Introduction

Author: Nicolas Termer, Microchip Technology Inc.

The Local Interconnect Network (LIN) is a serial network protocol developed for communication between automotive components. It was developed to be a simple, inexpensive serial alternative for smaller components that don’t need to be connected to the Controller Area Network (CAN) bus of the vehicle. Microchip’s Universal Asynchronous Receiver Transmitter (UART) module includes added protocol support features that both simplify and allow for several aspects of the LIN protocol to be performed in hardware. This technical brief will give a short summary of the LIN protocol and explain how to use the UART module for a basic LIN transaction.
# Table of Contents

Introduction ....................................................................................................................... 1  

1. LIN Overview ............................................................................................................. 3  

2. LIN Terminology ......................................................................................................... 4  

3. Bus Voltages and Levels ........................................................................................... 5  

4. Bus Layout and Communication ................................................................................ 6  
   4.1. Bus Layout ................................................................................................................. 6  
   4.2. Bus Communication .................................................................................................. 7  

5. Configuring the UART for LIN .................................................................................... 9  
   5.1. Master Mode ............................................................................................................. 9  
   5.2. Slave Mode ............................................................................................................. 10  
   5.3. Software ................................................................................................................. 10  

6. Conclusion ............................................................................................................... 11  

The Microchip Web Site ................................................................................................ 12  

Customer Change Notification Service ......................................................................... 12  

Customer Support ....................................................................................................... 12  

Microchip Devices Code Protection Feature ..................................................................... 12  

Legal Notice .................................................................................................................. 13  

Trademarks ................................................................................................................... 13  

Quality Management System Certified by DNV .............................................................. 14  

Worldwide Sales and Service ......................................................................................... 15
1. **LIN Overview**

In vehicles, the primary communication method is the CAN bus. However, it can be prohibitively expensive to put every electronic component onto the CAN bus. In addition, such a setup could cause bandwidth issues given the large number of components on the modern vehicle. As such, the LIN protocol was developed as a complementary communications protocol.
## LIN Terminology

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Master Process</strong></td>
<td>The first part of a LIN transaction, which contains the header and is followed by a response period where it waits for slave processes.</td>
</tr>
<tr>
<td><strong>Slave Process</strong></td>
<td>The second part of a LIN transaction, in which data is transmitted on the bus followed by a checksum. The device that initiates the master process may also be the one that performs the slave process.</td>
</tr>
<tr>
<td><strong>Header</strong></td>
<td>The primary section of the master process, which contains the Break, Delimiter bit, Sync Field, and PID byte.</td>
</tr>
<tr>
<td><strong>Response</strong></td>
<td>The section of the LIN transaction in which the master process relinquishes the bus and one or more slave processes transmit data.</td>
</tr>
<tr>
<td><strong>Break</strong></td>
<td>The start of a transaction, at least 13 dominant bits in a row, followed by a delimiter.</td>
</tr>
<tr>
<td><strong>Delimiter</strong></td>
<td>The separation between the break and sync fields consisting of at least one recessive bit.</td>
</tr>
<tr>
<td><strong>Sync</strong></td>
<td>The character following the Break and Delimiter bits, consisting of the value 0x55, which synchronizes the transmission rate between the master and the slaves.</td>
</tr>
<tr>
<td><strong>PID/Identifier</strong></td>
<td>The last piece of the header. This contains information about whether the master expects to send or receive data, how much data is to be sent/received, and which slave(s) will be communicated with. What each individual identifier means differs for each LIN network, and these definitions are outlined in a LIN Description File (LDF). The identifier also includes two Parity bits.</td>
</tr>
<tr>
<td><strong>V_{BAT}</strong></td>
<td>Battery voltage of the vehicle. V_{SUP} is derived from this voltage.</td>
</tr>
<tr>
<td><strong>V_{SUP}</strong></td>
<td>Primary voltage of the ECU and LIN transceivers. This can differ from V_{BAT} due to protection circuitry between the battery and the actual components.</td>
</tr>
<tr>
<td><strong>LDF</strong></td>
<td>The LIN Description File, an ASCII document that contains detailed information of a LIN bus, including the nodes on the bus, the signals, and the frames that will be sent on the bus and their meanings.</td>
</tr>
</tbody>
</table>
3. **Bus Voltages and Levels**

The LIN bus data signal operates between 0 and $V_{SUP}$ volts, with the absolute maximums of transceivers running between -0.3 and 40 volts. $V_{SUP}$ is specified to be between 7 and 18V and is typically a single power source across the entire bus. The LIN protocol is a dominant-low protocol, and both dominant and recessive levels are defined relative to $V_{SUP}$. The recessive level is any level higher than 60% of $V_{SUP}$ and dominant is any level lower than 40% of $V_{SUP}$. As these levels are outside of the normal operation range of the PIC® microcontroller (MCU), LIN communication between the PIC MCU is achieved by connecting to an external transceiver device, such as the ATA6662C, that runs off of $V_{SUP}$, and interfaces between the LIN bus and the UART RX and TX lines on the PIC MCU.
4. Bus Layout and Communication

4.1 Bus Layout

The LIN protocol specifies a bus with no more than 16 nodes: one master and up to 15 slaves. Communication is over a single shared line, terminated through a pull-up resistor and diode to the primary V\textsubscript{BAT}/V\textsubscript{SUP}. LIN is message-based with different message IDs in the header of the master process, indicating which nodes will be transmitting/receiving data during the slave process. Bit rates of LIN communications are specified to be between 1 kbps and 20 kbps.

Figure 4-1. Example LIN Bus
4.2 **Bus Communication**

Transactions on the LIN bus are broken into master and slave processes. The master process is the first part of the message. It is always initiated by the master node on the LIN bus, and is comprised of the header followed by a response period where it waits for the slave period. The header contains the following sections:

- Break
- Delimiter
- Sync byte
- PID/Identifier byte

The Break is comprised of a level transition on the line from recessive to dominant, and persists for a total of 13 bit times. Immediately following the Break is a delimiter bit, which is at least one bit time of the recessive level. Following the delimiter is a Sync byte, in which the master sends 0x55 on the LIN line, allowing for bit rate synchronization between the master and the slave devices on the bus. Finally, the master process finishes with an identifier byte for the message. Identifiers are determined by the design of each specific LIN bus and outlined in the LDF.

**Figure 4-2. LIN Header**

![LIN Header Diagram]
After the master process completes, it allows the slave process to take over the LIN bus. The slave process consists of two portions: data and checksum. The data can be sent by any of the nodes (including the master). The ID field of the header determines which specific node is transmitting, how much is being transmitted, and what data is sent. After the data has finished transmitting, the final piece of the slave process is the checksum. There are two methods of calculating checksums in the LIN protocol. LIN 1.3 and older devices calculated the checksum only on the data bytes of the message, while newer LIN standards also include the PID byte in the checksum calculation. In both, the checksum is calculated by adding together the bytes and subtracting 255 every time the sum is greater than 256. An example calculation is shown below.

**Example Data Frame:**

**PID:** 0x15  
**Data bytes:** 0x34, 0x55, 0x67, 0x97  
Checksum type: Includes PID  
0x34+0x55=0x89  
0x89+0x67=0xF0  
0xF0+0x97=0x187  
this is greater than 256(0x100), so 255 (0xFF) is subtracted.  
0x187-0xFF=0x88  
0x88+0x15=0x9D  
Final Checksum value is 0x9D.
5. Configuring the UART for LIN

The UART module can be configured for LIN mode communication via the UART Mode Select (MODE<3:0>) bits of the UART Control Register 0 (UxCON0). There are two mode selections for LIN; one for Master mode and one for Slave mode (note that the Master mode will still utilize a slave process for transmitting data).

5.1 Master Mode

The LIN Master mode is configured using the following settings:

- Load the MODE [3:0] bits of the UxCON0 register with '0b1100', placing the UART into LIN Master/Slave mode.
- Set the Transmit Enable bit (TXEN) of the UxCON0 register to allow transmission.
- Set the Receive Enable bit (RXEN) of the UxCON0 register to allow reception.
- Load a value into the UART Baud Rate Generator Register (UxBRGH:UxBRGL) pair to achieve the desired baud rate.
- Clear the Transmit Polarity Control (TXPOL) bit of the UART Control Register 2 (UxCON2) to set the transmit polarity to a high Idle state.
- Clear the Receive Polarity Control (RXPOL) bit of the UxCON2 register to set the receive polarity to a high Idle state.
- Clear the two STP [1:0] bits of the UxCON2 register for one Stop bit on both transmissions and receptions.
- If enhanced Checksum mode (including the PID in the checksum) is desired, set the C0EN bit of the UxCON2 register. Otherwise, clear this bit.
- Load the RxYPSS register (PPS output) with the TX pin selection code to map the TX output to the desired pin.
- Clear the TRIS bit associated with the TX output pin.
- Set the TRIS bit associated with the RX input pin.
- Clear the ANSEL bit associated with the RX input pin.
- Set the Serial Port Enable (ON) bit of UxCON1.

A master process will begin upon writing the PID to the lower six bits of the UxP1L register (The 2 parity bits of the PID are automatically calculated by the UART module). Doing this will automatically clear the UxTXCHK and UxRXCHK registers and have the UART module generate the LIN header (Break, Break Delimiter, Sync byte, and PID) and send them onto the bus. The PID is also received by the UART module and is stored in the receive FIFO of the UART. At this point, the LIN Master mode transitions into its slave process.

To send or receive data, software must read the PID and determine how many bytes are to be sent or received based on predetermined settings for each PID. Then, either program the UxP2H/L register pair with the number of bytes to transmit, or the UxP3H/L register with the number of bytes to receive. For transmission, the bytes must be written to the transmit FIFO of the UART, while reception will store the bytes read to the receive FIFO. In addition, the PID will determine which checksum should be used for the data that is sent or received, and the C0EN of the UxCON2 register will need to be set or cleared accordingly. After the data transmission/reception is complete, the module will return to the Idle state until the UxP1 register is written again. Any writes to the UxP1 register before the master process is complete will not create a new master process and will instead set the Transmit Write Error (TXWRE) bit of the UxFIFO register.
### 5.2 Slave Mode

The LIN slave mode is configured using the following settings:

- Load the MODE [3:0] bits of the UxCON0 register with '0b1011', placing the UART into LIN Slave mode.
- Set the Transmit Enable bit (TXEN) of the UxCON0 register to allow transmission.
- Set the Receive Enable bit (RXEN) of the UxCON0 register to allow reception.
- Clear the Transmit Polarity Control (TXPOL) bit of the UART Control Register 2 (UxCON2) to set the transmit polarity to a high Idle state.
- Clear the Receive Polarity Control (RXPOL) bit of the UxCON2 register to set the receive polarity to a high Idle state.
- Clear the two STP [1:0] bits of the UxCON2 register for one stop bit on both transmissions and receptions.
- Load the RxyPPS register (PPS output) with the TX pin selection code to map the TX output to the desired pin.
- Configure the UxRXPPS register to match the desired input pin.
- Clear the TRIS bit associated with the TX output pin.
- Set the TRIS bit associated with the RX input pin.
- Clear the ANSEL bit associated with the RX input pin.
- Set the Serial Port Enable (ON) bit of UxC.

The slave process will begin automatically upon the module receiving a Break over the LIN bus. At the end of the Break, the device will use the Sync byte to automatically configure the baud rate of the UART to match the master. The module will then automatically read the PID and store it in the receive FIFO. From that point on, it acts much in the same way as the Master/Slave mode example above: software must configure UxP2, UxP3, and the C0EN bit as predetermined by the PID, and will then transmit/receive the proper bytes from the transmit/receive FIFOs, as determined by the settings. The checksum is also automatically calculated on any received data, and if there is a checksum mismatch, the checksum error (CERIF) bit of UxERRIR will be set.

### 5.3 Software

In both Slave and Master modes, the UART module will automatically format and send/receive the proper headers and data. However, all processing of PIDs and deciding which data to send needs to be handled by application software. An example of both a basic send and receive can be found in the "LIN Master/Slave Example code using the Protocol UART" code example on MPLAB XPRESS (http://mplabxpress.microchip.com/mplabcloud/example/details/741).
6. **Conclusion**

The Local Interconnect Network (LIN) protocol is an automotive standard that connects devices within vehicles in a simpler and less expensive manner, when CAN is not required for specific applications. Microchip’s UART module has added protocol support features that simplify LIN. Module hardware automatically generates master processes, synchronizes slaves to the master, sends and receives data bytes, and calculates checksums required by the LIN protocol. For more information, please visit http://www.microchip.com.
The Microchip Web Site

Microchip provides online support via our web site at http://www.microchip.com/. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user’s guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Customer Change Notification Service

Microchip’s customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.


Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: http://www.microchip.com/support

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip’s Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
• Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable.”

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip’s code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

**Legal Notice**

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

**Trademarks**

The Microchip name and logo, the Microchip logo, AnyRate, AVR, AVR logo, AVR Freaks, BitCloud, chipKIT, chipKIT logo, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, Heldo, JukeBlox, KeeLoq, Kleer, LANCheck, LINK MD, maXSylus, maXTouch, MediaLB, megaAVR, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, Prochip Designer, QTouch, SAM-BA, SpyNIC, SST, SST Logo, SuperFlash, tinyAVR, UNI/O, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

ClockWorks, The Embedded Control Solutions Company, EtherSynch, Hyper Speed Control, HyperLight Load, IntelliMOS, mTouch, Precision Edge, and Quiet-Wire are registered trademarks of Microchip Technology Incorporated in the U.S.A.


SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.
Quality Management System Certified by DNV

ISO/TS 16949
Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company’s quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip’s quality system for the design and manufacture of development systems is ISO 9001:2000 certified.
## Worldwide Sales and Service

<table>
<thead>
<tr>
<th>AMERICAS</th>
<th>ASIA/PACIFIC</th>
<th>ASIA/PACIFIC</th>
<th>EUROPE</th>
</tr>
</thead>
</table>
| Corporate Office  
2355 West Chandler Blvd.  
Chandler, AZ 85224-6199  
Tel: 480-792-7200  
Fax: 480-792-7277  
Technical Support:  
http://www.microchip.com/  
Web Address:  
www.microchip.com  
Atlanta  
Duluth, GA  
Tel: 678-957-9614  
Fax: 678-957-1455  
Boston  
Westborough, MA  
Tel: 774-760-0087  
Fax: 774-760-0088  
Chicago  
Itasca, IL  
Tel: 630-285-0071  
Fax: 630-285-0075  
Dallas  
Addison, TX  
Tel: 972-818-7423  
Fax: 972-818-2924  
Detroit  
Novi, MI  
Tel: 248-848-4000  
Houston, TX  
Tel: 281-894-5983  
Indianapolis  
Noblesville, IN  
Tel: 317-773-8323  
Fax: 317-773-5453  
Los Angeles  
Mission Viejo, CA  
Tel: 949-462-9523  
Fax: 949-462-9608  
Tel: 951-273-7800  
Raleigh, NC  
Tel: 919-844-7510  
New York, NY  
Tel: 631-435-6000  
San Jose, CA  
Tel: 408-735-9110  
Tel: 408-436-4270  
Canada - Toronto  
Tel: 905-695-1980  
Fax: 905-695-2078  
Australia - Sydney  
Tel: 61-2-9868-6733  
China - Beijing  
Tel: 86-10-8569-7000  
China - Chengdu  
Tel: 86-28-8665-5511  
China - Chongqing  
Tel: 86-23-8980-9588  
China - Dongguan  
Tel: 86-769-8702-9880  
China - Guangzhou  
Tel: 86-20-8755-8029  
China - Hangzhou  
Tel: 86-571-8792-8115  
China - Hong Kong SAR  
Tel: 852-2943-5100  
China - Nanjing  
Tel: 86-25-8473-2460  
China - Qingdao  
Tel: 86-652-8502-7355  
China - Shanghai  
Tel: 86-21-3326-8000  
China - Shenyang  
Tel: 86-24-2334-2829  
China - Shenzhen  
Tel: 86-755-8864-2200  
China - Suzhou  
Tel: 86-186-6233-1526  
China - Wuhan  
Tel: 86-27-5980-5300  
China - Xian  
Tel: 86-29-8833-7252  
China - Xiamen  
Tel: 86-992-2388138  
China - Zhuhai  
Tel: 86-756-3210040  
India - Bangalore  
Tel: 91-80-3090-4444  
India - New Delhi  
Tel: 91-11-4160-8631  
India - Pune  
Tel: 91-20-4121-0141  
Japan - Osaka  
Tel: 81-6-6152-7160  
Japan - Tokyo  
Tel: 81-3-6880-3770  
Korea - Daegu  
Tel: 82-53-744-4301  
Korea - Seoul  
Tel: 82-2-554-7200  
Malaysia - Kuala Lumpur  
Tel: 60-3-7651-7906  
Malaysia - Penang  
Tel: 60-4-227-8870  
Philippines - Manila  
Tel: 63-2-634-9065  
Singapore  
Tel: 65-6334-8870  
Taiwan - Hsin Chu  
Tel: 886-3-577-8366  
Taiwan - Kaohsiung  
Tel: 886-7-213-7830  
Taiwan - Taipei  
Tel: 886-2-2508-8600  
Thailand - Bangkok  
Tel: 66-2-694-1351  
Vietnam - Ho Chi Minh  
Tel: 84-28-5448-2100  
Austria - Wels  
Tel: 43-7242-2244-39  
Denmark - Copenhagen  
Tel: 45-4450-2828  
Finland - Espoo  
Tel: 358-9-4520-820  
France - Paris  
Tel: 33-1-69-53-63-20  
Germany - Garching  
Tel: 49-8931-9700  
Germany - Haan  
Tel: 49-2129-3766400  
Germany - Heilbronn  
Tel: 49-7131-67-3636  
Germany - Karlsruhe  
Tel: 49-721-625370  
Germany - Munich  
Tel: 49-89-627-144-0  
Germany - Rosenheim  
Tel: 49-8031-354-560  
Israel - Ra'anana  
Tel: 972-9-744-7705  
Italy - Milan  
Tel: 39-0331-742611  
Fax: 39-0331-466781  
Italy - Padova  
Tel: 39-049-7625286  
Netherlands - Drunen  
Tel: 31-416-690399  
Fax: 31-416-690340  
Norway - Trondheim  
Tel: 47-72884388  
Poland - Warsaw  
Tel: 48-22-3325737  
Romania - Bucharest  
Tel: 40-21-407-87-50  
Spain - Madrid  
Tel: 34-91-708-08-90  
Fax: 34-91-708-08-91  
Sweden - Gothenberg  
Tel: 46-31-704-60-40  
Sweden - Stockholm  
Tel: 46-8-5090-4654  
UK - Wokingham  
Tel: 44-118-921-5800  
Fax: 44-118-921-5820