6. Ausgewählte Bussysteme

6.1 LIN

6.1.1 Einleitung

Der LIN- Bus (Local Interconnect Network) ist das jüngste und universellste serielle Low-Cost-Kommunikationssystem im Fahrzeug. Er wurde ins Leben gerufen, um einen offenen Standard “unterhalb” von CAN zu generieren, dort, wo CAN zu aufwändig und zu teuer ist. Federführend bei der Spezifikation war ein Konsortium von Motorola, Audi, BMW, DaimlerChrysler, Volcano, VW und Volvo. 22 weitere Firmen sind als assoziierte Mitglieder angeschlossen (Adam Opel, Alcatel, Atmel, Elmos, Infineon, Philips, STM, Visteon, ZMD, und andere).
Hintergrund war, in der Klasse A einen ersten einheitlichen Standard zu schaffen im Bezug auf Systemkonfiguration, Signalübertragung, Software-Programmierung, etc. LIN ermöglicht eine kostengünstige Kommunikation für intelligente Sensoren und Aktuatoren, bei denen die Bandbreite und Flexibilität des CAN nicht erforderlich ist. Das Datenformat basiert auf SCI (UART), einem Single-Master/Multiple-Slave-Konzept. Physikalisch liegt dem LIN ein Single-Wire 12V-Bus zugrunde, eine stabilisierte Zeitbasis für eine Clock-Synchronisation ist nicht nötig.

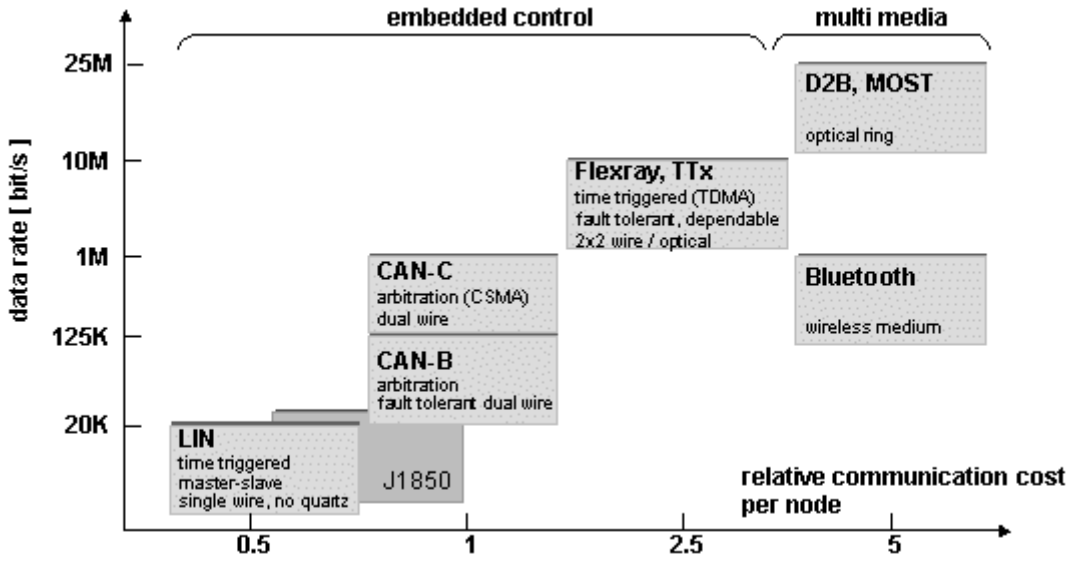

Abb. 6-1 : Einordnung des LIN in die Bussystemwelt im Kfz

LIN ist ein ganzheitliches Entwicklungskonzept. Die Spezifikation besteht nicht nur aus dem Übertragungsprotokoll, sondern umfasst gleichermaßen den Physical Layer, die Anwendungssoftware, sowie die Interfaces zu den Entwicklungs-Tools. Es beschleunigt daher die Konfiguration und Entwicklung des LIN-Netzes.
Typische Kandidaten für eine Ansteuerung über LIN sind Türmodule (Fensterheber, Schlossverriegelung, Spiegeleinstellung), Schiebedach, Steuerungen am Lenkrad

(Radio, Telefon, …), Sitzsteuerung, Heizung und Klimaregelung, Smart Scheibenwischer, Licht, Regensensoren, Starter/Generator u. v. m.

6.1.2 Das LIN Konzept

LIN ist ein Ein-Draht-Kommunikations-Protokoll, basierend auf dem genormten SCI (UART) 8-Bit Interface. UART Interfaces sind überall verfügbar, sei es als preiswerte Module auf fast allen Mikrokontrollern, sei es in Software oder Firmware, sei es als reine State-Machines in ASICs integriert. Die Arbitrierung geschieht über einen Bus-Master, so dass für für alle Slave-Knoten kein Kollisionsmanagement mehr nötig ist. Damit wird auch die maximale Übertragungszeit definiert und garantiert.

Eine Besonderheit des LIN ist der Synchronisations-Mechanismus. Er passt die Taktrate des Slave-Knotens and den Master ohne Quarz bzw. Keramik-Resonator an. Der Bus-Treiber besteht im Wesentlichen aus dem ISO 9141 Eindraht Transceiver mit wenigen Modifikationen. Mehr als 20 kbit/s werden nicht benötigt, dabei sind EMV-Aspekte (Elektromagnetische Verträglichkeit) und und Taktsynchro-nisation noch gut handhabbar.

Ein Knoten im Netzwerk muss nicht die Systemkonfiguration kennen, außer dem Master. Es können daher Knoten ohne Software- oder Hardwareänderungen der bestehenden Slaves hinzugefügt oder herausgenommen werden.

Alle diese Fakten (automatische Taktsynchronisation, die Einfachheit der UART-Kommunikation, die Ein-Draht-Verkabelung, etc.) machen den LIN zu einem kostengünstigen Medium.

6.1.3 Das Schichtenmodell

Das ISO/OSI Modell teilt Netzwerkverbindungen sieben Schichten auf. Je höher die Schicht im Modell angesiedelt ist, desto abstrakter sind ihre Funktionen.

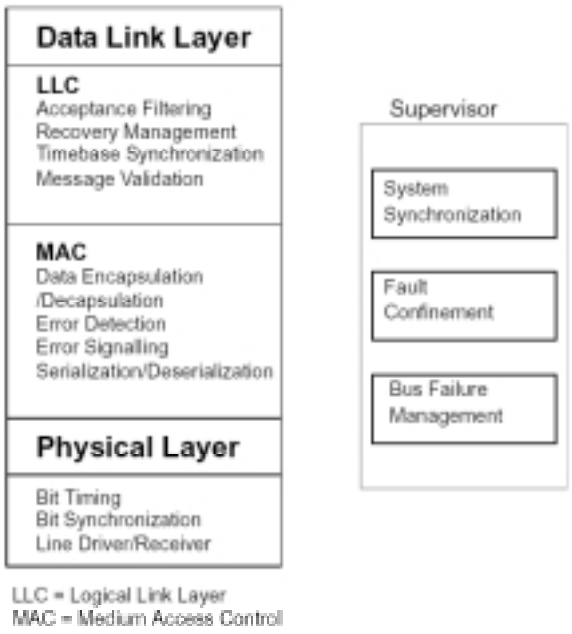

Abb. 6-2 : Einordnung des LIN in das OSI-Schichtenmodell

Der Data Link Layer entspricht der Schicht 2 (mit dem Logic Link Layer 2b und dem Medium Access Layer 2a). Darüber stehen die Vermittlungsschicht 3 und Transportschicht 4. Die obersten Schichten sind reine Anwendungsschichten. Die unterste Schicht 1 entspricht dem Physical Layer und der Leitung.

6.1.4 Der Data Link Layer

Das Netzwerk besteht aus einem Master und einen der mehreren Slave-Knoten. Jeder Slave-Knoten besteht aus einem Sende- und einem Empfangs-Teil, den sogenannten „Tasks“. Der Master-Knoten besitzt zusätzlich einen Master-Sende-Task.

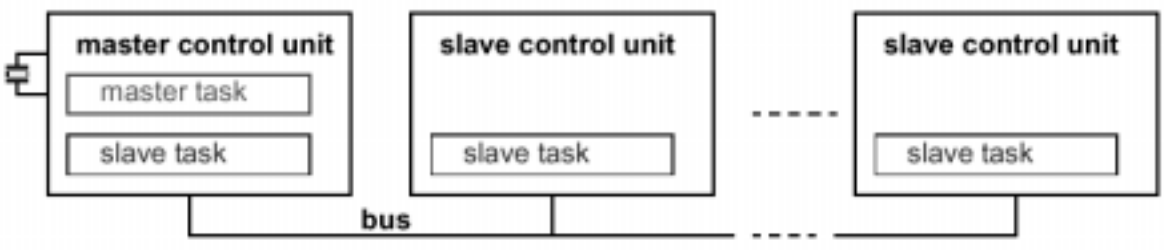

Abb. 6-3 : Master ↔ Slave Architektur

Die Übertragung im LIN-Netz wird immer vom Master mit einer Headernachricht initiiert. Dieser Message-Header besteht aus einem Synchronisations-Start (Sync Break), ein Synchronisations-Byte, sowie einem 6 Bit langen Identifier (+2 Check Bits).

Der Identifier beinhaltet Informationen des Absenders, den Empfänger, die Absicht und die Anzahl der zu übertragenen Bytes.

Der Nachrichten-Identifier aktiviert den passenden Slave Task und gibt die Anzahl der zu übertragenden Bytes an.

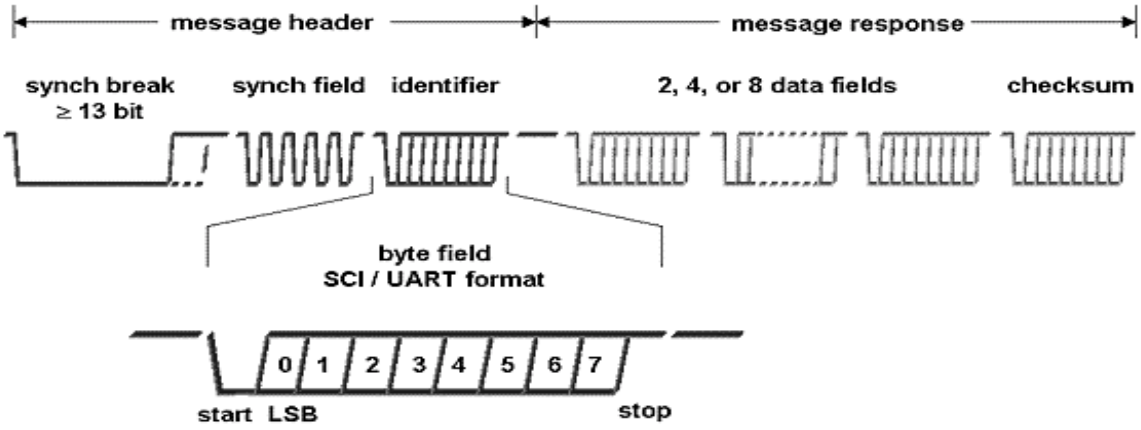

Abb. 6-4 : Codierung des Frame Headers

Der Slave seinerseits wartet auf einen „Sync Break“, anschließend wird er über das „Sync Field“ synchronisiert. Nach Auswertung der Slave Adresse und Befehl, welche sich im „Identifier Field“ befinden, wird entschieden, ob Daten zu senden oder zu empfangen, sind. Die nachfolgenden Daten bestehen aus zwei, vier oder acht Bytes plus einem Byte Checksumme. Alles zusammen umfasst den sog. Message Frame.

6.1.5 Fehlererkennung

In der Master-Nachricht schützen 2 ID Parity Check Bits das sensitive Identifier Feld:

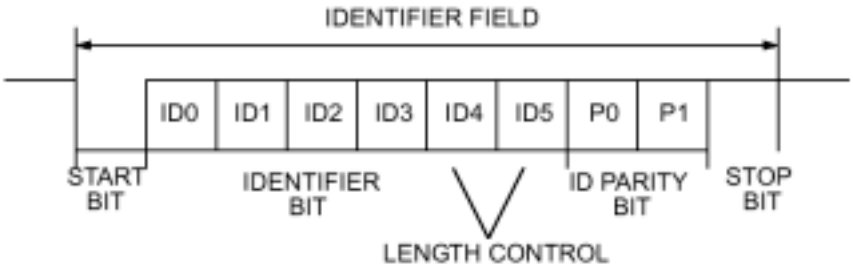

Abb. 6-5 : Codierung des Identifier-Felds

In der Slave-Antwort befindet eine 8-Bit-Wort Checksumme zur Fehlererkennung:

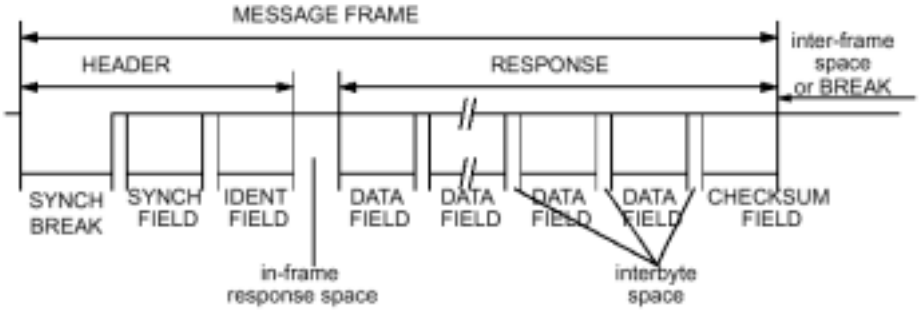

Abb. 6-6 : Codierung des gesamten Message Frames

6.1.6 Der LIN Physical Layer

Der Physical Layer ist ein Ein-Draht-Übertrager auf Basis der Kfz-Batteriespannung U_{batt}, basierend auf dem ISO9141-Standard.

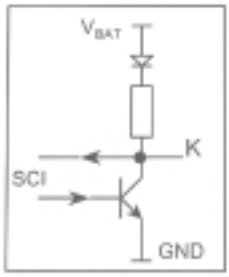

Abb. 6-7 : Prinzipschaltbild

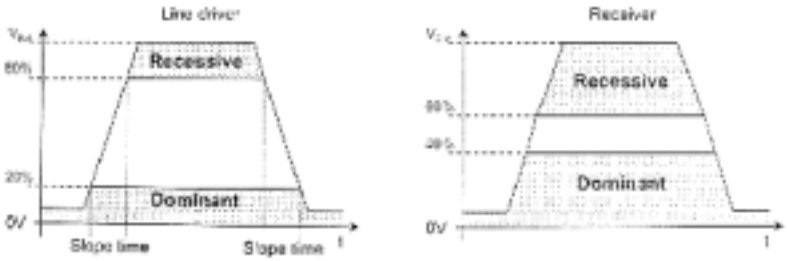

Abb. 6-8 : Sende- bzw Empfangstoleranzen

Sendeseitig wird eine Spannung von mindestens 80% U_{batt} als „high“ definiert, ein Wert unter 20% von U_{batt} gilt als „low“. Empfangseitig ist > 60% von U_{batt} = „high“, < 40% = „low“. Damit ist eine gewisse Sicherheit gegenüber Spannungsabfällen über Leitungslängen gegeben. Die Steigrate wird über den Pull-Up-Widerstand geregelt und beträgt 1-2 V / µs.

6.1.7 Software-Entwicklungsumgebung

Wir schon erwähnt, gehört zu dem Ganzheitlichen LIN-Konzept auch die entsprechende Software-Entwicklungsumgebung. Dazu gibt es im wesentlichen 4 Tools, die u. a. von dem Consortium-Mitglied Volcano Communication Technologies AB (VCT) in Göteborg entwickelt wurden.

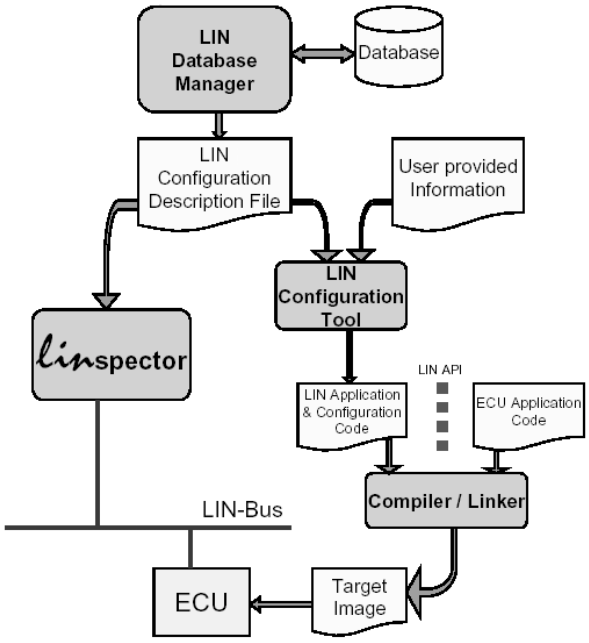

Abb. 6-9 : LIN - Entwicklungsumgebung

Mit dem LIN Database Manager (LDM) lassen sich LIN-Systeme auf einer hohen Abstraktions-Ebene logisch beschreiben und konfigurieren. Es läuft offline auf jedem PC, und zeichnet sich durch eine benutzerfreundliche Oberfläche aus.
Das LIN Application Programmers Interface (API) erlaubt es, auf einer relativ abstrakten Ebene zu entwickeln, ohne auf „Bits und Bytes“ des Datentransfers eingehen zu müssen. Zusammen mit dem LIN Configuration Tool (LCFG) und erstsprechenden Software-Tools erhält der Benutzer sowohl einerseits Flexibilität als auch andererseits die Sicherheit der Korrektheit.

Der LINspector schließlich ist ein flexibles Test- und Verifikationswerkzeug.

6.1.8 Zusammenfassung

Bussysteme unterhalb der Performance von CAN und höherwertigen Systemen stellen heute im Kraftfahrzeug die Mehrheit. Und der Markt wächst signifikant weiter! Mit LIN ist es gelungen, ein einheitliches Buskonzept bis zu einer Datenrate von ca. 20 kBit/s zu entwickeln. Überall, wo bisher konkurrierende Systeme (ISO9141, J1850, …) oder eigenentwickelte Insellösungen eingesetzt wurden, ist ein ganzheitliches System vorhanden. Entsprechende Hardware von verschiedenen Herstellern ist genauso verfügbar wie passende Software und geeignete Entwicklungs- und Test-Tools.
Beim LIN handelt es sich um ein offenes System, das einem Quasi-Standard gleichkommt. Eine formale Standardisierung ist sicher nur eine Frage der Zeit.

Literatur

LIN Specifiacation Package Revision 1.2, Nov. 2000, Issued by the LIN Consortium, Contact: Hans-Chr. V. d. Wense, Motorola, Munich

Introduction to Local Interconnect Network (LIN), SAE 2000,
Hans-Chr. V. d. Wense, Motorola, Munich

LIN-Protocol, Development Tools, and Software Interfaces for Local Interconnect Vehicles, 9th International Conference on Electronic Systems for Vehicles, Baden-Baden, Oct. 5&6, 2000, Dr.-Ing J. Will Specks, Motorola, Munich and Antal Rajnák, Volcano Communication Technologies, Gothenburg

SAE Vehicle Networks for Multiplexing and Data Communications Standards Committee, SAE J1850 Standard, “Class B Data Communications Network Interface”, Rev. May 1994

Vernetzung im Fahrzeug – von der Idee zum Silizium: K-Bus, LIN, CAN, VAN, byteflight, L. Krücke, Dr. J. Gondermann, H. Pera, W. Wetzel, ELMOS Semiconductor AG, Dortmund