Local Interconnect Network Communication Protocol

1st Okosa, Stephanie Chinenye
Electronic Engineering
Hochschule Hamm-Lippstadt
Hamm, Germany
stephanie-chinenye.okosa@stud.hshl.de

2nd Al Jourdi, Jad
Electronic Engineering
Hochschule Hamm-Lippstadt
Lippsatdt, Germany
jad.al-jourdi@stud.hshl.de

3rd Speczyk, Maxim Emile
Electronic Engineering
Hochschule Hamm-Lippstadt
Paderborn, Germany
maxim-emile.speczyk@stud.hshl.de

***Abstract*—The Local Interconnect Network (LIN) bus protocol is a low-cost, single-wire communication protocol commonly used in the automotive industry for controlling various subsystems within a vehicle. This academic paper presents a comprehensive review of the LIN bus protocol, specifying its physical and general requirements, as well as it's application while being a part of a complex distributed system. The challenges that LIN may encounter are also briefly brought to light at the end of this document with a summary overview of the text along with some future applications.**

***Index Terms*—LIN communication, master nodes, slave nodes.**

I. Motivation

Technological advancement in the automotive industry has exponentially increased over the years and as such, so has the complexity of electronic control units (ECU) which has lead to the realization of advanced features such as; central locking and improved the passengers' comfort. This increasing complexity has impacted the conventional wiring, and diagnostic of vehicles which has introduce multiplexed wired communications in order to achieve the complex electrical architecture using various proprietary protocols for in-vehicle networks [6].

The Local Interconnect network LIN Consortium started in 1998 and was initiated by five car manufacturers Audi, BMW, Daimler-Chrysler, Volvo, and Volkswagen, the tool manufacturer Volcano Communications Technologies (VCT), and the semiconductor manufacturer Motorola. Their intent was to formulate a standardized solution that was cost effective for the society of automotive engineers SAE communication class A (systems up to 20 kbps) systems.

LIN is an open communication standard, which is a universal asynchronous receiver transmitter (UART) based single-master, multiple-slave architecture [6], that allows fast and cost-efficient implementation of low-cost multiplex systems [1].

II. Protocol Implementation

LIN operates using the Master-slave principle i.e. it has a master node and connects to several slave nodes. The LIN master node connects the LIN network with higher level networks [6], very often as a subnet to a main network communication area network CAN. LIN communication operates on a producer-consumer model [6] where, the LIN network is message identifier based. This enables messages to be transmitted to multiple destinations with each message transmitted by the message header though the master. The message header consists of; a synchronization break that acts as a unique identifier to start the frame, a synchronization field with the clock information, and the message identifier that indicates the message's meaning [1]. The nodes understand precisely the message received from the message identifiers and one node sends the response message whilst others listen or are unresponsive.

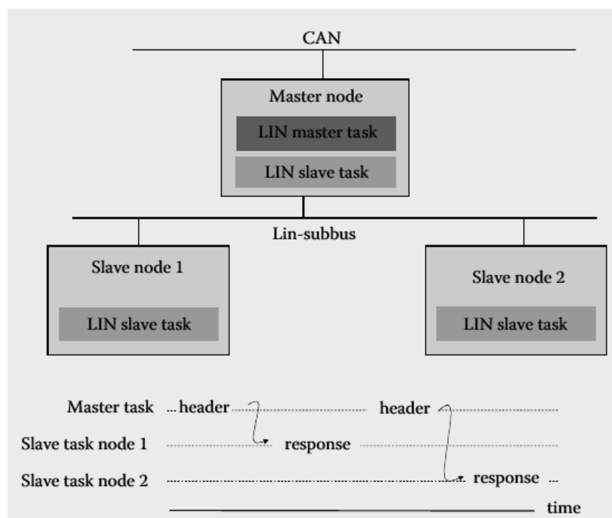

Fig. 1. Master-slave concept of LIN
[6]

The master task is responsible for initiating every communication. To begin communication, a message is generated by the master task which transmits a header with a frame identifier(ID), and then a response associated with the ID is received by the slave task by evaluating the frame header. This response consists of one to eight data bytes and a protective checksum byte.The master node allows messages to be sent to the slave nodes along with the master task, and also permits slave tasks to be merged [6].

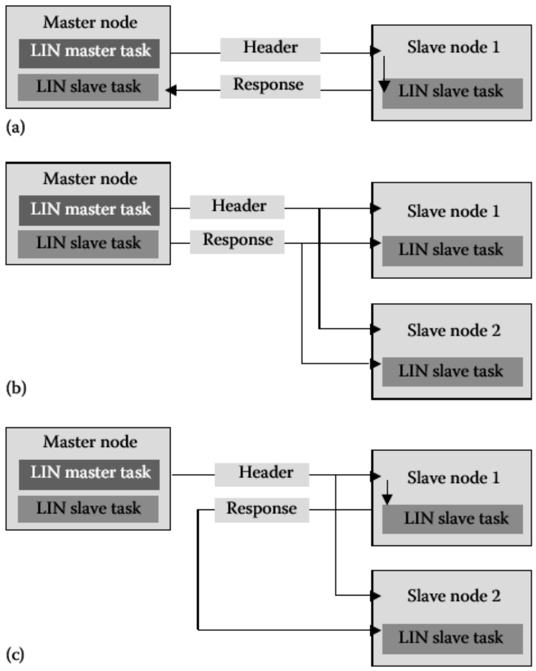

Fig. 2. Communication relationships within a LIN cluster: (a) a slave responds to a master's request, (b) the master sends a message to one or more slaves, and (c) the master initiates the communication between the two slaves. [6]

The master and slave nodes are connected by means of a simple wire bus. The LIN bus is a low cost serial protocol with low data transfer rate. It is comprised of a master node and with a maximum of 16 slave units [3]. Figure [3] is a sequence diagram illustration of LIN Communication between two modules.

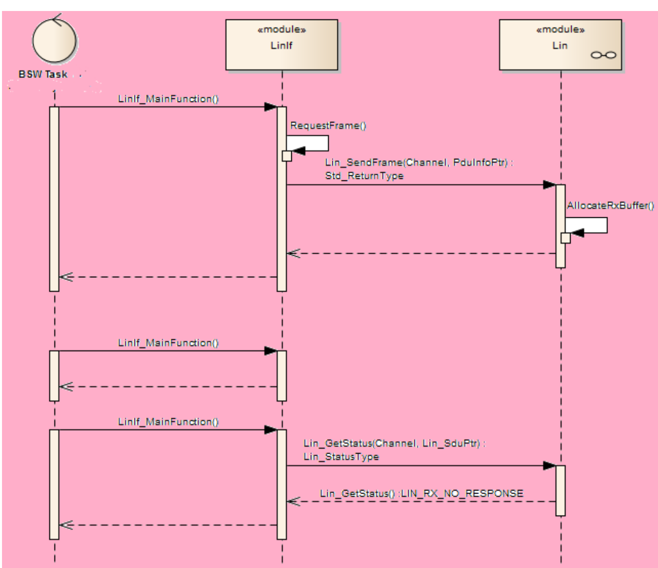

Fig. 3. Sequence Diagram LIN communication between two Module [8]

A. Requirements Specification

To be able to implement different nodes within a LIN cluster, two types of nodes have to be specified. The first type is referred to as a slave or responder node, while the other is the master or commander node. Only one master node can exist within a cluster however, up to 16 slave nodes can be present.

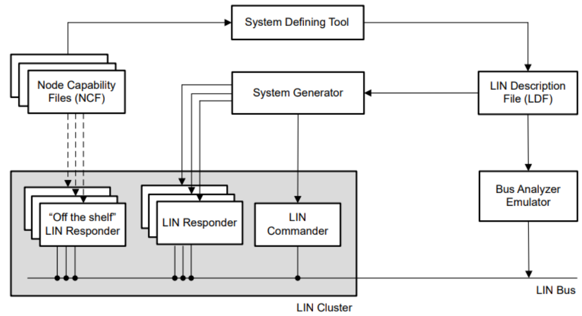

Fig. 4. LIN Cluster [5]

B. Design Process Workflow

- **Node concept**
  Using a frame transceiver, a node in a cluster connects to the physical bus wire. The application adds a layer of signal-based interaction before accessing frames. A transport layer interface further exists between the frame handler and the application. This concept can be seen in Fig.5 [4]

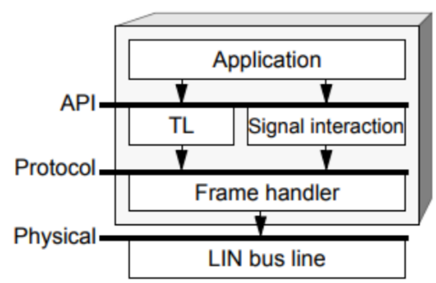

Fig. 5. API to Phys Layer Diagram [5]

- **Workflow concept**
  The LIN workflow concept accelerates development and increases the dependability of the LIN cluster by enabling the integration of a continuous chain of design and development tools, through the following specifications. [4]
  - **The Configuration Language Specification**
    The LDF is written using syntax defined by the LIN Configuration Language Specification. [5] Safe node subcontracting is permitted by specification without compromising the functionality of the LIN system. It is also an effective tool for LIN cluster debugging. [4]

- **The Node Capability Language Specification**
  The Node Capability Language Specification offers a standard syntax for specifying off-the-shelf slave nodes, simplifies the ability to add and remove said nodes freely, as well as automating cluster generation in the process. [4]

C. Physical Layer Requirements

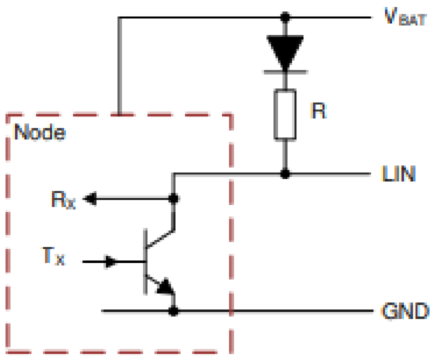

Fig. 6. LIN driver schematic [5]

The LIN physical layer, which is based on the ISO 9141 standard, is a bidirectional bus communication interface biased to a battery voltage through a resistor and diode, and is connected to the transceiver of every node in the LIN cluster.

The transceiver is what makes it possible for the bus and network to communicate. Bit logic from the microcontroller is converted by the LIN transceiver into higher voltage levels for transmission along the bus and vice versa. Through voltage translation that occurs when the signals flow through the transceiver, the TXD (transmit) and RXD (receive) within the LIN transceiver permit communication to and from the bus. The message is transmitted f to the TXD from the microcontroller, where it is broadcast on the LIN bus. In order for the microcontroller to understand and respond to the communication taking place on the bus, the RXD monitors the bus and transforms the messages on the LIN bus into voltage levels. The TXD and RXD's normal voltage levels are typically 3.3 V and 5 V, as most microcontroller levels, whereas LIN transceivers and the LIN bus typically run at voltages between 9 and 18 volts, while some can reach as high as 30 volts (depending on application). [5]

*1) **Bus Signaling Fundamentals**:* There are thresholds that most transceivers abide by. These thresholds are based on the battery voltage of the system, due to the fact it cannot be set by a voltage differential. The thresholds determine whether a bit is "recessive" or "dominant".
The network is equivalent to an open drain circuit, which means the bus requires a pull-up resistor and all nodes are connected the bus through transceivers. The pull-up resistor ensures the voltage levels on the bus are close to the battery voltage level when the TXD is in the off state, or close to ground level when the TXD is in the on state. [5]

*2) **Pullup values**:* The pull-up resistor values differ between the commander and responder nodes. The pull-up value of the responder node is 30KΩ, whereas the value of the external pull-up node of the commander node is typically 1kΩ in series with a diode. [5]

*3) **Threshold values**:* Dominant and recessive voltage level requirements are met by the sender node and receiver nodes at specified threshold values. Dominant pulses are sent by the sender by driving down the voltage to 20 percent of the battery level, whereas in the case of recessive pulses, the voltage is driven to 80 percent of the battery level. Receivers interpret dominant and recessive bits at different levels. Dominant bits are perceived when the voltage level reaches 40 percent while recessive bits are perceived at 60 percent. The difference in these levels is caused by the difference of the voltage of the LIN bus and external power supply .

*4) **Bit-rate tolerance**:* LIN's bit rate ranges from 1 to 20 kbps and has a bit tolerance of ± 14%. The tolerance is driven by an on-chip oscillator that is used. The reception and transmission of messages is ensured regardless of fluctuations caused by temperature changes and voltage drift through the use of the aforementioned tolerance.

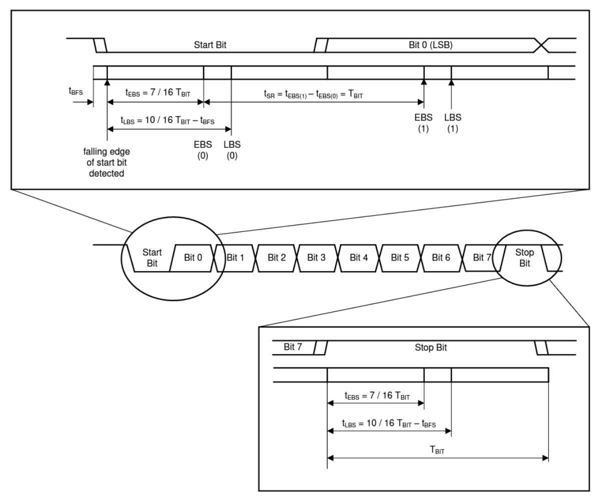

Fig. 7. BIT sampling illustration [5]

*5) **Synchronization and Bit Sampling**:* The synchronization byte consists of 8 bits alternating between 1s and Os. A start and stop are also present resulting in a total of 10 bits. Synchronization is dependant on the bit timing of the commander node, meaning that the 4 falling edges are therefore used to synchronize the responder nodes. After synchronization of the responder nodes, bit sampling occurs to ensure that the messages were interpreted properly by the LIN cluster. Every bit between t_{EBS}, the earliest bit sample, and t_{LBS}, the last bit sample, will be sampled. Every bit after the first bit is sampled with a sample rate, t_{SR}, Fig.7 uses the following equations and is a representation of the synchronization sample and bit sampling that occurs.

- $t_{BFS} = \frac{1}{16}T_{BIT}$

- $t_{EBS} = \frac{7}{16} T_{BIT}$
- $t_{LBS} = \frac{10}{16} T_{BIT} - t_{BFS}$
- $t_{SR} = t_{EBS(n)} - t_{EBS(n-1)} = T_{BIT}$

III. Applications

Local Interconnect Network (LIN) is a communication protocol commonly used in low-speed, low-cost applications. One area where LIN is particularly prevalent is in the automotive industry, where it is often used to facilitate communication between different components of a vehicle. For example, LIN can be used to control door locks, window controls, and dashboard instruments. [7]

One of the main benefits of LIN is its low cost, which is even lower than the popular Controller Area Network (CAN) protocol. This is due to LIN's simpler hardware requirements and lower data rates. Despite its lower cost, LIN still offers reliable latency, making it suitable for real-time control applications. [7]

In addition to its cost and latency benefits, LIN is also easy to use and program. It features a simple message structure that is simple to comprehend and use, and it only needs a small number of hardware components, which lowers system complexity. LIN is a well-known protocol, thus developers have access to a wealth of resources they may utilize to learn more about it. [7]

Figure 7 provides a real-world illustration of LIN communication, showcasing the implementation in a modern vehicle's cruise control system. In this setup, the steering wheel is integrated with buttons that act as the primary means of communication. The steering wheel acts as the master device, transmitting crucial data to the RPM (revolutions per minute) controller. Specifically, it sends the required RPM (revolutions per minute) value necessary for maintaining the desired speed. Subsequently, the RPM controller adjusts the RPM accordingly, effectively responding to the transmitted information.

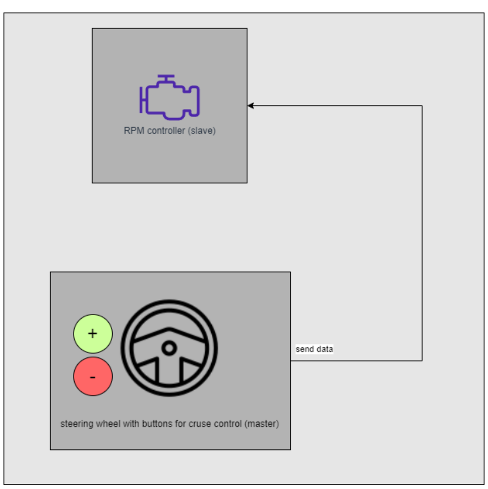

Fig. 8. Steering wheel with cruise control

IV. Challenges

Although LIN provides several benefits, there are a number of difficulties with the protocol as well.

The LIN protocol has a number of significant flaws that restrict where it can be used. Its limited data width of only 8 bits is a serious drawback. Due to this restriction, it is unsuited for handling more difficult communication tasks, such sending multimedia content or sharp photos. Larger data widths are frequently needed for certain types of data in order to ensure correct representation and dependable transmission. [7]

Additionally, the LIN protocol is vulnerable to possible security issues due to the poor level of safety it offers. Its simplicity and lack of sophisticated security protections can leave it open to hostile assaults, which could jeopardize the communication's integrity and confidentiality. [9]

The LIN protocol's poor data transfer rate of only maximum 20 Kbit/s is another constraint in addition to the data width and security issues. The slower speed of LIN, in contrast to alternative protocols that provide higher transfer rates, can lead to considerable delays and decreased effectiveness in applications that demand rapid data interchange. [7] Some industries that frequently rely on real-time and high-speed communication may find that LIN is insufficient for their requirements.

Furthermore, LIN's master-slave design poses a dependability issue. If the master node in a LIN network fails, the entire system may collapse. [7] LIN is less robust and reliable due to this single point of failure, particularly in important systems that need to operate continuously and without interruption.

Overall, while the LIN protocol has its strengths in simple, cost-effective, and low-power communication scenarios, these limitations related to data width, security, transfer rate, and reliability should be carefully considered when selecting a communication protocol for applications with more demanding requirements.

V. Conclusion

In conclusion, the Local Interconnect Network (LIN) protocol has emerged as a valuable communication tool in the automotive industry, particularly for low-speed and low-cost applications such as dashboard instruments, door locks, and window controls. Its cost-effectiveness sets it apart from other protocols like Controller Area Network (CAN), as LIN offers a more affordable solution due to its slower data rates and less demanding hardware requirements. Despite its lower cost, LIN still provides reliable latency and reliability, making it suitable for straightforward control applications.

It's crucial to remember that LIN's security measures are not as strong as those of certain other protocols. The protocol lacks sophisticated security features, which might expose it to security risks and jeopardize the communication's integrity and secrecy. While this may be a concern in certain contexts, it is worth mentioning that for the specific applications where LIN is commonly used the safety risks are generally low. LIN operates within closed systems with limited exposure to external threats, reducing the need for extensive security measures.

Overall, LIN offers an economical and reliable solution for communication within the automotive industry, particularly for simple control tasks. However, in applications where stronger security measures are required or where higher data rates are essential, alternative protocols may need to be considered. The selection of the appropriate communication protocol should always be based on a careful assessment of the specific requirements and constraints of the given application.

VI. Declaration of Originality

I, Maxim Emile Speczyk, herewith declare that I have composed the present paper and work by myself and without the use of any other than the cited sources and aids. Sentences or parts of sentences quoted literally are marked as such; other references with regard to the statement and scope are indicated by full details of the publications concerned. The paper and work in the same or similar form have not been submitted to any examination body and have not been published. This paper was not yet, even in part, used in another examination or as a course performance. I agree that my work may be checked by a plagiarism checker.

Paderborn, 14.06.203 Speczyk

Date&Place - Maxim Emile Speczyk

I, Stephanie Chinenye Okosa, herewith declare that I have composed the present paper and work by myself and without the use of any other than the cited sources and aids. Sentences or parts of sentences quoted literally are marked as such; other references with regard to the statement and scope are indicated by full details of the publications concerned. The paper and work in the same or similar form have not been submitted to any examination body and have not been published. This paper was not yet, even in part, used in another examination or as a course performance. I agree that my work may be checked by a plagiarism checker.

Hamm, 14.06.2023 Steph.O

Date&Place - Stephanie Chinenye Okosa

I, Jad Al Jourdi, herewith declare that I have composed the present paper and work by myself and without the use of any other than the cited sources and aids. Sentences or parts of sentences quoted literally are marked as such; other references with regard to the statement and scope are indicated by full details of the publications concerned. The paper and work in the same or similar form have not been submitted to any examination body and have not been published. This paper was not yet, even in part, used in another examination or as a course performance. I agree that my work may be checked by a plagiarism checker.

Lippstadt, 14.06.2023 al Jourdi

Date&Place - Jad Al Jourdi

References

[1] Richard Zurawski: Industrial Communication Technology Handbook. With assistance of Vlado Altmann, Dhananjay Anand, Herbert Barthel, Rudy Belliardi, Hendrik Bohn, Samuel P. Buckwalter et al. 2nd Edition. Available online at https://www.oreilly.com/library/view/industrial-communication-technology/9781482207323/xhtml/C58_section.xhtml, checked on 4.18.2023.

[2] Tan, Mary Tamar; Bailey, Brian; Lin, Han; Microchip Technology Inc. (2017): LIN Basics and Implementation of the MCC LIN Stack Library on 8-Bit PIC® Microcontrollers. Available online at http://ww1.microchip.com/downloads/en/appnotes/00002059b.pdf, checked on 4/18/2023.

[3] Artal, J. S.; Caraballo, J.; Dufo, R. (2014 - 2014): CAN/LIN-Bus protocol. Implementation of a low-cost serial communication network. In : 2014 XI Tecnologias Aplicadas a la Ensenanza de la Electronica (Technologies Applied to Electronics Teaching) (TAEE). 2014 XI Tecnologias Aplicadas a la Ensenanza de la Electronica (Technologies Applied to Electronics Teaching) (TAEE). Bilbao, Spain, 6/11/2014 - 6/13/2014: IEEE, pp. 1–8. Available online at https://ieeexplore.ieee.org/document/6900168/metrics#metrics, checked on 4/18/2023.

[4] LIN Consortium (2010): LIN Specification Package Revision 2.2A

[5] Eric Hackett; Texas Instruments Incorporated (2022): LIN Protocol and Physical Layer Requirements, document number: SLLA383A

[6] Bogdan M. Wilamowski, J. David Irwin, Industrial Communication Systems: CRC Press, 2018. Accessed: 02.05.23. [Online]. Available: https://learning.oreilly.com/library/view/industrial-communication-systems/9781439802823/chapter-154.html

[7] F. Oberti, E. Sanchez, A. Savino, F. Parisi, M. Brero and S. D. Carlo, ”LIN-MM: Multiplexed Message Authentication Code for Local Interconnect Network message authentication in road vehicles,” 2022 IEEE 28th International Symposium on On-Line Testing and Robust System Design (IOLTS), Torino, Italy, 2022, pp. 1-7, doi: 10.1109/IOLTS56730.2022.9897819.F. Oberti, E. Sanchez, A. Savino, F. Parisi, M. Brero and S. D. Carlo, ”LIN-MM: Multiplexed Message Authentication Code for Local Interconnect Network message authentication in road vehicles,” 2022 IEEE 28th International Symposium on On-Line Testing and Robust System Design (IOLTS), Torino, Italy, 2022, pp. 1-7, doi: 10.1109/IOLTS56730.2022.9897819.

[8] Nyssen, Alexander (2013): Model-based Automotive Software Development Model-based Automotive Software Development using Autosar, UML, and Domain-Specific Languages.

[9] J. M. Ernst and A. J. Michaels, ”LIN Bus Security Analysis,” IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society, Washington, DC, USA, 2018, pp. 2085-2090, doi: 10.1109/IECON.2018.8592744.