vehicle. Using a CAN based network, this process takes 10 h when flashing an 81-MB firmware update. Using an Ethernet network and a much larger 1-GB update, this procedure takes 20 min. Driver assistance applications are a rapidly expanding area of research. The placement of a variety of sensors around and throughout a vehicle allows for the development of new and exciting safety features such as collision avoidance, lane departure detection, traffic sign classification, blind spot detection, driver intent detection, pedestrian detection, automatic cruise control, and many others. These sensors are being used to communicate information to the driver in useful and innovative ways. These applications take advantage of high-bandwidth sensors around the vehicle, such as 24-GHz short-range or 77-GHz long-range RADAR sensors, ultrasonic, infrared cameras, and RGB optical video cameras. In, we estimate the raw bandwidth requirements of a single 1280×960 pixel resolution camera stream at 30 frames/s, with a depth of 8 bits per pixel for each of the red, green, and blue colour channels. This calculation assumes the transmission of uncompressed video, which is not uncommon among currently commercially available Ethernet camera modules. The transmission of uncompressed video is far beyond the capabilities of current generation technologies but could be supported using gigabit Ethernet.

III. CONCLUSION

Hence the improvement in the automobile sector which is totally heading towards the communication and networking which further leads to development in the automobile has been reviewed. The network schemes involved in safety, comfort and infotainment systems for the passengers have been considered for the study.

REFERENCES

[1] H. J. Song, H. P. Hsu, R. Wiese, and T. Talty, "Modeling signal strength range of TPMS in utomobiles," in Proc. IEEE Antennas Propag. Soc. Int. Symp., 2004, vol. 3, pp. 3167–3170.

[2] W. Niu, J. Li, and T. Talty, "Intra-vehicle UWB channel measurements and statistical analysis," in Proc. IEEE GLOBECOM, 2008, pp. 1–5.

[3] T. ElBatt, C. Saraydar, M. Ames, and T. Talty, "Potential for intra-vehicle wireless automotive sensor networks," in Proc. IEEE Sarnoff Symp., 2006, pp. 1–4.

[4] I. Rouf, "Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study," in Proc. USENIX Security Symp., 2010, pp. 323–338.

[5] CAN Specification, Robert Bosch GmbH, Stuttgart, Germany, 1991.

[6] R. Makowitz and C. Temple, "FlexRay—A communication network for automotive control systems," in Proc. IEEE Int. Workshop Factory Commun. Syst., 2006, pp. 207–212.

[7] OPEN Alliance SIG, OPEN Alliance: About, 2012.

[8] A Vnu Alliance, No Title, 2012. [Online]. Available: http://www.avnu.org/

[9] JasPar, JasPar, 2013. [Online]. Available: https://www.jaspar.jp/english/ index_e.php

[10] S. Tuohy, M. Glavin, E. Jones, M. Trivedi, and L. Kilmartin, "Next generation wired intra-vehicle networks, a review," in Proc. IEEE Intell. Veh. Symp., 2013, pp. 777–782.

[11] LIN Specification Package, Revision 2.0, LIN Consortium, 2003.

[12] MOST Specification Revision 2.3, MOST Cooperation, Karlsruhe, Germany, 2008.

[13] S. Huq and J. Goldie, "An overview of LVDS technology," Nat. Semicond. Appl. Note, vol. 971, pp. 1–6, 1998.

[14] D. Anderson, FireWire System Architecture: IEEE 1394a. Reading, MA, USA: Addison-Wesley, 1999.

[15] N. Navet and F. Simonot-Lion, "In-vehicle communication networks-a historical perspective and review," in Industrial Communication Technology Handbook, 2nd ed, vol. 96. Boca Raton, FL, USA: CRC Press, 2013, pp. 1204–1223.

[16] N. Navet and F. Simonot-Lion, A Review of Embedded Automotive Protocols. Boca Raton, FL, USA: CRC Press, 2008, pp. 1–42.

[17] N. Navet, Y. Song, F. Simonot-Lion, and C. Wilwert, "Trends in automotive communication systems," Proc. IEEE, vol. 93, no. 6, pp. 1204–1223, Jun. 2005.

[18] T. Nolte, H. Hansson, and L. Bello, "Automotive communications— Past, current and future," in Proc. 10th IEEE Conf. ETFA, 2005, vol. 1, p. 992.

[19] D. Paret and R. Riesco, Multiplexed Networks for Embedded Systems. Hoboken, NJ, USA: Wiley, 2007.

[20] G. Karagiannis et al., "Vehicular networking: A survey and tutorial on requirements, architectures, challenges, standards and solutions," IEEE Commun. Surveys Tuts., vol. 13, no. 4, pp. 584–616, Fourth Quarter, 2011.

[21] M. Gerla and L. Kleinrock, "Vehicular networks and the future of the mobile Internet," Comput. Netw., vol. 55, no. 2, pp. 457–469, Feb. 2011.

[22] H. Kopetz and G. Grunsteidl, "TTP—A time-triggered protocol for faulttolerant real-time systems," in Proc. 23rd Int. Symp. FTCS, Dig. Papers, 1993, pp. 524–533.

[23] G. Leen and D. Heffernan, "TTCAN: A new time-triggered controller area network," Microprocess. Microsyst., vol. 26, no. 2, pp. 77–94, Mar. 2002.

[24] H. Kopetz, "A Comparison of TTP/C and FlexRay," Institut fur Technische Informatik, Technische Universitat Wien, Wien, Austria, Res. Rep., pp. 1–22, 2001, vol. 10.

[25] M. Ruff, "Evolution of local interconnect network (LIN) solutions," in Proc. IEEE 58th VTC-Fall, 2003, vol. 5, pp. 3382–3389.

[26] M. Rabel, "Integrating IEEE 1394 as infotainment backbone into the automotive environment," in Proc. IEEE VTS 53rd VTC 2001 Spring, 2001, vol. 3, pp. 2026–2031.

[27] T. Nolte, "Share-driven scheduling of embedded networks," Ph.D. dissertation, Mälardalen Univ., Västerås, Sweden, 2006.

[28] M. Rahmani, J. Hillebrand, W. Hintermaier, R. Bogenberger, and E. Steinbach, "A novel network architecture for in-vehicle audio and video communication," in Proc. 2nd IEEE/IFIP Int.Workshop BcN, 2007, pp. 1–12.

[29] D. O. Cualain, C. Hughes, M. Glavin, and E. Jones, "Automotive standards-grade lane departure warning system," IET Intell. Transp. Syst., vol. 6, no. 1, pp. 44–57, Mar. 2012.

[30] R. M. Metcalfe and D. R. Boggs, "Ethernet: Distributed packet switching for local computer networks," Commun. ACM, vol. 19, no. 7, pp. 395–404, Jul. 1976.

[31] T. Thomsen and G. Drenkhan, "Ethernet for AUTOSAR," EB Automotive Gmbh, Erlangen, Germany, 2008.

[32] A. Doshi, B. T. Morris, andM.M. Trivedi, "On-road prediction of driver's intent with multimodal sensory

cues," IEEE Pervasive Comput., vol. 10, no. 3, pp. 22–34, Jul./Sep. 2011.

[33] B. Morris and M. Trivedi, "Vehicle iconic surround observer: Visualization platform for intelligent driver support applications," in Proc. IEEE IV Symp., 2010, pp. 168–173.

[34] M. Schneider, "Automotive radar—Status and trends," in German Microw. Conf., 2005, pp. 3–6.

[35] R. O'Malley, M. Glavin, and E. Jones, "An efficient region of interest generation technique for far-infrared pedestrian detection," in Proc. ICCE, Dig. Tech. Papers, 2008, pp. 1–2.

[36] S. Tuohy, D. O'Cualain, E. Jones, and M. Glavin, "Distance determination for an automobile environment using inverse perspective mapping in OpenCV," in IET Irish Signals Syst. Conf., 2010, pp. 100–105.

[37] R. Bruckmeier, "Ethernet for automotive applications," in Freescale Technol. Forum, 2010. [Online].Available:https://www.yumpu.com/en/document/view/21046604/ethernet-for-automotive applications freescale

[38] P. Hank, T. Suermann, and S. Müller, "Automotive Ethernet, a holistic approach for a next generation in-vehicle networking standard," in Advanced Microsystems for Automotive Applications. Berlin, Germany: Springer-Verlag, 2012, pp. 79–89.

[39] S. Carlson, Reduced Twisted Pair Gigabit Ethernet (RTPGE) Group, 2012. [Online]. Available: http://www.ieee802.org/3/RTPGE/

[40] M. Strobl et al., "Using Ethernet over powerline communication in automotive networks," in Proc. 10th Workshop WISES, 2012, pp. 39–44.

[41] W. Gouret, F. Nouvel, and G. El-Zein, "High data rate network using automotive powerline communication," in Proc. 7th Int. Conf. ITS Telecommun., Jun. 2007, pp. 1–4.

[42] F. Nouvel and P. Maziéro, "X-by-wire and intra-car communications: Power line and/or wireless solutions," in Proc. ITST, 2008, pp. 443–448.

[43] P. Tanguy, F. Nouvel, and P. Maziearo, "Power line communication standards for in-vehicule networks," in Proc. Intell. Transp. Syst. Telecommun., 2009, pp. 533–537.

[44] F. Nouvel and P. Tanguy, "What is about future high speed power line communication systems for in-vehicles networks?" in Proc. 7th Int. Conf. Inf., Commun. Signal Process., 2009, pp. 1–6.

[45] W. Gouret, F. Nouvel, and G. El-Zein, "Powerline communication on automotive network," in Proc. IEEE 65th VTC-Spring, Apr. 2007