SAE International™ SURFACE VEHICLE RECOMMENDED PRACTICE

SAE J2602-3 JAN2010

Issued 2010-01

File Structures for a Node Capability File (NCF)

RATIONALE

The scope of this document covers the outcome of an analysis, performed by the SAE J2602-3 Task Force, of the Application Program Interface, Node Configuration File, and LIN Description File portions of the LIN 2.0 specification. The purpose of the analysis was to identify and resolve requirements in these specific specifications that are ambiguous, conflicting, or optional and to document a method for accommodating SAE J2602 node and network specific data elements in the LIN 2.0 NCF/LDF file structure(s). Development and adoption of this recommended practice will allow the design and development tools to consistently identify, extract, interpret and utilize the required information relative to specific network operational characteristics.

FOREWORD

SAE J2602-3 provides additional requirements that are not present in LIN 2.0 (e.g., NCF syntax, LDF syntax, data definitions, etc.). This document is to be utilized by Tier 1 suppliers of components, software developers, tool developers, vehicle or OEM system engineers, and vehicle or OEM communications engineers.

INTRODUCTION

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 SAE J2602 networks will be further specified in this document.

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 systems, design and development tools, software, ECUs and media shall be designed. This will assure consistent and unambiguous serial data communication among all connected devices regardless of supplier.

This document may be referenced by any vehicle OEM component technical specification that describes any given ECU in which the single wire data link controller and physical layer interface is located.

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 Purpose

The goal of SAE J2602 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 provides additional requirements that are not present in LIN 2.0 (e.g., fault tolerant operation, network topology, etc.).

2. REFERENCES

2.1 Applicable Publications

The following publications form a part of this specification to the extent specified herein. Unless otherwise specified, the latest issue of SAE publications shall apply. In the event of a conflict between this document and one of the referenced documents this document takes precedence.

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

LIN Specification package version 2.1 dated November 24th, 2006 available at www.lin-subbus.org.

See Appendix A for list of supplier documents/devices.

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 J1213-1 Glossary of Vehicle Networks for Multiplexing and Data Communications

SAE J1930 Electrical/Electronic Systems Diagnostic Terms, Definitions, Abbreviations, and Acronyms

SAE J2602-1 LIN Network for Vehicle Applications

2.1.2 ISO Publication

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

ISO 7498 Open systems interconnection (OSI)

3. DEFINITIONS AND ACRONYMS

3.1 Definitions

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 Must

The word "**Must**" is used to indicate that a binding requirement exists on components or devices which are outside the scope of this specification.

3.1.8 Physical Layer

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

3.1.9 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.

3.1.10 Publisher

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

3.1.11 Radiated Emissions

The energy that radiates from the LIN physical layer.

3.1.12 Radiated Immunity

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

3.1.13 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.

3.1.14 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.

3.1.15 Shall

The word "**Shall**" is to be used in the following ways:

a. To state a binding requirement on the LIN interfaces which comprise the ECU, which is verifiable by external manipulation and/or observation of an input or output.

b. To state a binding requirement upon an ECU's NCF that is verifiable through a review of the document.

3.1.16 Should

The word "**Should**" is used to denote a preference or desired conformance.

3.1.17 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.

3.1.18 Subscriber

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

3.1.19 Will

The word "**Will**" is used to state an immutable law of physics.

3.2 Acronyms

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/sec – Thousands of data bits per second

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
SAE – SAE International
SCI – Serial Communication Interface
UART – Universal Asynchronous Receiver/Transmitter

4. COMPATIBILITY WITH LIN 2.0

4.1 String Variable Type Node Attributes

Node Attributes “LIN Protocol Version” in both the LDF and NCF is defined as a string variable type instead of as a “real” or “integer” value.

4.2 Optional Node Attributes

The following Node Attributes are optional for SAE J2602 Slaves:

- P2_min in NCF and LDF (default = 0 mS)
- ST_min in NCF and LDF (default = 0 mS)
- Configurable Frames in LDF (only needed if NAD = 0x6s)
- response_tolerance (default = 40 %)
- wakeup_time (default = 100 ms)
- poweron_time (default = 100 ms)
- Support_SID in NCF
- Max Message Length in NCF (default = 0)

4.3 Frames Defined in NCF

Frames defined in NCF must have signals, i.e., the definition of an empty frame is not allowed.

4.4 Message_ID in NCF

Message_ID in NCF is mandatory for SAE J2602.

4.5 Frame Length Definition

Frame length definition in NCF is mandatory for SAE J2602.

4.6 Definition of Encoding Bytes

The definition of encoding bytes is also allowed for byte array signals, i.e., it is possible to define physical and logical encodings for signals larger than 16 bits.

4.7 Error Signal

The Error Signal must be a 3 bit signal and should be defined in the NCF using the keyword error_response (as it is in the LDF).

4.8 Free Text in NCF

Free Text in NCF should be defined as a char string. Note that this change corresponds to the definition in the LIN 2.1 specification.

4.9 Additional Master Parameters

Additional Master parameters have been added to the LDF definition for max. header length and Master response tolerance.

4.10 Additional Node Attributes

Additional node attributes have been added to NCF and LDF definitions for Slave response tolerance, wakeup time and power on time.

4.11 Dynamic Frames

Dynamic frames shall not be used for SAE J2602 networks.

4.12 Event-Triggered Frames

Event-triggered frames shall not be used for SAE J2602 networks.

4.13 Sporadic Frames

Sporadic frames shall only be published by the Master node.

5. LIN APPLICATION PROGRAM INTERFACE REQUIREMENTS

5.1 LIN 2.0 Node Configuration API

The Node Configuration API is not required for a SAE J2602 only network, but can be optionally included in the Master node if needed, i.e., SAE J2602 does not prohibit the use of read by identifier or reconfigurations provided in this API. Note that slave devices never need this API.

In the event that dynamic node configuration operations are required in a SAE J2602 network, then the LIN 2.0 Node Configuration API shall be used.

5.2 LIN 2.0 Diagnostic API

The LIN 2.0 Diagnostics API is not required for a SAE J2602 only network, but can be optionally included in the Master node if needed, i.e., SAE J2602 does not prohibit the use of the diagnostic functionality provided in this API. Note that slave devices never need this API.

In the event that diagnostic operations are required in a SAE J2602 network, then the LIN 2.0 Diagnostics API shall be used.

5.3 SAE J2602 Status Byte

5.3.1 Signal Consistency

SAE J2602-3 requires that all bytes in a byte-array are read and write consistent with each other.

5.3.2 Functional Partitioning

A “Communications Handler” is required only when a Slave device is implemented in software.

If required, the Slave Communications Handler is responsible for writing the SAE J2602 Error Signal. The Slave application is responsible for writing the Application Information bits.

The SAE J2602 Status Byte is defined as a 3 bit error signal and a 5 bit application info signal. The error signal shall not be written by the application but only by the LIN handler. In the case of a target reset, the application shall request the handler to reset the error signal.

5.4 Signal Management

5.4.1 Same Signal in Multiple Frames

SAE J2602 Master and Slave devices can send the same signal in multiple frames. (Note: a signal has only one publisher. Each slave has its OWN status byte signals that must be part of every frame that a slave publishes.)

5.4.2 Unique Name

Each signal used in a SAE J2602 network shall be uniquely named.

5.4.3 Broadcast and Command Frames

A J2602 Master may publish a signal both in SAE J2602 broadcast frame(s) and command frame(s).

5.4.4 SAE J2602 Status Byte

All signals in the SAE J2602 Status Byte shall be the same in each frame that a single slave publishes – except Targeted Reset response (section 5.7.3 of SAE J2602-1). (Note that neither the positive nor negative Targeted Reset response frame contains a SAE J2602-1 Status byte.)

5.5 Changing Active Schedule Table Clarification

The handler will allow the application to change schedule tables only at the end of a frame slot. A frame slot is defined in the LIN 2.0 specification as:

- The time period reserved for the transfer of a specific frame on the LIN bus. Corresponds to one entry in the schedule table.

While not stated in this definition, the value of the delay parameter that accompanies a specific frame declaration in a schedule table entry is considered to be the end of a frame slot by the protocol. In actuality, the “delay” parameter is the total time allotted for transmission of the specified frame and delay, which must be a multiple or sub-multiple of the tick time specified for the handler. Therefore, schedule tables can only be changed after the transmission of the specified frame + the expiration of the delay time specified in the Schedule Table entry.

6. LIN NODE CAPABILITY LANGUAGE SPECIFICATION

6.1 Generic NCF

If broadcast messages are used, then separate NCFs shall be created for each node even if they are otherwise identical. Furthermore the DNN must be known prior to creating the NCF. Changing the DNN would mean changing the NCF.

Example:

```
Node_capability_file;
<LIN_language_version> ::= LIN_language_version = "J2602_3_1.0" ;
```

6.2 Message Identifier Clarification in the NCF

The definition of Message Identifiers shall be mandatory for SAE J2602 Slaves and corresponds to the LIN frame IDs for the DNN of 0. According to the DNNs used in the network, the LDF design tool must correct the frame IDs defined. (default = 0 mS)

6.3 General Node Attributes

6.3.1 General Definition

For SAE J2602 NCF files, the keyword “LIN_protocol_version“ shall be used. (This is a tool/file editor and consistency checker issue and corresponds to the LIN2.1 specification.)

The following example, then illustrates the correct form of the General section as defined in this document.

```
<general_definition> ::=
general {
      LIN_protocol_version = <protocol_version> ; mandatory e.g. “J2602_1_1.0”
      supplier = <supplier_id> ; mandatory e.g. 0x0001, (if unavailable use 0x7FFF)
      function = <function_id> ; mandatory e.g. 0x0001
      variant = <variant_id> ; mandatory e.g. 0x01
      bitrate = <bitrate_definition> ; // mandatory e.g. 10417
      (response_tolerance = <integer> %;) // optional e.g. 30 %, default = 40 %
}
```

6.3.2 Diagnostic Definition

```
<diagnostic_definition> ::=
diagnostic {
      NAD = <integer> ; // mandatory e.g. 0x61 i.e. DNN $1
      (P2_min = <integer> ms ;) // optional for J2602, default = 0
      (ST_min = <integer> ms ;) // optional for J2602, default = 0
      support_sid
      {
         0xb5; // mandatory for J2602-Slaves this is the targeted reset as defined in J2602-1.
      }
      (max_message_length = <integer> ;) // optional for J2602-Slaves, default = 0
}
```

6.3.3 Frame Definition

```
<frame_definition> ::=
frames {
       [<frame_kind> <frame_name> {
            length = <integer> ; // mandatory
            (min_period = <integer> ms ;) // optional
            (max_period = <integer> ms ;) // optional
            (frame_id = <identifier> ;) // optional
            signals {
                  [<signal_name> {
                        init_value = <init_value_scalar> | <init_value_array> ; // mandatory
                        size = <integer> ; // mandatory
                        offset = <integer> ; // mandatory
                        (encoding_name = <identifier> ;) // optional
                  }]
            }
     }]
}
```

6.3.4 Encoding Definition

The Encoding Definition is only required if an encoding_name is identified for a signal in the Frame Definition. See Appendix A for details on encoding types.

```
<encoding> ::=
encoding <encoding_name> {
[<logical_value> |
<physical_range> |
<bcd_value> |
<ascii_value>]
}
```

6.3.5 Status Management

The 3-bit SAE J2602 Communications Error signal shall be allowed as a valid entry in the “response_error” field.

```
<status_management> :==
status_management {
        response_error = <published_signal> ; // mandatory, 3-bit signal allowed
        (fault_state_signals = <identifier> ([, <identifier>]) ;) // optional
}
```

Example – may be cut and paste to the NCF example later.

```
{
// byte 0
   LINStatus
   {
      init_value = 1;
      size = 3;
      offset = 5;
      encoding LINStatus
      {
         logical_value, 0, "No Detected Fault" ;
         logical_value, 1, "Reset" ;
         logical_value, 2, "Reserved" ;
         logical_value, 3, "Reserved" ;
         logical_value, 4, "Tx Bit Error" ;
         logical_value, 5, "Checksum Error" ;
         logical_value, 6, "Byte Field Framing Error" ;
         logical_value, 7, "ID Parity Error" ;
      }
   }
}

status_management
{
   Response_error = LINStatus;
}
```

7. LIN DESCRIPTION FILE DESCRIPTION

7.1 Protocol, Language, and Conformance Declarations

<LIN_protocol_version_def> ::= LIN_protocol_version = "J2602_1_1.0";

<LIN_language_version_def> ::= LIN_language_version = "J2602_3_1.0" ;

7.2 Master Frame Header and Response Tolerances

The available bandwidth of a LIN or SAE J2602 network can be further improved by using a header and response tolerances less than 40%. In this case, a design tool will need to take account of both tolerances in order to calculate the correct slot delay times for a schedule.

NOTE: The header length can be varied in the range 34 to 48 bit times by changing the sync break length, sync delimiter length, and interbyte spaces (space between sync and protected identifier). It is therefore more convenient to define the header tolerance as a maximum header length in bit times.

In order to communicate to tools, which max. header length and response length was used by the design tool, two new optional definitions have been added to node definition of the LIN Description File, for example:

```
< Node_def > ::=
Nodes
{
        Master: <node_name>, <time_base> ms, <jitter> ms
               (, <max_header_length> bits, <response_tolerance> %); // default = 48 bits, 40 %
        Slaves: <node_name>([, <node_name>]);
}
```

7.2.1 Node Attributes – Frame Tolerence

In the LIN Protocol Specification, the frame tolerance is defined as being 40%. This frame tolerance can be split up into a header tolerance and a response tolerance.

The response tolerance includes both the response space between header and response and the interbyte spaces of the response. Since the interbyte spaces will usually be zero, the response tolerance implicity defines the minimum interframe space. This is the minimum time needed by a Slave, before it can start responding to a header.

ASIC implementations of SAE J2602 nodes will not need the 40% response tolerance typically reserved for micro-based nodes. The definition of response tolerances between 0 and 40% has been defined. Micro based slaves that are capable of faster responses may also indicate a tolerance of less than 40%.

```
< Node_attributes_def > ::=
Node_attributes
{
        [<node_name> {
               LIN_protocol = <protocol_version> ; // mandatory
               configured_NAD = <diag_address> ; // mandatory
               product_id = <supplier_id>, <function_id>, <variant> ; // mandatory
               response_error = <signal_name> ; // mandatory
               (P2_min = <real_or_interger> ms ; ) //optional
               (ST_min = <real_or_interger> ms ; ) //optional
               (configurable_frames {
                      [ <frame_name> = <message_id> ; ]} ) //optional
               (response_tolerance = <real_or_integer> %;) // optional e.g. 30 %, default = 40 %
               (wakeup_time = <integer> ms;) // optional e.g. 50 ms, default = 100 ms
               (poweron_time = <integer> ms;) // optional e.g. 90 ms %, default = 100 ms
        }]
}
```

8. NOTES

8.1 Marginal Indicia

A change bar (l) located in the left margin is for the convenience of the user in locating areas where technical revisions, not editorial changes, have been made to the previous issue of this document. An (R) symbol to the left of the document title indicates a complete revision of the document, including technical revisions. Change bars and (R) are not used in original publications, nor in documents that contain editorial changes only.

PREPARED BY THE SAE J2602-3 SUBCOMMITTEE
OF THE SAE VEHICLE ARCHITECTURE FOR DATA COMMUNICATIONS COMMITTEE

APPENDIX A - SAE J2602 TO LIN 2.0 SIGNAL ENCODING AND DATA TYPES

```
encoding <encoding_name> {
        [<logical_value> |
        <physical_range> |
        <bcd_value>      |
        <ascii_value>]
}

<ascii_value> ::= ascii_value ;

<bcd_value> ::= bcd_value ;

<logical_value> ::= logical_value, <signal_value> (,<text_info>);

<physical_range>::= physical_value,<min_value>,<max_value>,<scale>, <offset> (,<text_info>);
```

ASCII (ASC)

Use <ascii_value>

Boolean (BLN)

```
logical_value, 0, "False";
logical_value, 1, "True";
```

Enumerated (ENM)

Example: Days of the Week

```
logical_value, 0x0, "Invalid Day of the Week";
logical_value, 0x1, "Sunday";
logical_value, 0x2, "Monday";
logical_value, 0x3, "Tuesday";
logical_value, 0x4, "Wednesday";
logical_value, 0x5, "Thursday";
logical_value, 0x6, "Friday";
logical_value, 0x7, "Saturday";
```

Binary Coded Decimal (BCD)

Use <bcd_value>

Signed Numeric (SNM)

physical_value, -R[$2^{(L-1)}$], R[$2^{(L-1)}$ - 1], 1, 0;

Unsigned Numeric (UNM)

physical_value, Min(N), Max(N), R, offset (,<text_info>);

Where E = R * N + Offset

E = engineering units = physical_value

R = resolution = scale

N = computer units = raw value

Signed Floating Point [Scientific Notation] (SFP)

No equivalent LIN 2.0 encoding. Use a byte-array.

Byte Array (BA)

For 4 byte signals, in order to guarantee inter-signal consistency, use a byte array.

For 32-bit Signed floating point, use a 4-byte byte array.