SAE International®

SURFACE VEHICLE RECOMMENDED PRACTICE

J2602-1 NOV2012

Issued 2004-08
Revised 2012-11

Superseding J2602-1 SEP2005

LIN Network for Vehicle Applications

RATIONALE

The LIN protocol is a low speed low cost communication protocol that is capable of reducing wire counts to simple devices like switches and sensors. This document defines an implementation of the LIN protocol with the focus on enabling ASIC designs commonly found in switches and sensors.

FOREWORD

The objective of this document is to define a level of standardization in the implementation of low speed vehicle serial data network communications using the Local Interconnect Network (LIN) protocol.

The goal of this document is to define a serial data physical layer, data link layer and media design criteria to be installed in various automotive Electronic Control Units (ECU). This standard will allow ECU and tool manufacturers to satisfy the needs of multiple end users with minimum modifications to the basic design. This standard will benefit vehicle Original Equipment Manufacturers (OEMs) by achieving lower ECU costs due to higher industry volumes of the basic design.

NOTE: Understanding of this document requires a working knowledge of the LIN 2.0 Specification Package.

TABLE OF CONTENTS

1. SCOPE

This document covers the requirements for SAE implementations based on LIN 2.0. Requirements stated in this document will provide a minimum standard level of performance to which all compatible ECUs and media shall be designed. This will assure full serial data communication among all connected devices regardless of supplier.

The goal of SAE J2602-1 is to improve the interoperability and interchangeability of LIN devices within a network by resolving those LIN 2.0 requirements that are ambiguous, conflicting, or optional. Moreover, SAE J2602-1 provides additional requirements that are not present in LIN 2.0 (e.g., fault tolerant operation, network topology, etc.).

This document is to be referenced by the particular vehicle OEM component technical specification that describes any given ECU in which the single wire data link controller and physical layer interface is located. Primarily, the performance of the physical layer is specified in this document. ECU environmental and other requirements, when provided in the component technical specification, shall supersede the requirements of this document.

The intended audience includes, but is not limited to, ECU suppliers, LIN controller suppliers, LIN transceiver suppliers, component release engineers and vehicle system engineers.

1.1 Mission/Theme

This serial data link network is intended for use in applications where high data rate is not required and a lower data rate can achieve cost reductions in both the physical media components and in the microprocessor and/or dedicated logic devices (ASICs) which use the network.

1.2 Overview

LIN is a single wire, low cost, Class A communication protocol. LIN is a master-slave protocol, and utilizes the basic functionality of most Universal Asynchronous Receiver Transmitter (UART) or Serial Communication Interface (SCI) devices as the protocol controllers in both Master and Slave devices. To meet the target of “Lower cost than either an OEM proprietary communications link or CAN link” for low speed data transfer requirements, a single wire transmission media based on the ISO 9141 specification was chosen. The protocol is implemented around a UART/SCI capability set, because the silicon footprint is small (lower cost). Many small microprocessors are equipped with either a UART or SCI interface (lower cost), and the software interface to these devices is relatively simple to implement (lower software cost). Finally, the relatively simplistic nature of the protocol controller (UART/SCI) and the nature of state-based operation, enable the creation of Application Specific Integrated Circuits (ASICs) to perform as input sensor gathering and actuator output controlling devices, in the vein of Mechatronics.

All message traffic on the bus is initiated by the Master device. Slave devices receive commands and respond to requests from the Master. Since the Master initiates all bus traffic, it follows that the Slaves cannot communicate unless requested by the Master. However, Slave devices can generate a bus wakeup, if their inherent functionality requires this feature.

The “LIN Consortium” developed the set of LIN specifications. The Consortium is a group of automotive OEMs, semiconductor manufacturers, and communication software and tool developers. The LIN specification set is “released” by the LIN Steering Committee, a closed subset of the members. Associate Consortium members contribute to the formation of the specifications through participation in LIN Work Groups; however, the direction of the Work Groups and the final released content of the specifications is the responsibility of the LIN Steering Committee.

The LIN Specifications contain more than just a definition of the LIN protocol and physical layer. In addition, a Work Flow Process, Diagnostics and Configuration methods, definition of an Application Program Interface (API), file structures for a Node Capability File (NCF) and a LIN Description File (LDF) and semantics are identified as required (mandatory in all implementations). However, since there is a great deal of flexibility in the protocol and physical layer, applicability of these specifications to J2602-1 networks will be further specified in this document.

1.3 Relationship to the LIN Specifications

As described in the LIN Specification Package, the LIN 2.0 protocol specification suite consists of seven documents:

1.3.1 LIN Specification Package

The **LIN Specification Package** provides an overview of the LIN Protocol, its features, and work flow. This includes the Revision History, LIN Overview and Glossary.

1.3.2 LIN Physical Layer Specification

The **LIN Physical Layer Specification** describes the physical layer, including bit rate, clock tolerances, etc.

1.3.3 LIN Protocol Specification

The **LIN Protocol Specification** describes the data link layer of LIN.

1.3.4 LIN Diagnostic and Configuration Specification

The **LIN Diagnostic and Configuration Specification** describe the services that can/may be layered on top of the data link layer to provide for diagnostic messages and node configuration.

1.3.5 LIN API Specification

The **LIN API Specification** describes the interface between the network and the application program, including the diagnostic module.

1.3.6 LIN Configuration Language Specification

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.

1.3.7 LIN Node Capability Language Specification

The **LIN Node Capability Language Specification** describes a format used to describe off-the-shelf slave nodes that can be used with a system definition tool to automatically create LIN Description Files.

The remainder of this document (SAE J2602-1) will directly reference these LIN specifications.

2. REFERENCES

2.1 Applicable Documents

The following publications form a part of this specification to the extent specified herein. Unless otherwise indicated, the latest issue of SAE publications shall apply.

2.1.1 SAE Publications

Available from SAE International, 400 Commonwealth Drive, Warrendale, PA 15096-0001, Tel: 877-606-7323 (inside USA and Canada) or 724-776-4970 (outside USA), www.sae.org.

SAE J551	All Parts - Performance Levels and Methods of Measurement of Electromagnetic Compatibility for Vehicles and Devices
SAE J1113	All Parts - Electromagnetic Compatibility Measurement Procedures for Vehicle Components
SAE J1213-1	Glossary of Vehicle Networks for Multiplexing and Data Communications
SAE J1930	Electrical/Electronic Systems Diagnostic Terms, Definitions, Abbreviations, and Acronyms - Equivalent to ISO/TR 15031-2

2.1.2 ISO Documents

Available from American National Standards Institute, 25 West 43rd Street, New York, NY 10036-8002, Tel: 212-642-4900, www.ansi.org.

ISO 7498	Data processing systems, open systems interconnection standard reference mode
ISO 7637	Road vehicles - Electrical interference by conduction and coupling - Parts 1 and 2
ISO 9141	Road vehicles - Diagnostic systems - Requirements for interchange of digital information

2.1.3 Supplier Publications

See Appendix A for list of supplier documents/devices.

2.1.4 Other Publications

LIN Specification Package version 2.0 dated September 23, 2003 available at www.lin-subbus.org.

CISPR 25	Limits and Methods of Measurement of Radio Disturbance Characteristics for the Protection of Receivers Used on Board Vehicles available at webstore.iec.ch.
ES-XW7T-1A278-AC	Ford Component and Subsystem Electromagnetic Compatibility Worldwide Requirements and Test Procedures available at www.fordemc.com. This document shall be referred to as the Ford EMC Spec.

3. DEFINITION OF TERMS

3.1 GLOSSARY

3.1.1 Command Frame

A frame with data published by the slave task in the Master Node and subscribed to by one or more slave tasks in slave nodes.

3.1.2 Data Link Layer

This provides for the reliable transfer of information across the physical layer. It includes the message structure, framing and error control.

3.1.3 Dominant Signal

The driven and low voltage state of the LIN bus. If multiple devices access the bus, this state dominates the recessive or non-driven state.

3.1.4 "Dormant" State

The state in which the slave task state machine is waiting for reception of the Break / Synch sequence.

3.1.5 Master Node

Responsible for initiating all message traffic. See the Glossary of the LIN Specification Package for additional information.

3.1.6 Media

The physical entity that conveys the electrical (or equivalent means of communication) signal transmission among ECUs on the network.

3.1.7 Physical Layer

This ISO 7498 subsection consists of the media, mechanical interconnections, and transceivers that provide the interconnection between all ECU nodes.

3.1.8 Protocol

The formal set of conventions or rules for the exchange of information among the ECUs. This includes the specification of the signal frame administration, frame transfer and physical layer.

3.1.9 Publisher

A Master or Slave Node that is the source of the data transmitted onto the bus within a LIN message.

3.1.10 Radiated Emissions

The energy that radiates from the LIN physical layer.

3.1.11 Radiated Immunity

The level of susceptibility of physical layer components to communication errors in the presence of high energy electromagnetic fields.

3.1.12 Recessive Signal

The undriven and high voltage state of the LIN bus. If multiple devices access the bus, this state is overridden by the dominant state.

3.1.13 Request Frame

A frame with data published by the slave task in one and only one Slave Node and only subscribed to by the slave task in the Master Node.

3.1.14 Slave Node

A device that receives messages from the Master Node, or responds to messages initiated by the Master Node. See the Glossary of the LIN Specification Package for additional information.

3.1.15 Subscriber

A Master or Slave Node that receives the data within a LIN message.

4. ACRONYMS, ABBREVIATIONS, AND SYMBOLS

API	Application Program Interface
ASIC	Application Specific Integrated Circuit
CAN	Controller Area Network
DLC	Diagnostic Link Connector
DNN	Device Node Number
ECU	Electronic Control Unit
EMC	Electromagnetic Compatibility
ESD	Electrostatic Discharge
ISO	International Organization for Standardization
Kbits/s	Thousands of data bits per s
LDF	LIN Description File
LIN	Local Interconnect Network
LSB	Least Significant Byte
Lsb	Least significant bit
MSB	Most Significant Byte
Msb	Most significant bit
NAD	Node Address for Diagnostics
NCF	Node Capability File
OEM	Original Equipment Manufacturer
RE	Radiated Emissions
RI	Radiated Immunity
SCI	Serial Communication Interface
UART	Universal Asynchronous Receiver/Transmitter

5. LIN SYSTEM REQUIREMENTS

All ECU LIN interfaces shall conform to the LIN Specification Package of September 2003 unless otherwise specified in this specification.

5.1 LIN Specification Package

The LIN Specification Package as described in 1.3.1 is informative only and contains no formal requirements. Also note that the information contained in this section of LIN 2.0 may or may not be representative of J2602-1-based implementations. However, the Glossary included in this section of the LIN specification contains definitions and terms needed to comprehend the LIN Protocol and J2602-1.

5.2 J2602-1 Serial Data Link Characteristics

1. Master/Slave Collision Avoidance

2. Capable of operating with LIN 2.0 and previous devices.

3. Additional definition to meet SAE requirements for serial data communication networks.

4. Slave-to-slave communication is not supported and is highly discouraged.

5.3 Detection of Errors by Master

The master task state machine shall detect errors during the transmission of the Break / Synch / Protected ID sequence. If an error is detected (e.g., data mismatch or data not received), the master shall cease transmission of the frame and shall start transmission of the next frame as dictated by the schedule table.

5.4 Frame Processing by Slave Tasks

5.4.1 Slave Task Error Detection

The slave task state machine (either in a slave node or the master) shall detect the following errors:

- ID Parity Error
- Byte Field Framing Error (i.e., invalid stop bit)
- Data Error (i.e., data transmitted does not match data read, data transmitted is not received, fixed form data received is incorrect)
- Checksum Error

5.4.2 Slave Behavior in the Presence of Errors When Transmitting

When a slave task state machine detects a Byte Field Framing Error, Data Error, or Checksum Error, the slave task state machine shall cease transmission prior to transmission of the next byte field, unless the error occurs during the transmission of the Checksum byte, and return to the "Dormant" state. The slave task shall also set the appropriate error flags as defined in 5.8.6.

5.4.3 Slave Behavior in the Presence of Errors When Receiving

When a slave task machine detects an ID Parity Error, Byte Field Framing Error, Data Error or Checksum Error, the slave task state machine shall discard any data buffered from the current frame and return to the "Dormant" state. The slave task shall also set the appropriate error flags as defined in 5.8.6.

5.5 Message Transmission Time Tolerance

Each maximum message transmission time may be specified to be within the range of $T_{Frame\_Minimum}$ and $T_{Frame\_Maximum}$, providing both publisher and subscriber of the frame support a maximum message transmission time of less than $T_{Frame\_Maximum}$.

The minimum $T_{Frame\_Maximum}$ value to which a slave node can respond shall be identified in its Node Capability File. In the event a value is not provided in the Node Capability File, the Master node shall presume a value of $T_{Frame\_Maximum}$ as defined in the LIN 2.0 Protocol Specification.

5.6 LIN Product Identification

The following requirements provide further clarification to the intent of the LIN Function ID and Variant ID assignment conditions.

5.6.1 Clarification to Function ID Paragraph

Two products (ECUs) are identical in function and shall be assigned identical Function Identifiers if they exhibit all of the following characteristics:

- They exhibit identical functional behavior
- They exhibit identical mandatory Node Capability File declarations
- Both LIN communications and Application functionality are configured identically by the IC supplier

5.6.2 Clarification to Variant ID Paragraph

Two products (ECUs) are invariant and shall be assigned identical Variant Identifiers if they exhibit all of the following characteristics:

- They have identical operating range characteristics (voltage, temperature)
- They are constructed from the identical integrated circuit processes and manufacturing technology

NOTE: Any change in the binary image loaded in a microprocessor-based slave implementation shall constitute a difference in the variant ID.

5.7 Mandatory Node Configuration Requests (reference LIN 2.0 Diag and Config Specification)

These messages are to be used for configuration only; they are not to be used for LIN diagnostics since all relevant diagnostic information is included in the J2602-1 Status Byte. Furthermore, these configuration messages are only initiated by the master; they are not initiated through the SAE J1962 connector since the LIN bus does not go to this connector.

Support of diagnostics and configuration as defined in the LIN Diagnostic and Configuration Specification in the LIN 2.0 Specification Package is **OPTIONAL** in J2602-1 nodes. In other words, support of the services defined as mandatory in of the LIN Diagnostic and Configuration Specification is optional and is **NOT** required for J2602-1 compliance.

The following sections define the required SAE J2602-1 node configuration command/response messages.

5.7.1 General Configuration Requirements

5.7.1.1 Slave Execution of Configuration

Within the limit of its capabilities, a slave device shall immediately execute any configuration command upon receipt of the $3C message; it shall not wait for a $3D message.

5.7.1.2 Slave Device Configuration Capabilities

A NCF file shall be included with a SAE J2602-1 device.

5.7.1.3 Master Configuration Message Pairing

A master shall always transmit $3C / $3D coupled pairs. The master shall never send multiple successive $3C messages without corresponding (interleaved) $3D messages. There shall be exactly one $3D response for each $3C command. The only exception to this is in the case of a broadcast $3C message which shall have no corresponding $3D message.

5.7.2 NAD and Message ID Assignment

5.7.2.1 NAD Assignment

The NAD for a J2602-1 device shall be in the range $60 to $6D, where the lower nibble of the J2602-1 NAD contains a four bit Device Node Number (DNN). An uninitialized node shall have a NAD of $6F. A NAD of $6E may be used; however, its message IDs must be assigned via $3C or $3E messages.

IC manufacturers may provide the ability to select the desired Device Node Number, and hence the NAD, based on up to four memory bits, four external pins, etc. These bits or pins shall only impact the value of the lower nibble of the NAD. If the device has a selectable NAD, it shall default to $6F prior to initialization.

J2602-1 NAD

Bit 7 (msb)	6	5	4	3	2	1	Bit 0 (lsb)
0	1	1	0	DNN3	DNN2	DNN1	DNN0

5.7.2.2 Message ID Assignment

Each device shall be assigned 4 message IDs based on the DNN, and, therefore, the NAD. If a device does not need that many messages, it shall use the messages with the lowest IDs. If a device requires more than 4 message IDs, it shall be assigned messages in powers of 2, i.e., 4, 8, 16, 32. The system designer must ensure that multiple devices do not use the same Message IDs.

As a consequence of having the message IDs associated with the DNN, after a power on reset, or reset command the protected identifiers are still marked as valid for devices with a DNN in the range $0 - $D. Devices with a DNN of $E or $F will have the protected identifiers marked as invalid as described in of the LIN Diagnostic and Configuration Specification.

The following table shows the boundaries for 4, 8, or 16 messages per node. If 32 Message IDs are required, NAD $60 must be used for this node. J2602-1 networks can combine nodes that use various numbers of Message IDs.

TABLE 1 - NAD TO MESSAGE ID MAPPING

NAD	Message ID		NAD	Message ID		NAD	Message ID
$60	$00 $01 $02 $03		$60	$00 $01 $02 $03		$60	$00 $01 $02 $03
$61	$04 $05 $06 $07			$04 $05 $06 $07			$04 $05 $06 $07
$62	$08 $09 $0A $0B		$62	$08 $09 $0A $0B			$08 $09 $0A $0B
$63	$0C $0D $0E $0F			$0C $0D $0E $0F			$0C $0D $0E $0F
$64	$10 $11 $12 $13		$64	$10 $11 $12 $13		$64	$10 $11 $12 $13
$65	$14 $15 $16 $17			$14 $15 $16 $17			$14 $15 $16 $17
$66	$18 $19 $1A $1B		$66	$18 $19 $1A $1B			$18 $19 $1A $1B
$67	$1C $1D $1E $1F			$1C $1D $1E $1F			$1C $1D $1E $1F
$68	$20 $21 $22 $23		$68	$20 $21 $22 $23		$68	$20 $21 $22 $23
$69	$24 $25 $26 $27			$24 $25 $26 $27			$24 $25 $26 $27
$6A	$28 $29 $2A $2B		$6A	$28 $29 $2A $2B			$28 $29 $2A $2B
$6B	$2C $2D $2E $2F			$2C $2D $2E $2F			$2C $2D $2E $2F
$6C	$30 $31 $32 $33		$6C	$30 $31 $32 $33		$6C	$30 $31 $32 $33
$6D	$34 $35 $36 $37			$34 $35 $36 $37		$6D	$34 $35 $36 $37
$6E	No Message IDs defined		$6E	No Message IDs defined		$6E	No Message IDs defined
$6F	No Message IDs defined		$6F	No Message IDs defined		$6F	No Message IDs defined

NOTE: The Message IDs listed in this table are the unprotected Identifiers.

5.7.2.3 Configuration Messages

LIN protocol level and application level configuration of each slave node may be accomplished using any combination of the following:

1. Using optional method defined in the LIN Diagnostic and Configuration Specification in the LIN 2.0 Specification Package.

2. Using $3C messages with NADs in the User reserved range of $80 - $FF.

3. Using $3E messages with any NAD.

When using $3C or $3E messages to configure the slaves, the lower nibble of the NAD shall be the DNN to prevent conflicts between slaves.

$3C NAD

Bit 7 (msb)	6	5	4	3	2	1	Bit 0 (lsb)
1	X	X	X	DNN3	DNN2	DNN1	DNN0

$3E NAD

Bit 7 (msb)	6	5	4	3	2	1	Bit 0 (lsb)
X	X	X	X	DNN3	DNN2	DNN1	DNN0

5.7.2.4 Response Message to Options 2 and 3 in 5.7.2.3

The $3D Response message that follows each $3C and $3E Command message shall return the J2602-1 Status Byte (defined in 5.8.6) in Data Byte 0. The value of the other seven data bytes is implementation dependent and beyond the scope of this specification.

Tx by Master	*Tx by Slave*	*Tx by Slave*	*Tx by Slave*	*Tx by Slave*	*Tx by Slave*	*Tx by Slave*	*Tx by Slave*	*Tx by Slave*
LIN ID	Data0	Data1	Data2	Data3	Data4	Data5	Data6	Data7
$3D	J2602-1 Status Byte	XX	XX	XX	XX	XX	XX	XX

5.7.2.5 DNN Based Broadcast Messages

There are four Message IDs reserved for Broadcast messages. The NAD that will use a specific Broadcast Message, and the data byte within the message that it will use is based on its DNN. The Broadcast Message IDs for a specific NAD are b11 100x and b11 101x, where x = DNN3. The relevant data byte number is equivalent to the three lsbs of the DNN. In the case where a node has more than 4 message IDs, i.e., 8 or 16, it shall also be assigned a proportionate number of Broadcast Message bytes.

5.7.2.5.1 DNN Based Broadcast Message Assignment for 4 Messages per Node

LIN ID	Data0	Data1	Data2	Data3	Data4	Data5	Data6	Data7
$38	DNN = $0	DNN = $1	DNN = $2	DNN = $3	DNN = $4	DNN = $5	DNN = $6	DNN = $7
$39	DNN = $8	DNN = $9	DNN = $A	DNN = $B	DNN = $C	DNN = $D	XX	XX
$3A	DNN = $0	DNN = $1	DNN = $2	DNN = $3	DNN = $4	DNN = $5	DNN = $6	DNN = $7
$3B	DNN = $8	DNN = $9	DNN = $A	DNN = $B	DNN = $C	DNN = $D	XX	XX

5.7.2.5.2 DNN Based Broadcast Message Assignment for 8 Messages per Node

LIN ID	Data0	Data1	Data2	Data3	Data4	Data5	Data6	Data7
$38	DNN = $0	DNN = $0	DNN = $2	DNN = $2	DNN = $4	DNN = $4	DNN = $6	DNN = $6
$39	DNN = $8	DNN = $8	DNN = $A	DNN = $A	DNN = $C	DNN = $C	XX	XX
$3A	DNN = $0	DNN = $0	DNN = $2	DNN = $2	DNN = $4	DNN = $4	DNN = $6	DNN = $6
$3B	DNN = $8	DNN = $8	DNN = $A	DNN = $A	DNN = $C	DNN = $C	XX	XX

5.7.2.5.3 DNN Based Broadcast Message Assignment for 16 Messages per Node

LIN ID	Data0	Data1	Data2	Data3	Data4	Data5	Data6	Data7
$38	DNN = $0	DNN = $0	DNN = $0	DNN = $0	DNN = $4	DNN = $4	DNN = $4	DNN = $4
$39	DNN = $8	DNN = $8	DNN = $8	DNN = $8	DNN = $C	DNN = $D	XX	XX
$3A	DNN = $0	DNN = $0	DNN = $0	DNN = $0	DNN = $4	DNN = $4	DNN = $4	DNN = $4
$3B	DNN = $8	DNN = $8	DNN = $8	DNN = $8	DNN = $C	DNN = $D	XX	XX

5.7.3 Targeted Reset

The Targeted Reset command provides a mechanism for the Master to cause a re-initialization of a specific slave device on the network, designated by the NAD in the command. Upon receipt of the Targeted Reset command, the slave device shall cause an internal reset of operational variables to occur. Examples of operational variables include, but are not limited to, program counters, mode control variables, communications error counters, input source re-initializations, and output device re-initializations, but shall not alter any previously configured application level configuration information stored in non-volatile memory. <u>This Reset operation shall not cause the slave to destructively alter any LIN configuration data or addresses stored in non-volatile memory</u>. Also, the Reset operation shall not cause any configuration parameters in the LIN Data Link device (UART) to be altered. Upon conclusion of the Reset operation, the slave device shall remain configured, and shall assume a state consistent with a power-on initialization, with the exception of the LIN configuration information. The slave shall also retain knowledge that it has undergone a reset operation, such that a positive response can be provided to the Master.

The LIN Slave device shall be able to respond to a $3D request frame which starts 20 ms after the start of the Break symbol of the $3C Targeted Reset command frame.

For those devices that are always in the power-on state, i.e., input only devices, a Reset Command may have no impact on the device state; however, this device shall still set the reset flag within the J2602-1 status byte and shall send a positive response.

A device may elect not to perform the requested Reset command based on the application program requirements. In this case, the device will respond with a Negative Response.

5.7.3.1 Command

Tx by Master	*Tx by Master*	*Tx by Master*	*Tx by Master*	*Tx by Master*	*Tx by Master*	*Tx by Master*	*Tx by Master*	*Tx by Master*
LIN ID	Data0 NAD	Data1 PCI	Data2 SID	Data3	Data4	Data5	Data6	Data7
$3C	NAD	$01	$B5	$FF	$FF	$FF	$FF	$FF

5.7.3.2 Positive Response

Tx by Master	*Tx by Slave*	*Tx by Slave*	*Tx by Slave*	*Tx by Slave*	*Tx by Slave*	*Tx by Slave*	*Tx by Slave*	*Tx by Slave*
LIN ID	Data0 NAD	Data1 PCI	Data2 RSID	Data3	Data4	Data5	Data6	Data7
$3D	NAD	$06	$F5	Supplier ID LSB	Supplier ID MSB	Function ID LSB	Function ID MSB	Variant ID

5.7.3.3 Negative Response

Tx by Master	*Tx by Slave*	*Tx by Slave*	*Tx by Slave*	*Tx by Slave*	*Tx by Slave*	*Tx by Slave*	*Tx by Slave*	*Tx by Slave*
LIN ID	Data0 NAD	Data1 PCI	Data2 RSID	Data3	Data4	Data5	Data6	Data7
$3D	NAD	$06	$7F	Supplier ID LSB	Supplier ID MSB	Function ID LSB	Function ID MSB	Variant ID

5.7.4 Broadcast Reset

Device behavior when a Broadcast Reset Command is received shall be the same as the behavior when a Targeted Reset Command is received.

5.7.4.1 Command

Tx by Master	*Tx by Master*	*Tx by Master*	*Tx by Master*	*Tx by Master*	*Tx by Master*	*Tx by Master*	*Tx by Master*	*Tx by Master*
LIN ID	Data0 NAD	Data1 PCI	Data2 SID	Data3	Data4	Data5	Data6	Data7
$3C	$7F	$01	$B5	$FF	$FF	$FF	$FF	$FF

5.7.4.2 Response

There shall be no positive or negative response to this command. Positive acknowledgment of this message can be ascertained via the reset bit in the J2602-1 Status Byte.

5.8 Message Format

5.8.1 Checksum

The enhanced checksum method shall be used for all protected identifiers except for those in the range of $3C to $3F.

5.8.2 Signal Consistency

The requirement that all "scalar signal writing or reading must be atomic" is not externally verifiable in all LIN implementations. Consequently this requirement is a guideline.

NOTE: Since there is a very close coupling between the J2602-1 Error signal and the APPINFO4 signal, implementers of J2602-1 slave nodes are encouraged to make every effort to publish these two status signals ”with consistency”, so that any/all error signaling between the Slave and the Master can be interpreted properly.

5.8.3 Signal Encoding Types

Since only two signal encoding types are allowed in the LIN 2.0 protocol specification (scalar and byte array), and there is an expectation to employ other standard signal encoding types commonly used in other serial data implementations, the following recognized signal encoding types shall be mapped into the two LIN signal encoding types in accordance with the following table. Appendix B Signal Encoding Types contains descriptions and examples of each of the signal encoding types listed.

TABLE 2 - COMMON SIGNAL ENCODING TYPE MAPPING TO LIN DEFINED DATA ENTITIES

Signal Encoding Type	Slot Type	Mapped to LIN Data Type	Notes
ASCII	ASC	8-bit Scalar	Most Significant Bit reserved and set to "0", ASCII codes 0 through 127
Binary Coded Decimal	BCD	4 bit Scalar per BCD character	Must be on nibble boundaries.
Boolean	BLN	1 bit Scalar	
Enumerated	ENM	N bit Scalar, in increments of 1 bit (1 – 16 bits)	
Signed Floating Point	SFP	4 byte - Byte Array	As defined in ANSI/IEEE Std 754-1985
Signed Numeric	SNM	8, 16, or 32 bit Byte Array	Unsigned with a negative offset number
Unsigned Numeric	UNM	N bit Scalar, increments of 8 bits (1 byte)	Not to exceed 16 bits (2 bytes)

Note that slave devices may transmit a maximum of 7 data bytes as the first data byte has been reserved for the J2602-1 Status Byte. The master device may transmit 8 data bytes, as it is not required to transmit the J2602-1 Status Byte. (See 5.8.6 J2602-1 Status Byte.)

5.8.4 Signal Management

Signals may belong to multiple frames, e.g., J2602-1 Status Byte. Signals may also be placed in the same frame multiple times, e.g., Broadcast messages.

5.8.5 Unused Bits in the Data Field (reference 2.3)

The length of each frame in bytes, used for LIN normal communications (not diagnostics or configuration), shall be defined for each LIN network, on a frame-by-frame basis. Any bits of a defined frame that are either not used or not defined shall be transmitted as recessive symbols. Padding of unused data bytes within LIN normal communications messages is not required. That is, normal communications messages may vary in length between 1 byte and 8 bytes.

5.8.6 J2602-1 Status Byte

This section defines a single standardized format for a J2602-1 Status Byte, which encompasses both LIN Protocol Error reporting and application specific information.

The J2602-1 Status Byte contains two enumerated bit fields, a 3 bit error field, and a 5 bit field for application specific information. Only the error field is defined, since it is the only one with the error and status states explicitly defined. The application information field is defined and implemented on a case-by-case basis, dependent on the requirements of each slave node application. The resulting application information field implementation is documented in the NCF for each slave node. This status byte shall be transmitted as the first byte of every slave transmission, where the identifier is in the range of $00 through $3B.

J2602-1 Status Byte

Bit 7 (msb)	6	5	4	3	2	1	Bit 0 (lsb)
ERR2	ERR1	ERR0	APINFO4	APINFO3	APINFO2	APINFO1	APIFNO0

5.8.6.1 Error Field Definition

ERR[2:0] – Error Field (Bits 7-5)

These bits report the four defined Error States that have been seen by the Slave node since an error state has been last reported to the Master node. The four states, as shown in Table 3 below, are “sticky”, which means that they are retained until they are successfully reported to the Master node in a message without any detected errors, after which they are automatically cleared. Only one state may be reported at a time. There is an inherent hierarchy to the states, with the highest latched state reported first. Bit 7 also serves the purpose of the Response_Error bit from LIN Rev. 2.0, 6.3.

TABLE 3 - ERR STATES

ERR2	ERR1	ERR0	Fault State	Priority
0	0	0	No Detected Fault	0 (lowest)
0	0	1	Reset	1
0	1	0	Reserved	2
0	1	1	Reserved	3
1	0	0	Data Error	4
1	0	1	Checksum Error	5
1	1	0	Byte Field Framing Error	6
1	1	1	ID Parity Error	7 (highest)

5.8.6.1.1 No Detected Fault

A slave node shall indicate this state whenever none of the other detectable fault states are active. This is the default state of the LIN device.

5.8.6.1.2 Reset

A slave node shall set this state upon interruption and resumption of power, after a watchdog timeout, or after receipt of a Reset Command. Note that for those devices that require configuration and store the configuration information in volatile memory, this state indicates that the device is currently unconfigured and requires configuration. For those devices that use non-volatile memory for configuration information storage, then the state indicates configuration is required the first time the part is powered on and only indicates a reset from then on.

5.8.6.1.3 Data Error

A slave or master node that is transmitting a bit on the bus shall also monitor the bus. A Data Error shall be detected when the bit or byte value that is received is different from the bit or byte value that is transmitted.

A slave node that is receiving shall detect a Data Error when the data in the fixed form Sync Byte is received incorrectly, i.e. is not $55. A slave node that performs autobauding shall detect this error but is not required to set the error bit.

5.8.6.1.4 Checksum Error

A Checksum Error shall be detected if the inverted modulo-256 sum over all received data bytes and the protected identifier (when using enhanced checksum) and the received checksum byte field does not result in $FF.

5.8.6.1.5 Byte Field Framing Error

The receiver shall detect a Byte Field Framing Error if the ninth bit after a valid start bit is dominant.

5.8.6.1.6 ID Parity Error

The receiver shall detect an ID Parity Error if the received ID parity (bits 6 & 7) does not match the ID parity calculated from the equations in 2.1.3 based on the received identifier (bits 0 – 5).

5.8.6.2 Application Information Field

APINFO[4:0] - Application Information Field (Bits 4-0)

5.8.6.2.1 APINFO4

APINFO4 shall be used to indicate when the application requires attention from the Master Device. This shall be indicated by setting the bit to “1”.

When a Reset state is indicated by the Error Field when this bit is “1” it shall indicate that the device needs to be configured.

The action taken by the Master when this bit is “1” and no Reset state is indicated shall be documented (e.g., in the NCF, datasheet, or Application Note, etc.) and may include loading a special schedule table which queries the Application as to its status.

5.8.6.2.2 APINFO[3:0]

The lower four bits of the application information field are not explicitly defined, since their structure will be dependent on the specific application and implementation of the slave node. This field may be implemented as discrete status bits, a state encoded bit field, or a combination of the two. The clearing mechanism used is also implementation dependent. The explicit definition of the Application Information Field shall be documented (e.g., in the NCF, datasheet, or Application Note, etc.).

5.8.6.2.3 No Application Information to Report

A value of zero (00000b) indicates that no application information currently needs to be reported. All other encodings are application specific and beyond the scope of J2602-1.

5.9 Message Types

5.9.1 Availability of Unconditional Frames

The requirement that subscribers of an unconditional frame shall make it available to the application is not externally verifiable in all LIN implementations. Consequently this requirement is a guideline.

5.9.2 Event Triggered Frames

Event Triggered frames shall not be utilized in J2602-1 compliant LIN networks.

5.9.2.1 Identifier Assignment (J2602-1 Requirement Resulting from Event Triggered Frame Anomaly)

One and only one slave shall be defined as the publisher to a single frame identifier.

5.9.3 Sporadic Frame

Messages of the Sporadic Frame type may be utilized within the J2602-1 environment, if appropriate. The Master Node shall be the only publisher of Sporadic Frames.

5.10 Calibration/Configuration Methods

This section defines two methods for calibrating/configuring slave nodes in a J2602-1 network using normal messaging. The Small Calibration/Configuration should only be used for calibrations/configurations of 160 bytes or less. This requires 1 message ID for every 8 bytes of data. The Large Calibration/Configuration method can be used for any size calibration/configurations and always requires two message IDs. The Large Calibration/Configuration requires additional complexity in the drivers.

5.10.1 Large Calibration/Configuration method

Calibrating/Configuring a Slave

The Mechanics of Calibrating/Configuring a LIN Slave

Calibrating/Configuring a LIN Slave is not a download. It uses a functional message to transfer data one time to the LIN slave. The LIN slave must retain this configuration information in EEPROM (or equivalent). In order to support a relatively large amount of configuration info (inclination/intrusion sensor) a multiplex message is used.

5.10.1.1 Calibrating/Configuration Data

Calibrating/Configuration information is limited to data that is downloaded when the LIN slave is first installed and **never changes**. If the information can change, it is deemed to be personalization information and must be transmitted to the slave using one or more normal (non-configuration) *LINxx* schedule tables.

In order to avoid limiting the amount of calibrating/configuration information to what can fit into one LIN frame, we have chosen to use a multiplex type transmit style. The LIN Master will transmit an index byte (*Calibrate/ConfigIndex*) plus seven data bytes (*Calibrate/ConfigData*). The master will always start with *Calibrate/ConfigIndex x* = 0 and then increment the *Calibrate/Config Index* every time it transmits the next set of *Calibrate/ConfigData*. When all of the data has been transmitted, the LIN Master will then go back and start transmitting from zero again. This is a simple algorithm in the LIN Master and allows different slaves to have differing amounts of *Calibrate/ConfigData*.

A LIN Slave will "know" that it has completely received all of the *Calibrate/ConfigData* when it sees *Calibrate/ConfigIndex x* = 0 a second time. At this point, the LIN Slave will need to write out the calibrate/config data to EEPROM. Once the write is completed the LIN Slave will lower the "I'm not configured" error flag (APINFO4 = 1).

5.10.1.2 Calibrate/Configuration Data Mapping

The table below explicitly lists how the multiplexed Calibrate/Config information signals are uniquely mapped in the LIN Slave.

TABLE 4 - MULTIPLEXED CALIBRATE/CONFIGINDEX/CALIBRATE/CONFIGDATA MAPPING TO UNIQUE INTERNAL LIN SLAVE CALIBRATE/CONFIG DATA.

Calibrate/ConfigIndex Value	**Calibrate/ConfigData Byte**	**LIN Slave Calibrate/Config Data**
0	Data0	LINCalibrate/Config[0]
0	Data1	LINCalibrate/Config[1]
0	Data2	LINCalibrate/Config[2]
0	Data3	LINCalibrate/Config[3]
0	Data4	LINCalibrate/Config[4]
0	Data5	LINCalibrate/Config[5]
0	Data6	LINCalibrate/Config[6]
1	Data0	LINCalibrate/Config[7]
1	Data1	LINCalibrate/Config[8]
1	Data2	LINCalibrate/Config[9]
1	Data3	LINCalibrate/Config[10]
1	Data4	LINCalibrate/Config[11]
1	Data5	LINCalibrate/Config[12]
1	Data6	LINCalibrate/Config[13]
2	Data0	LINCalibrate/Config[14]
2	Data1	LINCalibrate/Config[15]
2	Data2	LINCalibrate/Config[16]
2	Data3	LINCalibrate/Config[17]
2	Data4	LINCalibrate/Config[18]
2	Data5	LINCalibrate/Config[19]
2	Data6	LINCalibrate/Config[20]
3	Data0	LINCalibrate/Config[21]
3	Data1	LINCalibrate/Config[22]
3	Data2	LINCalibrate/Config[23]
3	Data3	LINCalibrate/Config[24]
3	Data4	LINCalibrate/Config[25]
3	Data5	LINCalibrate/Config[26]
3	Data6	LINCalibrate/Config[27]
•	•	•
•	•	•
•	•	•

TABLE 5 - CALIBRATE/CONFIGURING LIN SLAVES - SLAVE REQUIREMENTS

Rule
All LIN slaves shall store any configuration data in EEPROM/NVM. Configuration data will not be continuously transmitted during normal operation.
A slave shall infer that *Configuration Mode* has started when it first receives any of these signals: *Calibration/ConfigIndex* signal *Calibration/ConfigData* signal *PartNumIndex* signal *PartNumData* signal **Note:** Calibration/Configuration information is repeatedly transmitted until all slaves on a subnet report "Calibrated/Configured" or 4 seconds elapse (whichever is sooner).
Once a slave detects that the subnet as switched to *Calibration/Configuration Mode* then it shall immediately start reporting "Not Calibrated/Configured" until the next requirement is met. The LIN Slave must transmit the "Not Calibrated/Configured" status at least one time (even if the slave doesn't support configuration information).
The slave shall only indicate that it is configured after: All calibration/config information has been received All calibration/config information is saved in NVM/EEPROM
Note 1: The slave can detect that it has received all *Calibration/ConfigData* when the *Calibration/ConfigIndex* is zero the **second** time. **Note 2:** Even after a slave has been completely calibrated/configured, the slave may continue to receive the calibration/configuration information repeatedly. This allows other LIN slaves with more calibration/configuration information to receive their calibration/config data. It also allows the LIN Master to collect all LIN Slave part numbers (see next section).

5.10.1.3 Slave Part Number Reporting

LIN Slaves are given the opportunity to report their Part Number at the same time as the LIN Slave is placed into *Calibrate/ConfigurationMode*. See above for a description of when calibration/configuration is started.

The Mechanics of LIN Slave Part Number Reporting

Since we are limited to only 7 bytes of data (besides the LIN Status Byte), transmitting a 23 character part number will take several packets. So, like the configuration data, we use one byte to indicate an index and the rest of the bytes (6 others) are the data.

5.10.1.4 Part Number Sequencing

In order to allow the LIN Slave manufacturer to reduce cycle time on their production line, the LIN Slave is allowed to start with any *PartNumIndex*. They then cycle through the remaining indices. Once they repeat the very first PartNumIndex value, the LIN Master will consider the Part Number complete. All unused bytes must be set to 0 (NULL).

Notice that *PartNumIndex* is multiplied by 6 when placing the partial part number data in the correct location. Refer to *Slave* Part Number Mapping

Table 6 - Multiplexed PartNumIndex/ParNumData Mapping to LIN Master PartNumber Data for more information.

5.10.1.5 Master Part Number Retention

The LIN Master will erase these part numbers when the ignition is cycled in order to avoid misreporting LIN Slave part numbers due to replacing one in service.

5.10.1.6 Slave Part Number Mapping

TABLE 6 - MULTIPLEXED PARTNUMINDEX/PARNUMDATA MAPPING TO LIN MASTER PARTNUMBER DATA

PartNumIndex Value	PartNumData Byte	LIN Master PartNumber Data
0	Data0	PartNum[0]
0	Data1	PartNum[1]
0	Data2	PartNum[2]
0	Data3	PartNum[3]
0	Data4	PartNum[4]
0	Data5	PartNum[5]
1	Data0	PartNum[6]
1	Data1	PartNum[7]
1	Data2	PartNum[8]
1	Data3	PartNum[9]
1	Data4	PartNum[10]
1	Data5	PartNum[11]
2	Data0	PartNum[12]
2	Data1	PartNum[13]
2	Data2	PartNum[14]
2	Data3	PartNum[15]
2	Data4	PartNum[16]
2	Data5	PartNum[17]
3	Data0	PartNum[18]
3	Data1	PartNum[19]
3	Data2	PartNum[20]
3	Data3	PartNum[21]
3	Data4	PartNum[22]
3	Data5	PartNum[23]

TABLE 7 - PART NUMBER = ABCD-123456-EF EXAMPLE

[x]	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
Index Signal	**0**						**1**						**2**						**3**					
Data Signal	**0**	**1**	**2**	**3**	**4**	**5**	**0**	**1**	**2**	**3**	**4**	**5**	**0**	**1**	**2**	**3**	**4**	**5**	**0**	**1**	**2**	**3**	**4**	**5**
Contents	A	B	C	D	-	1	2	3	4	5	6	-	E	F	NUL	NUL	NUL	NUL	NUL	NUL	NUL	NUL	NUL	NUL

NOTE: In this example, the LIN slave could have only published PartNumIndex = 0, 1 and 2 (skipping 3).

LIN Slaves Part Number Reporting – Slave Requirements

TABLE 8 - LIN SLAVE PART NUMBER REPORTING - SLAVE REQUIREMENTS

Rule Num	Rule
DLPL_LIN_20_004	Once *Calibration/Configuration Mode* has started, the LIN Slave shall Set *PartNumIndex* = initial value (as defined by supplier but limited to the range of 0 to 3, inclusive). It shall also load *PartNumDataX* signals with the appropriate values.
DLPL_LIN_20_005	Every time *PartNumIndex* and *PartNumDataX* bytes are transmitted, the LIN Slave shall increment *PartNumIndex* modulo 4 and then load *PartNumDataX* bytes with the appropriate value.

5.10.2 Small Calibration/Configuration

5.10.2.1 Introduction

Depending on the complexity of the function that a slave device(s) is tasked to perform, vehicle/system/performance criteria data may be required specifically for the slave application to perform properly. A method of providing this application specific data for use within the LIN framework is defined in this section of this specification. Identifiers within the $00 to $37 range shall be used for all application configuration data transfers.

For devices requiring application configuration data, a variable length data volume can be loaded using multiple protected identifiers. This method shall be defined through a set of requirements under the Slave Application Calibration methodology. The maximum volume of data that can be transferred using this method is limited to 160 bytes. The maximum data allocation per slave is ultimately arrived at through an agreement between the Network designer and the Component Design/Release Engineer and shall be balanced between the number of slaves on the network, the total network normal operation messaging requirements and the specific Slave's functional requirements, but shall always be less than 160 bytes. This method requires a Calibration Part Number and a checksum value to be transferred to the slave during a single pass multi-frame schedule table. This method also requires the slave to have the ability to confirm that a compatible calibration has been loaded and that the calibration data has been loaded and locally stored correctly.

The requirements for this method are described in the following sections.

5.10.2.2 Slave Application Calibration File Structure in the Master and Related Requirements

5.10.2.2.1 Slave Application Calibration File Structure

The structure of the Slave Application Calibration data, in order of storage in the Master and in order of transmission on the LIN communications network shall be:

- Checksum (1 byte)
- Slave Application Calibration Part Number(4 bytes)
- Slave Application Calibration Data (byte 5 through byte n, where n – 5 = actual Slave Application Calibration Data length)
- $00 fill symbols (if necessary)

5.10.2.2.2 Slave Application Calibration Checksum

The first byte of the Slave Application Calibration shall contain a 1 byte checksum value.

5.10.2.2.3 Slave Application Calibration Part Number

The second through fifth bytes shall contain a 32 bit Unsigned Numeric representation (U_32) of the 8 digit decimal Slave Application Calibration Part Number. The Slave Application Calibration Part Number shall be the "released" Calibration Part Number (and, as well, the actual Part Number of the calibration). The following represents an example part number, the encoding, and location of the data within the Slave Application Calibration Data file.

Position (Byte)	2	3	4	5
SlaveApplication CalibrationPartNumber	16265965			
Hex value	00	F8	32	ED

5.10.2.2.3.1 Slave Application Calibration Data Contents

The contents and order of the Slave Application Calibration Data shall be determined by the slave supplier, with the agreement of the DRE.

5.10.2.2.3.2 $00 Fill Symbol

Any unused data space within a Slave Application Calibration Data block shall be filled after the Slave Application Calibration Data with the “$00” fill symbol in the Master memory.

5.10.2.2.4 File Length and Memory Allocation

5.10.2.2.4.1 Maximum Master Memory Allocation

An absolute maximum of 160 bytes total, of calibration memory shall be allocated in the Master for each Slave Application Calibration Data that the Master supports. Additional memory must also be allocated to maintain the Calibration File Header.

The exact length of the calibration data to be allocated for each slave shall be less than the maximum 160 bytes total when the additional following requirements are comprehended. Each calibration shall contain a 1 byte Checksum and a 4 byte Slave Application Calibration Part Number. Therefore, a maximum of 155 bytes of Slave Application Configuration Data can be contained in the 160 byte total block.

5.10.2.2.4.2 Master Memory Allocation for Slave Application Calibration Data Volume Less Than 160 Bytes

When the “maximum” Slave Application Calibration Data requirement for a given type of slave is less than 160 total bytes for all members of a slave family that may be allocated to an identical NAD, the “maximum” shall be allocated in the Master, instead of 160 bytes. The “maximum” allocation shall include sufficient additional space for the 1 byte Checksum and the 4 byte Slave Calibration Part Number.

5.10.2.2.4.3 Maximum Frame Consumption

Each member of a slave family that may be optionally allocated to a single NAD, shall be expected to consume the maximum number of frames allocated for the family.

5.10.2.2.4.4 $00 Fill Symbol

Any unused data space within a Slave Application Calibration Data block shall be filled after the Slave Application Calibration Data with the “$00” fill symbol in the Master memory.

For example, for a family of 2 Slave devices, where the Slave Application Calibration Data maximum length required 56 bytes of application data, the <u>maximum</u> calibration length would be 61 bytes. [56 (application data) + 1 (for the Checksum) + 4 (for the Part Number) = 61]. However, if the other Slave type required only 47 <u>maximum</u> (including the checksum and the Slave Application Calibration Part Number) bytes for its Slave Application Calibration, the remainder of the block of 61 bytes (14) shall be filled with $00 symbols.

5.10.2.2.5 Transmission of the Slave Application Calibration Data

5.10.2.2.5.1 Frame Transmit Order

The transmit order of Slave Application Calibration frames shall be specified in the Slave Component Technical Specification and the Slave specific NCF (and by incorporation, the Master LDF).

5.10.2.2.5.2 Transmit Schedule Table

There shall be 1 (one) single-pass schedule table constructed to provide for the loading of Slave Application Calibration Data to all members of a single slave family.

Therefore, (using the example provided above, for a slave with a maximum calibration length of 61 frames), the schedule table frame content will be composed of 7 frames of 8 byte length, followed by a single frame (frame 8) of 5 byte length. The same schedule table would be utilized for the slave requiring only 47 bytes. Therefore, the schedule table will also be composed of 7 frames of 8 byte length, and one frame of 5 byte length. During transmission of the last cited schedule table, the last 14 bytes (1 byte of frame 6, all of frame 7, and 5 bytes of frame 8) would be filled with the $00 symbol.

5.10.2.2.5.3 Calibration Load Invocation

A CPID shall be defined for the purpose of causing a Master to invoke all Slave Application Calibration schedule tables. The CPID may be imbedded in a Utility file (part 2) to cause the calibration load after the Master has been programmed or calibrated itself. As well, the CPID may be invoked through a tester request.

When the CPID request has been recognized by the Master, the Master Subnet Configuration List (MSCL) shall be utilized to determine which slaves are present in a particular configuration so that only those schedule tables which manage the calibration loading of slave that are present are invoked.

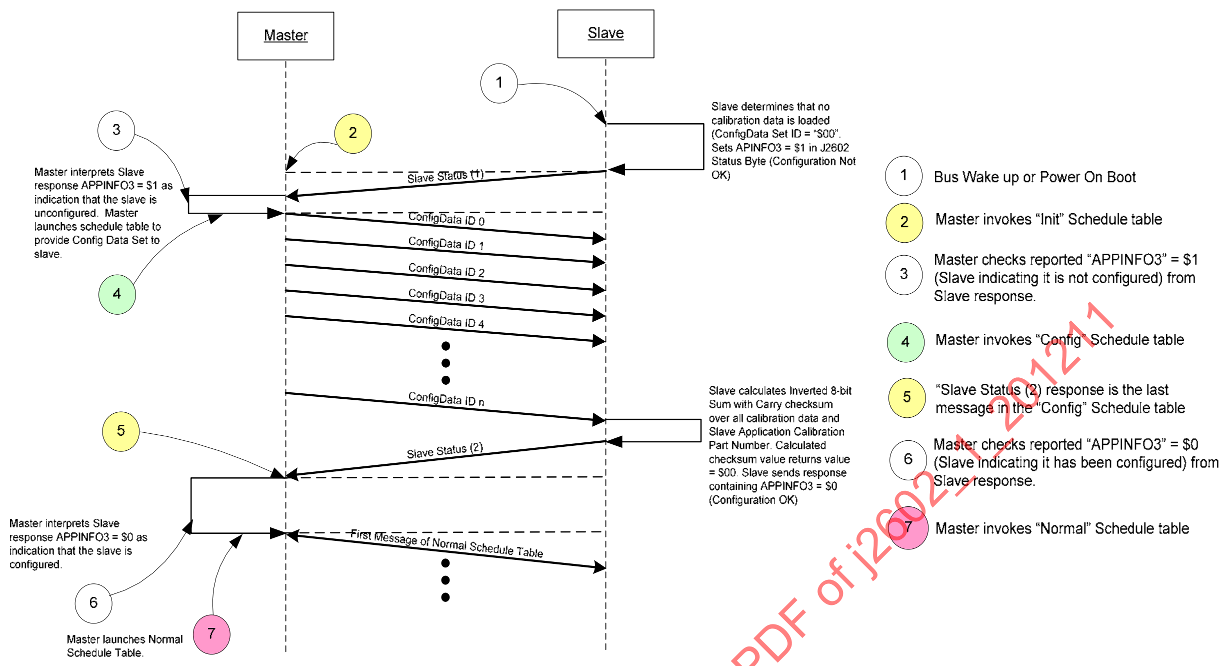
5.10.2.2.5.4 Calibration Loading during Normal Communications

The Master shall have the capability to respond to a slave device's recognition that its calibration data has either not been loaded or that it has been "lost". This automatic loading shall be prompted by the Master recognizing that the slave has indicated a loss of calibration data via the APINFO3 bit being set to a $1.

If the transfer was not successful, the Master shall make 1 (one) additional attempt to transmit the Slave Application Calibration data to the slave. In the event that the second attempt is also unsuccessful, the Master shall set a DTC consistent with [GGSE I-Document]. No further attempt to program the slave shall be made during that ignition cycle.

The following Figure 1 provides a Message Sequence Chart that depicts the significant events and factors which initiate a Slave Application Calibration sequence.

Slave Detected Calibration Not Loaded


Master
Slave
1
Slave determines that no calibration data is loaded (ConfigData Set ID = "$00". Sets APINFO3 = $1 in J2602 Status Byte (Configuration Not OK)
2
3
Master interprets Slave response APPINFO3 = $1 as indication that the slave is unconfigured. Master launches schedule table to provide Config Data Set to slave.
Slave Status (1)
ConfigData ID 0
ConfigData ID 1
ConfigData ID 2
ConfigData ID 3
ConfigData ID 4
4
ConfigData ID n
Slave calculates Inverted 8-bit Sum with Carry checksum over all calibration data and Slave Application Calibration Part Number. Calculated checksum value returns value = $00. Slave sends response containing APPINFO3 = $0 (Configuration OK)
5
Slave Status (2)
Master interprets Slave response APPINFO3 = $0 as indication that the slave is configured.
First Message of Normal Schedule Table
6
Master launches Normal Schedule Table.
7
1 Bus Wake up or Power On Boot
2 Master invokes "Init" Schedule table
3 Master checks reported "APPINFO3" = $1 (Slave indicating it is not configured) from Slave response.
4 Master invokes "Config" Schedule table
5 "Slave Status (2) response is the last message in the "Config" Schedule table
6 Master checks reported "APPINFO3" = $0 (Slave indicating it has been configured) from Slave response.
7 Master invokes "Normal" Schedule table

- ➢ The "Slave Status (1)" response message provides an indication that the target slave is alive and functioning, and that it recognizes that it is not configured ("APPINFO3 = $1" indicates that the slave's configuration criteria is not satisfied).
- ➢ The Master detects that the slave is not configured (APPINFO3 = $1"). Master launches the Config schedule table to load the ConfigData set.
- ➢ The "Slave Status (2)" response message provides an indication that the target slave has received and loaded the new "ConfigData" set. (Response message contains the APPINFO3 bit = $0, indicating that the slave's configuration criteria is satisfied).

MLRJr/LAP/LB 2006-11-30

FIGURE 1 - SLAVE DETECTED CALIBRATION NOT LOADED

NOTES:

1. The "Slave Status (1)" response message provides an indication that the target slave is alive and functioning, and that it recognizes that it is not configured ("APPINFO3 = $1" indicates that the slave's configuration criteria is not satisfied).
2. Master launches the Slave Application Calibration schedule table to load the ConfigData set.
3. The "Slave Status (2)" response message provides an indication that the slave has received and loaded the new "ConfigData" set. (Response message contains the APPINFO3 bit = $0, indicating that the slave's configuration criteria is satisfied).

5.10.2.2.6 Loading configuration data into slaves with volatile memory

The data flow for slaves with volatile memory only, is identical to the process as described in 5.10.2.2.5.4. Because a slave with only volatile memory will be un-configured upon start-up initialization after each power-up, therefore the slave will signal that it is un-configured in its first response message. This will trigger the same mechanism as with slaves with non-volatile memory requesting their configuration data. Once the status information in the response frame is received from the slave, it can be guaranteed that the slave in question is also ready to receive data. The Master will then start a configuration download for that particular slave.

5.10.2.3 Slave Node Application Configuration Introduction

Depending on the complexity of the function that a slave device(s) is tasked to perform, vehicle/system/performance criteria data may be required specifically for the slave application to perform properly. A method of providing this application specific data for use within the LIN framework is defined in this section of this specification. Identifiers within the $00 to $37 range shall be used for all application data transfers.

For devices requiring application configuration data, a variable length data volume can be loaded using multiple protected identifiers. This method shall be defined through a set of requirements under the Slave Application Calibration methodology. The maximum volume of data that can be transferred using this method is limited to 160[1] bytes. The maximum data allocation per slave is ultimately arrived at through an agreement between the Network designer and the Component Design/Release Engineer and shall be balanced between the number of slaves on the network, the total network normal operation messaging requirements and the specific Slave's functional requirements. This method requires a Calibration Part Number and a checksum value to be transferred to the slave during a single pass multi-frame schedule table. This method also requires the slave to have the ability to confirm that a compatible calibration has been loaded and that the calibration data has been loaded and locally stored correctly. The requirements for this method are described in the following sections.

5.10.2.3.1 Slave Application Calibration File Structure

The structure of the Slave Application Calibration shall be:

Checksum (1 byte)

Slave Application Calibration Part Number(4 bytes)

Slave Application Calibration Data (byte 5 through byte n, where n – 5 = actual Slave Application Calibration Data length)

$00 fill symbols (if necessary)

5.10.2.3.2 Slave Application Calibration Checksum

The first byte of the Slave Application Calibration shall contain a 1 byte checksum value.

Slave Application Calibration Part Number

5.10.2.3.3 Slave Application Calibration Part Number

The second through fifth bytes shall contain a 32 bit Unsigned Numeric representation (U_32) of the 8 digit decimal Slave Application Calibration Part Number. The Slave Application Calibration Part Number shall be the "released" Calibration Part Number (and, as well, the actual Part Number of the calibration). The following represents an example part number, the encoding, and location of the data within the Slave Application Calibration Data file.

Position (Byte)	2	3	4	5
SlaveApplication CalibrationPartNumber	16265965			
Hex value	00	F8	32	ED

[1] This maximum of 160 bytes includes additional required data parameters, and does not reflect the total amount of application configuration data that can be transferred to a slave.

5.10.2.3.3.1 Slave Application Calibration Data Contents

The contents and order of the Slave Application Calibration Data shall be determined by the slave supplier, with the agreement of the DRE.

5.10.2.3.3.2 $00 Fill Symbol

Any unused data space within a Slave Application Calibration Data block shall be filled after the Slave Application Calibration Data with the "$00" fill symbol in the Master memory.

5.10.2.3.4 Slave Application Configuration Data Volume Requirement Declaration

The actual length of the Slave Application Calibration Data shall be identified in the Slave component CTS and shall be accommodated based on an agreement between the SSLT, the LIN Team, and Component Supplier, with the agreement being communicated and confirmed to the LIN Team via the LIN Questionnaire and the Component Technical Specification (CTS). The absolute maximum length of the Slave Application Calibration Data for any slave device shall not exceed 160 bytes. This 160 byte length shall be of the structure defined above (and shall include the 1 byte checksum and the 4 byte Calibration Part Number.

When the "maximum" Slave Application Calibration Data requirement for a given type of slave is less than 160 total bytes for all members of a slave family that may be allocated to an identical NAD, the "maximum" shall be allocated in the Master, instead of 160 bytes. The "maximum" calibration allocation shall include sufficient additional space for the 1 byte Checksum and the 4 byte Slave Calibration Part Number.

5.10.2.3.4.1 $00 Fill Symbol

Any unused data space within a Slave Application Calibration Data block shall be filled after the Slave Application Calibration Data with the "$00" fill symbol in the Master memory.

For example, for a family of 2 Slave devices, where the Slave Application Calibration Data maximum length required 56 bytes of application data, the maximum calibration length would be 61 bytes. [56 (application data) + 1 (for the Checksum) + 4 (for the Part Number) = 61]. However, if the other Slave type required only 47 maximum (including the checksum and the Slave Application Calibration Part Number) bytes for its Slave Application Calibration, the remainder of the block of 61 bytes (14) shall be filled with $00 symbols.

5.10.2.3.5 Transmission of the Slave Application Calibration Data

5.10.2.3.5.1 Transmit Order

The transmit order of Slave Application Calibration frames shall be specified in the Slave Component Technical Specification and the Slave specific NCF (and by incorporation, the Master LDF).

5.10.2.3.5.2 Transmit Schedule Table

There shall be 1 (one) single-pass schedule table constructed to provide for the loading of Slave Application Calibration Data to all members of a single slave family.

Therefore, (using the example provided above, for a slave with a maximum calibration length of 61 frames), the schedule table frame content will be composed of 7 frames of 8 byte length, followed by a single frame (frame 8) of 5 byte length. The same schedule table would be utilized for the slave requiring only 47 bytes. Therefore, the schedule table will also be composed of 7 frames of 8 byte length, and one frame of 5 byte length. During transmission of the last cited schedule table, the last 14 bytes (1 byte of frame 6, all of frame 7, and 5 bytes of frame 8) would be filled with the $00 symbol.

5.10.2.3.5.3 Maximum Frame Consumption

Each member of a slave family that may be allocated to a single NAD shall be expected to consume the maximum number of frames allocated for the family.

5.10.2.3.5.4 Slave Reception Rate Specification

The supplier shall be expected to identify the anticipated minimum and maximum reception periods for each frame of the Slave Application Calibration Data block in the LIN Questionnaire and on subsequent submittals of the NCF files for the slave.

Nota Bene: The slave supplier should take care to not underestimate the minimum reception periods for these frames. The slave is expected to receive the frame and handle (buffer or write to non-volatile memory) each of the 8 bytes before the next frame is transmitted. The frames may well be transmitted at the minimum reception period identified.

5.10.2.3.6 Slave Application Calibration Data Management Responsibilities

5.10.2.3.6.1 Calibration Compatibility Confirmation

The slave device is responsible for confirming that a compatible calibration has been loaded.

5.10.2.3.6.2 Calibration Compatibility Confirmation Methods

The Slave shall have the responsibility to confirm that the calibration data is compatible with its application requirements. Confirmation methods may include, but are not limited to, data length checking, specific data value plausibility, specific variable locations within the file, checksum over the loaded file. The type of confirmation checking to be accomplished shall be based on an agreement between the SSLT, DRE, and Component Supplier, and documented in the CTS.

5.10.2.3.6.3 Checksum Calculation

This confirmation shall be minimally accomplished by performing a checksum calculation over the contents of the application configuration data image to ensure that the checksum result identically matches the application configuration data checksum value (received in the calibration file). This confirmation shall be accomplished upon every initialization and reset operation (both running reset and commanded reset) of the slave.

5.10.2.3.6.4 Slave Reporting of Application Calibration Loading Errors

In the event the Slave device determines that it has not been satisfactorily loaded with the calibration data, the Slave shall communicate this deficiency to the Master through the APINFO3 Calibration Status bit in any normal mode response message.

5.10.2.3.6.5 Slave Reporting of Application Calibration Data Compatibility Errors

In the event the Slave device determines that it has been loaded, but the calibration is either not compatible or is not appropriate for the application, the Slave shall communicate this deficiency to the Master through the Slave specific Configuration Comparison Error Flag Status bit (xxxConfigCompErrFlg Status bit, where xxx is the Slave specific acronym) in any normal mode response message.

5.10.2.3.7 Additional Requirements

5.10.2.3.7.1 Slave Application Calibration Part Number Reporting

The Slave Application Calibration Part Number, an 8 digit part number, shall be defined as a normal mode LIN signal to be transmitted by the Slave when requested by the Master. The Master shall store the received part number as a DID parameter. (The Slave shall be able to transmit the part number signal upon Master request through normal mode messaging, and the Master shall make the parameter available in response to a diagnostic communications DID request). The Slave shall be delivered with the Slave Application Calibration Part Number value set to $00000000.

5.10.2.3.7.2 Slave Part Number

Any slave device which requires the loading of Slave Application Calibration Data shall also maintain, in memory, its assigned 8 digit part number. This part number shall be loaded (programmed) prior to delivery to the OEM. This stored value shall be known as the Slave Part Number.

This Slave Part Number must be specified in the Slave Component Technical Specification, by the responsible specification and release authority. This number may take the form of either the Base Model Part Number or the End Model Part Number.

The Slave Part Number, an 8 digit part number, shall be defined as a normal mode LIN signal to be transmitted by the Slave when requested by the Master. The Master shall store the received part number as a DID parameter. (The Slave shall be able to transmit the part number signal upon Master request through normal mode messaging, and the Master shall make the parameter available in response to a diagnostic communications DID request).

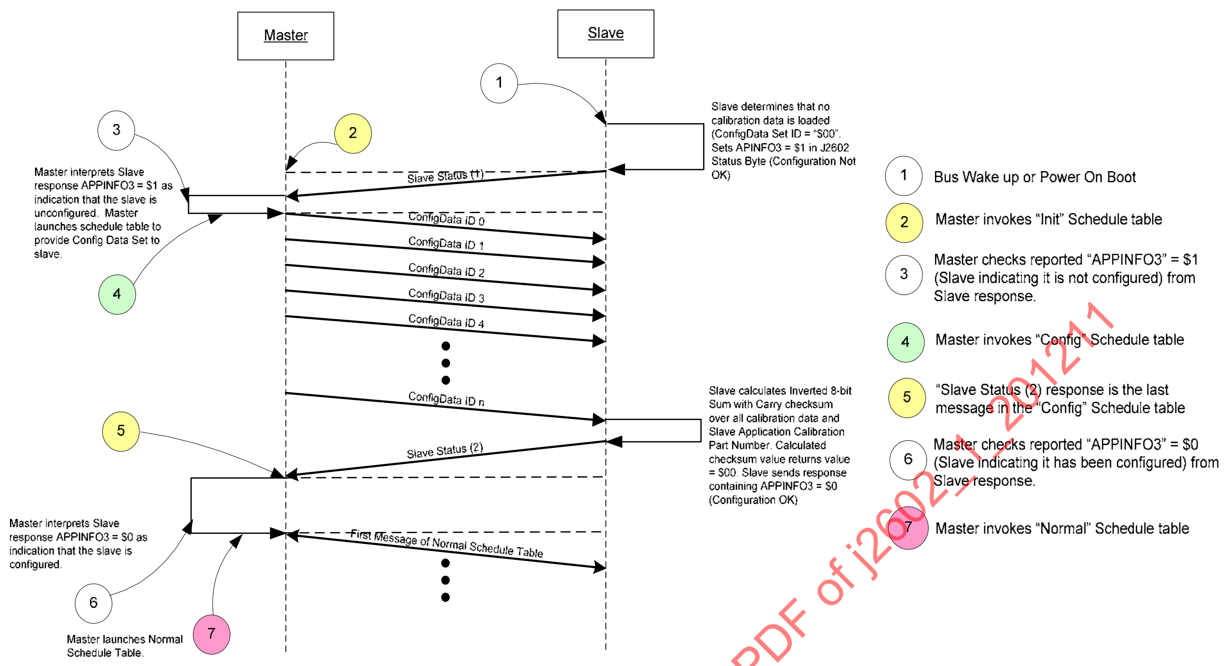
5.10.2.3.7.3 Application Configuration Status Bit

When the Master receives the APINFO3 bit set to a \$1 value, it shall interpret this response as an indication that the slave has either not had the appropriate Slave Application Calibration Data loaded, or that the slave has “lost” this data. When this condition is reported, the Master shall initiate a configuration download sequence.

Presuming that the transfer was successful, the Master shall then confirm that the slave has accurately received the Configuration Data by requesting that the slave respond with a single response frame to confirm that the slave has reset the APINFO3 Configuration Status bit (to a \$0 value, indicating that the slave is now configured). A Message Sequence Diagram representing this sequence is depicted in the below Figure 2.

If the transfer was not successful, the Master shall make 1 (one) additional attempt to configure the slave. In the event that the second attempt is also unsuccessful, the Master shall set a DTC consistent with [GGSE I-Document]. No further attempt to program the slave shall be made during that ignition cycle.

Slave Detected Calibration Not Loaded


Master
Slave
1
Slave determines that no calibration data is loaded (ConfigData Set ID = "$00". Sets APINFO3 = $1 in J2602 Status Byte (Configuration Not OK)
2
3
Master interprets Slave response APPINFO3 = $1 as indication that the slave is unconfigured. Master launches schedule table to provide Config Data Set to slave.
Slave Status (1)
ConfigData ID 0
ConfigData ID 1
ConfigData ID 2
ConfigData ID 3
ConfigData ID 4
4
ConfigData ID n
Slave calculates Inverted 8-bit Sum with Carry checksum over all calibration data and Slave Application Calibration Part Number. Calculated checksum value returns value = $00. Slave sends response containing APPINFO3 = $0 (Configuration OK)
5
Slave Status (2)
Master interprets Slave response APPINFO3 = $0 as indication that the slave is configured.
First Message of Normal Schedule Table
6
7
Master launches Normal Schedule Table.
1 Bus Wake up or Power On Boot
2 Master invokes "Init" Schedule table
3 Master checks reported "APPINFO3" = $1 (Slave indicating it is not configured) from Slave response.
4 Master invokes "Config" Schedule table
5 "Slave Status (2) response is the last message in the "Config" Schedule table
6 Master checks reported "APPINFO3" = $0 (Slave indicating it has been configured) from Slave response.
7 Master invokes "Normal" Schedule table

- The "Slave Status (1)" response message provides an indication that the target slave is alive and functioning, and that it recognizes that it is not configured ("APPINFO3 = $1" indicates that the slave's configuration criteria is not satisfied).
- The Master detects that the slave is not configured (APPINFO3 = $1"). Master launches the Config schedule table to load the ConfigData set.
- The "Slave Status (2)" response message provides an indication that the target slave has received and loaded the new "ConfigData" set. (Response message contains the APPINFO3 bit = $0, indicating that the slave's configuration criteria is satisfied).

MLRJr/LAP/LB 2006-11-30

FIGURE 2 - SLAVE DETECTED CALIBRATION NOT LOADED

NOTES:

The "Slave Status (1)" response message provides an indication that the target slave is alive and functioning, and that it recognizes that it is not configured ("APPINFO3 = $1" indicates that the slave's configuration criteria is not satisfied).

Master launches the Slave Application Calibration schedule table to load the ConfigData set.

The "Slave Status (2)" response message provides an indication that the slave has received and loaded the new "ConfigData" set. (Response message contains the APPINFO3 bit = $0, indicating that the slave's configuration criteria is satisfied).

5.10.2.3.7.4 Loading configuration data into slaves with volatile memory

The data flow for slaves with volatile memory only, is identical to the process as described in 5.10.2.3.7.3. Because a slave with only volatile memory will be un-configured upon start-up initialization after each power-up, therefore the slave will signal that it is un-configured in its first response message. This will trigger the same mechanism as with slaves with non-volatile memory requesting their configuration data. Once the status information in the response frame is received from the slave, it can be guaranteed that the slave in question is also ready to receive data. That is, the Master shall use the un-configured status indication from all slaves as the trigger criteria to begin a configuration loading sequence.

6. J2602-1 API REQUIREMENTS

6.1 Master Node Configuration API

For Master implementations, the Node Configuration API (Section 3 of the LIN 2.0 API specification) is mandatory if any Node Configuration commands will be utilized on the network. If Node Configuration is not used, these API calls are optional.

The Node Configuration API only applies to Master implementations.

6.2 Diagnostic Transport Layer API

For Master or Slave implementations, the Diagnostic Transport Layer API (Section 4 of the LIN 2.0 API specification) is optional.

Diagnostic information is supplied via the J2602-1 Status Byte.

6.3 Additional API Requirements

Please see SAE J2602-3 for additional details on Master and Slave API Requirements.

7. J2602-1 BUS OPERATION

The physical layer is responsible for providing a method of transferring digital data symbols (1's and 0's) to the communication medium. The physical layer interface is a single wire, vehicle battery referenced bus, with low side voltage drive.

7.1 Normal Communication Mode and Transmission Rate

Transmission bit rate in the normal communication mode is 10.417 Kbits/s, this results in a nominal bit time of 96 μs. In the normal transmission mode, transmitters with controlled waveform fall and undershoot times should be used. Waveform rising edge control is recommended to assure that high frequency components are minimized at the beginning of the upward voltage slope. The remaining rise time occurs after the bus is inactive with drivers off and is determined by the RC time constant of the total bus load.

7.2 Sleep/Wake Mode

7.2.1 Wake-up (reference 5.1)

The node may continue to issue wake-up requests until the Master node responds by transmitting a frame identifier. Following a minimum delay of 1.5 s and a maximum delay of 3.0 s (after the third wake-up attempt), the slave device may initiate one additional sequence of three wake-up requests separated by a minimum of 150 ms and a maximum of 300 ms. The cycle is concluded following the sixth wake-up transmission. The minimum elapsed time for one complete wake-up cycle, as described, is 2.1 s.

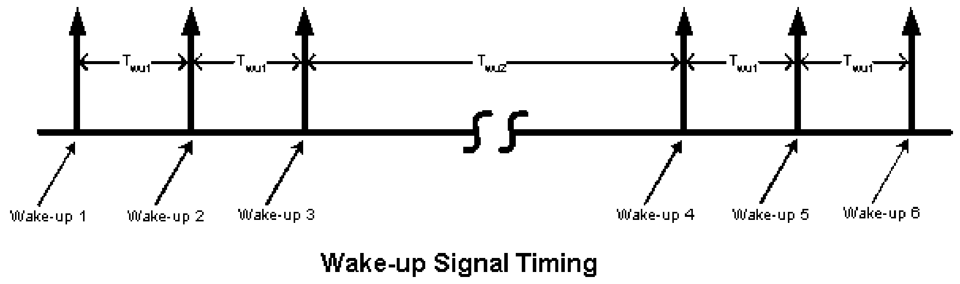


FIGURE 3 - WAKE-UP SIGNAL TIMING

Parameter	Minimum	Maximum
Wake-up Pulse	250 µs	5 ms
T_{wu1}	150 ms	300 ms
T_{wu2}	1.5 s	3 s

If no response (valid identifier transmission) is produced by the Master node in response to the wake-up requests, the slave node shall default to a network sleep state. In the event the slave node detects another local wake-up input after the sleep default, it may again attempt to wake the Master through a new wake-up signal cycle. There is no limit to the number of wake-up cycles that may be attempted.

7.2.2 Go to Sleep (reference 5.2)

The Master node must transmit an explicit “Go To Sleep” command to the network, prior to ceasing to transmit.

7.2.2.1 Slave Node Sleep

All slave node(s) shall interpret a cessation of all message traffic for 4 s on a given network, without receiving an explicit “Go To Sleep command” as a failure condition of the Master node or the physical layer. When this condition occurs, the slave node(s) shall assume a default state that may include sleep and/or low power consumption state or application defined functionality.

7.3 LIN Controller Clock Tolerance

7.3.1 Master-Slave Communication

TABLE 9 - MASTER-SLAVE COMMUNICATION CLOCK TOLERANCE

Device Type	Clock Tolerance	Notes
Master	±0.5%	Initial Tolerance + Divide Error
Slave	±1.5%	From the nominal bit time with a fixed clock → Initial Tolerance + Divide Error when synchronized
Slave (autobauding)	±2.0%	From the master bit time → Initial Tolerance + Divide Error when synchronized

7.3.2 Slave-Slave Communication

This mode is currently not supported and is not recommended. This is for information purposes only.

TABLE 10 - MASTER-SLAVE COMMUNICATION CLOCK TOLERANCE

Device Type	Clock Tolerance	Notes
Master	±0.5%	Initial Tolerance + Divide Error
Slaves communicating with Master only.	±1.5%	Initial Tolerance + Divide Error when synchronized
Slaves communicating with Master only. (autobauding)	±2.0%	From the nominal bit time with a fixed clock → Initial Tolerance + Divide Error when synchronized
Slaves communicating with Slave	±1.0%	From the nominal bit time → Initial Tolerance + Divide Error when synchronized (NOTE: This requires autobauding slaves to synchronize within ±0.5% of the master.)

7.4 Bus Electrical Parameters

This section describes the bus electrical voltage level parameters required by devices that drive and receive signals on the LIN bus.

7.4.1 LIN Bus Signals and Loading Requirements

TABLE 11 - LIN BUS SIGNALS AND LOADING REQUIREMENTS

Parameter	Symbol	Min.	Typ.	Max.	Units
ECU Battery Voltage Input [1]	$V_{batt\ ECU}$	8		18	V
Maximum ECU Battery Voltage Input for no Damage [7]	$V_{batt\ max\ ECU}$	-13		40	V
IC Battery Voltage Input	$V_{batt\ IC}$	7		18	V
Maximum IC Battery Voltage Input for no Damage	$V_{batt\ max\ IC}$	0		34	V
Output High Voltage	V_{oh}	0.8 $V_{batt\ IC}$		$V_{batt\ IC}$	V
High Voltage (Recessive) Input Threshold	V_{ih}	0.47 $V_{batt\ IC}$		0.6 $V_{batt\ IC}$	V
Output Low Voltage	V_{ol}	0.0		0.2 $V_{batt\ IC}$	V
Low Voltage (Dominant) Input Threshold	V_{il}	0.4 $V_{batt\ IC}$		0.53 $V_{batt\ IC}$	V
Input Threshold Hysteresis (V_{ih} - V_{il}) [6]	V_{HYS}	0.07 $V_{batt\ IC}$		0.175 $V_{batt\ IC}$	V
Ground Offset Voltage	$V_{g\ off}$	—		0.1 $V_{batt\ ECU}$	V
Battery ECU Offset Voltage	$V_{b\ off}$	—		0.1 $V_{batt\ ECU}$	V
LIN bus to Ground Isolation Resistance	$V_{L\text{-}G\ Iso}$	500 K			Ω
Network Total Resistance	R_{tl}	537		1081	Ω
Device Bus Leakage Current V_{batt} Disconnected	$I_{leak\ batt}$	-23		23	µA
Device Bus Leakage Current Ground Disconnected	$I_{leak\ gnd}$	-100		100	µA
Slave Device Capacitance [3]	C_{slave}	90	220	272	pF
Master Device Capacitance [3]	C_{master}	90	680	2450	pF
Network Total Capacitance [4]	C_{tl}	926		9310	pF
Bus Wiring Capacitance	C_w			100	pF/m
Network Time Constant [2]	$\tau_{network}$	1.0		5.0	µs
Master Termination Resistance	R_M	900	1000	1100	Ω
Slave Termination Resistance	R_S	20,000	30,000	60,000	Ω
$t_{REC(MAX)} - t_{DOM(MIN)}$ [5]	$T_{r\text{-}d\ max}$	—		15.9	µs
$t_{DOM(MAX)} - t_{REC(MIN)}$ [5]	$T_{d\text{-}r\ max}$	—		17.28	µs
Total Network length connecting all ECU nodes	bus length	—		40 (see Table 12 for application limits)	meters
Number of system nodes		2		16	

1. V_{batt} is measured at the ECU input power pins. All voltages are referenced to the local ECU ground.
2. The normal mode network time constant ($\tau_{network}$) is the product of R_{tl} and C_{ltl}. The network time constant incorporates the bus wiring capacitance. The minimum value is selected to limit radiated emissions. The maximum value is selected to ensure proper communication under all communication modes and is the absolute maximum allowed under normal operating conditions. The system should be designed to have a time constant no larger than 5.3 µs under an error condition at a slave node. This should be considered when determining the fusing for the vehicle. Not all combinations of R and C are possible. Only those combinations of R and C, and bus length, and PCB trace capacitance, etc. are possible that meet the specified network time constant.
3. The ECU capacitance includes the actual load capacitor as well as the PCB trace capacitance, connector capacitance, etc. The ECU capacitance is the room temperature capacitance and does not include temperature effects. The capacitance shall not change by more than 10 % over the entire operating temperature range. The Network total capacitance includes the capacitors placed on the ECUs as well as the capacitance of the bus wires.
4. Equations for converting from the Duty Cycle in the LIN Physical Layer Spec to the times in this table can be found in Appendix C.
5. Input Threshold Hysteresis (V_{ih} - V_{il}) cannot be less than 0.0
6. See 7.12 Operating Battery Power Voltage Range for details.

7.5 LIN Data Link (UART) Requirements

Any device (e.g., UART, SCI, software, etc.) chosen to implement a J2602-1 LIN data link interface shall meet all requirements in this section.

7.5.1 Sample Point

The device shall sample the data within the window specified in Figure 4. No samples shall be taken outside the specified window for the purpose of determining the value of the data on the bus. The device may take three samples and use the majority to determine the data, or it may take a single sample to determine the data.

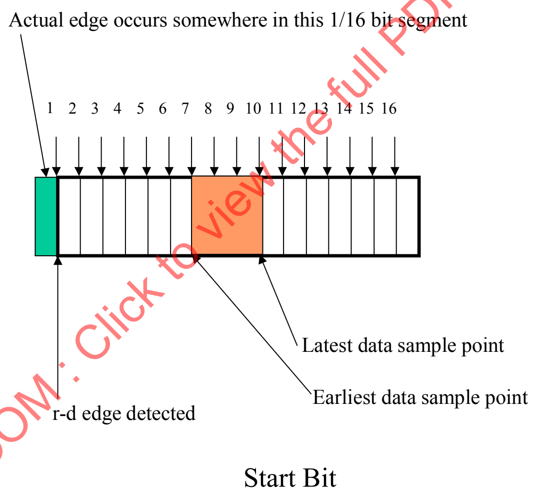


FIGURE 4 - BIT SAMPLE TIMING

7.5.2 Synchronization

1. The device shall be able to synchronize within 1/16 of a bit time.

2. The device shall only synchronize on recessive to dominant edges.

3. The device shall always synchronize on the recessive to dominant edge of the Start bit.

7.5.3 Transmit Message Buffering

Double buffering shall not be used for transmit messages to ensure that the requirement for error detection in 5.4.2 is not violated.

7.6 LIN ECU Requirements

7.6.1 ECU Circuit Requirements

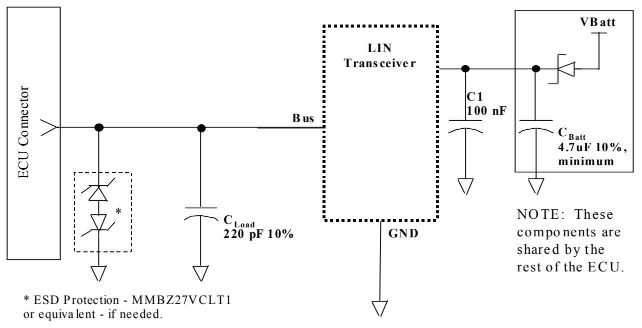

ECU Connector
Bus
LIN Transceiver
GND
C1 100 nF
VBatt
C_{Batt} 4.7uF 10%, minimum
C_{Load} 220 pF 10%
*
NOTE: These components are shared by the rest of the ECU.
* ESD Protection - MMBZ27VCLT1 or equivalent - if needed.

FIGURE 5 - TYPICAL LIN SLAVE BUS INTERFACE

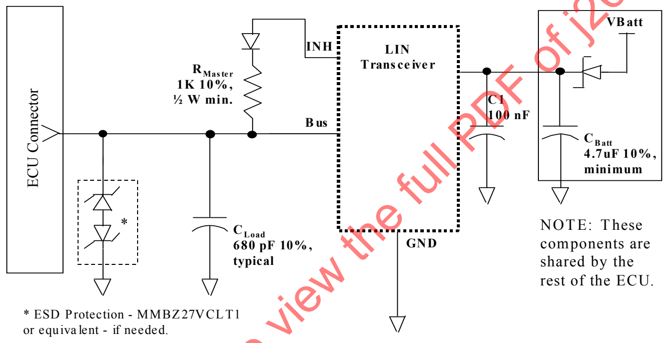

ECU Connector
R_{Master} 1K 10%, ½ W min.
INH
Bus
LIN Transceiver
GND
C1 100 nF
VBatt
C_{Batt} 4.7uF 10%, minimum
C_{Load} 680 pF 10%, typical
*
NOTE: These components are shared by the rest of the ECU.
* ESD Protection - MMBZ27VCLT1 or equivalent - if needed.

FIGURE 6 - TYPICAL LIN MASTER BUS INTERFACE

7.6.1.1 Master Node Resistor

The Master node resistor shall be as specified in Table 11 and Figure 6. It shall have a minimum power rating of 0.5 W, or 0.36 W at the maximum specified operating temperature.

This assumes that the transceiver limits the current to the load pin to a maximum of 20 mA. In the event the transceiver will provide more current to the resistor, the maximum power dissipation shall be the maximum of 0.78 W and that calculated by the following equation: P (watts) = I^2 * R where I is the maximum current in Amps and R is 900 Ω.

7.6.1.2 Master Node Pull-Up Reverse Blocking Diode

The maximum voltage drop in the Master node pull-up reverse blocking diode is 1.0 V at the maximum current; the current limited by the transceiver or 30 mA, whichever is smaller.

The minimum power dissipation of the diode is determined by the current through the diode. It shall be at least 20 mW if the transceiver limits the current to the load pin to a maximum of 20 mA; otherwise, it shall be 30 mW.

7.6.1.3 Master Node Capacitance

The Master node load capacitor shall be as specified in Table 11 and Figure 6 with voltage rating appropriate to a maximum loaded network under worst case environmental and electrical conditions.

7.6.1.4 Slave Node Capacitance

The Slave node load capacitor shall be as specified in Table 11 with voltage rating appropriate to a maximum loaded network under worst case environmental and electrical conditions.

7.6.1.5 ESD Transient Suppressor

If necessary, a circuit element such as a transorb (back-to-back zener) or a varistor device may be added to the network in one or more places to provide ESD protection. However, when these devices are used they may add capacitance or introduce voltage and/or temperature variability to the network time constant. When such devices are used the device load capacitor shall be reduced by an amount equivalent to the capacitance of the ESD transient suppressor. See Figures 3 and 4.

7.6.2 Board Layout Requirements

1. All grounding of the LIN transceiver and the filter capacitors shall be made to ECU signal ground.

2. C1 and C_{LOAD} shall be monolythic ceramic chip capacitors. (Ceramic chip capacitors have low ESR and high self-resonant frequencies.)

3. A ground plane is required under the transceiver chip on the same side of the board as the component.

4. Transceiver shall be located as close to edge connector as possible. Other IC's are not permitted between edge connector and the transceiver.

5. The LIN bus circuit between the edge connector and transceiver shall be as short as possible. Guard tracks are required for all LIN bus and Tx and Rx circuits.

6. All guard tracks shall be at least 0.5 mm wide and grounded at least every 10 mm. No signals shall be routed between the guard track and the LIN bus trace.

7.7 Network Topology

7.7.1 Loss of ECU Ground

The loss of ground by any single slave ECU, with or without an accompanying loss of V_{batt}, shall not cause any bus voltage offset that will disable normal communications (see $I_{leak\ gnd}$ in Table 11).

7.7.2 Loss of ECU Battery

The loss of battery by any single slave ECU, with or without an accompanying loss of ground, shall not cause any bus voltage offset that will disable normal communications (see $I_{leak\ batt}$ in Table 11).

7.7.3 Bus Electrical Load Distribution

Each Slave ECU shall contain a slave device capacitance load and a slave termination resistive load.

The total network equivalent minimum resistance (R_{tl}) and maximum capacitance (C_{tl}) shall comply with the totals specified in Table 11.

7.7.4 Bus Wiring Topology Configurations

The data link physical medium wiring mechanization can be implemented in any of the following ways:

1. Vehicles may be wired in a ring, a star, or a combination of both. Note that ECUs that are intended for use across multiple platforms may have two connector pins as shown below to allow a ring connection.

2. The pins of the ECU when applied in a ring, shall be adjacent and in the same connector, shorted together as close to the connector as possible, and share EMC and/or loading components.

3. If in a star configuration, the ECU requires only one pin for LIN.

The topology of the LIN bus shall be determined for each vehicle platform based on the vehicle's fault tolerance, serviceability, and bus length requirements. A second bus wire connector terminal for the LIN circuit at each ECU allows for implementation of a ring configuration, although it is not required that both terminals be used. A ring, star, or combination of ring and star configuration is acceptable as long as all other LIN wiring requirements are met. Illustrations of the LIN topologies are shown in Figures 7, 8, 9, and 10 below.

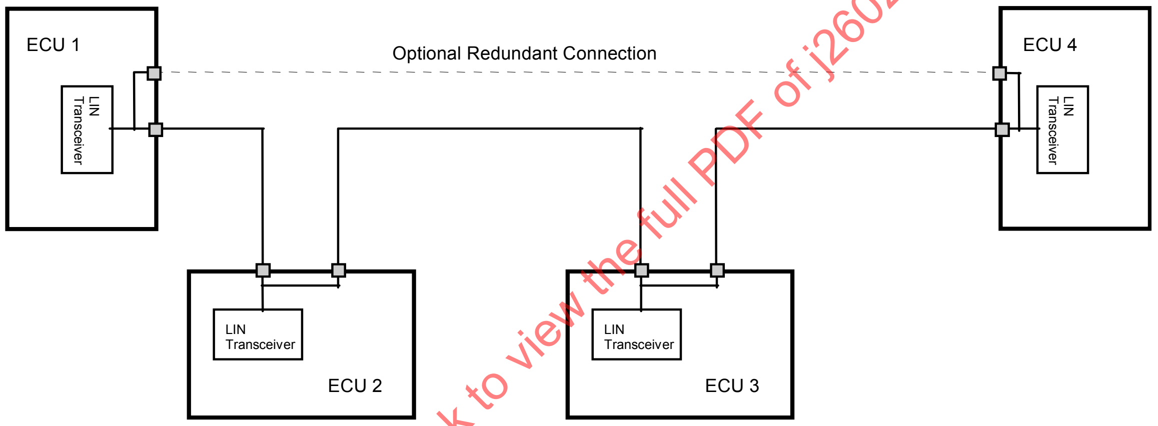


FIGURE 7 - LIN RING TOPOLOGY

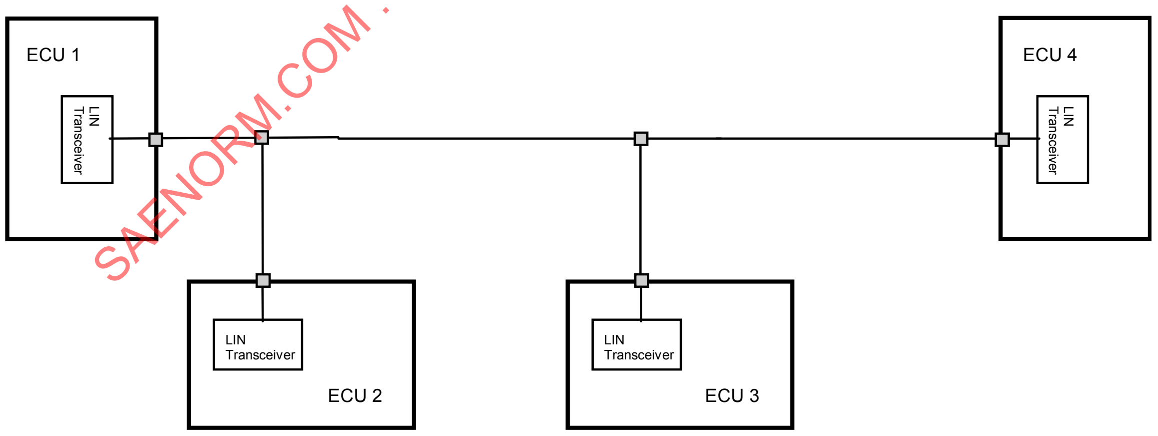


FIGURE 8 - LIN LINEAR TOPOLOGY

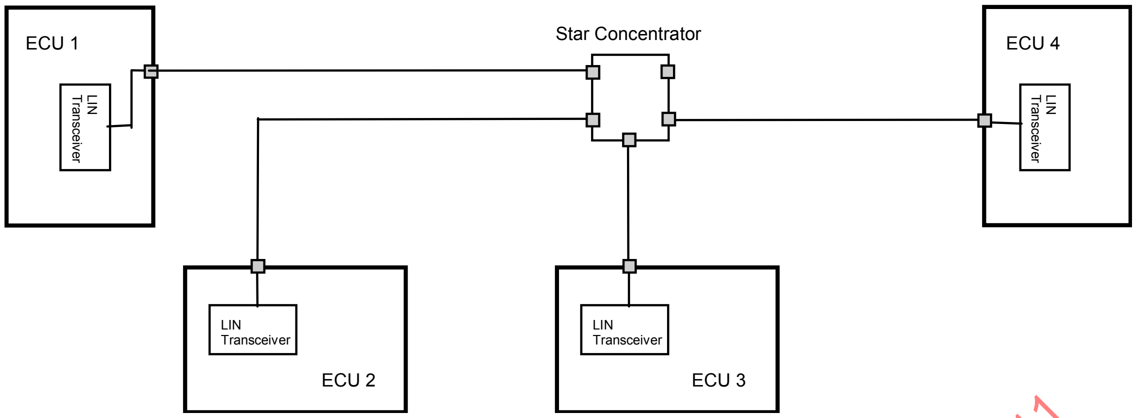


FIGURE 9 - LIN STAR TOPOLOGY

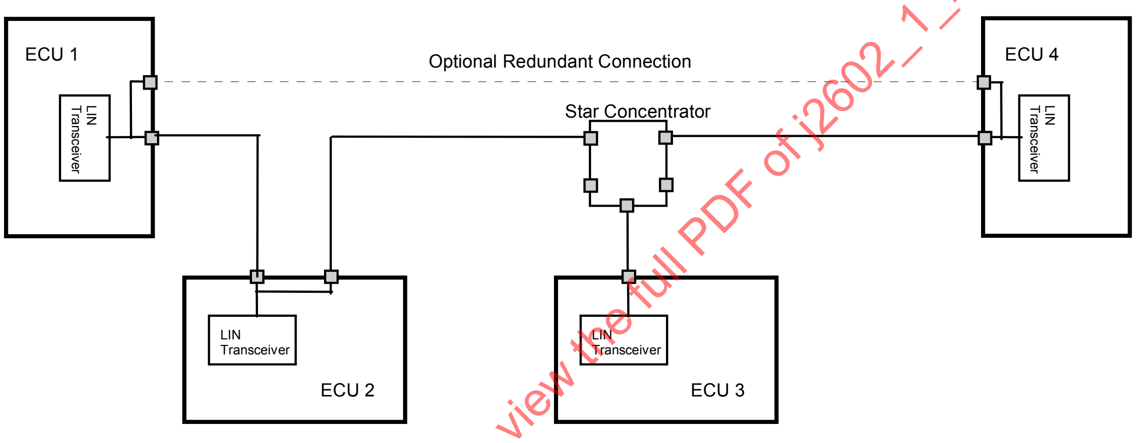


FIGURE 10 - LIN COMBINATION RING AND STAR TOPOLOGY

7.7.5 Bus Wiring Constraints

The vehicle network wiring and ECU system shall meet the following constraints:

1. The total bus wiring capacitance shall not cause the network time constant to be exceeded. (see Table 11). The maximum bus length allowed is determined by the number of nodes in the network system and their R-C characteristics.

2. There shall be no more than 40 m between any two network system ECU nodes.

The following table uses the maximum network time constant to determine the maximum wire length that can be used to connect a given number of ECUs. This maximum wire length depends on the Master Node Capacitance and the number of slave nodes. In calculating these numbers it was assumed that all resistors are at the maximum allowed values and capacitors (10%) are at their maximum tolerance, that the capacitance of the wire was 100 pF/m, that the slaves all have nominal capacitances of 220 pF and that an additional 30 pF of capacitance are added due to PCB traces and Connectors.

Table 12 shows the maximum wire length allowed on the vehicle based on the number of nodes on the vehicle and their characteristics. The legend in Table 12 gives the capacitance in the Master Node.

The maximum vehicle wire length is calculated using the following equation:

$$\text{Wire} = [\tau_{\text{Network}} / (R_s / \text{\# slave nodes} \parallel R_m) - C_m - C_s * \text{\# slave nodes}] / \text{wire(C/m)} \qquad \text{(Eq. 1)}$$