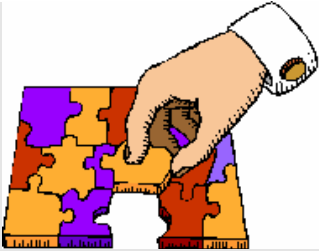


# LIN communication server for embedded ECU

## Software components



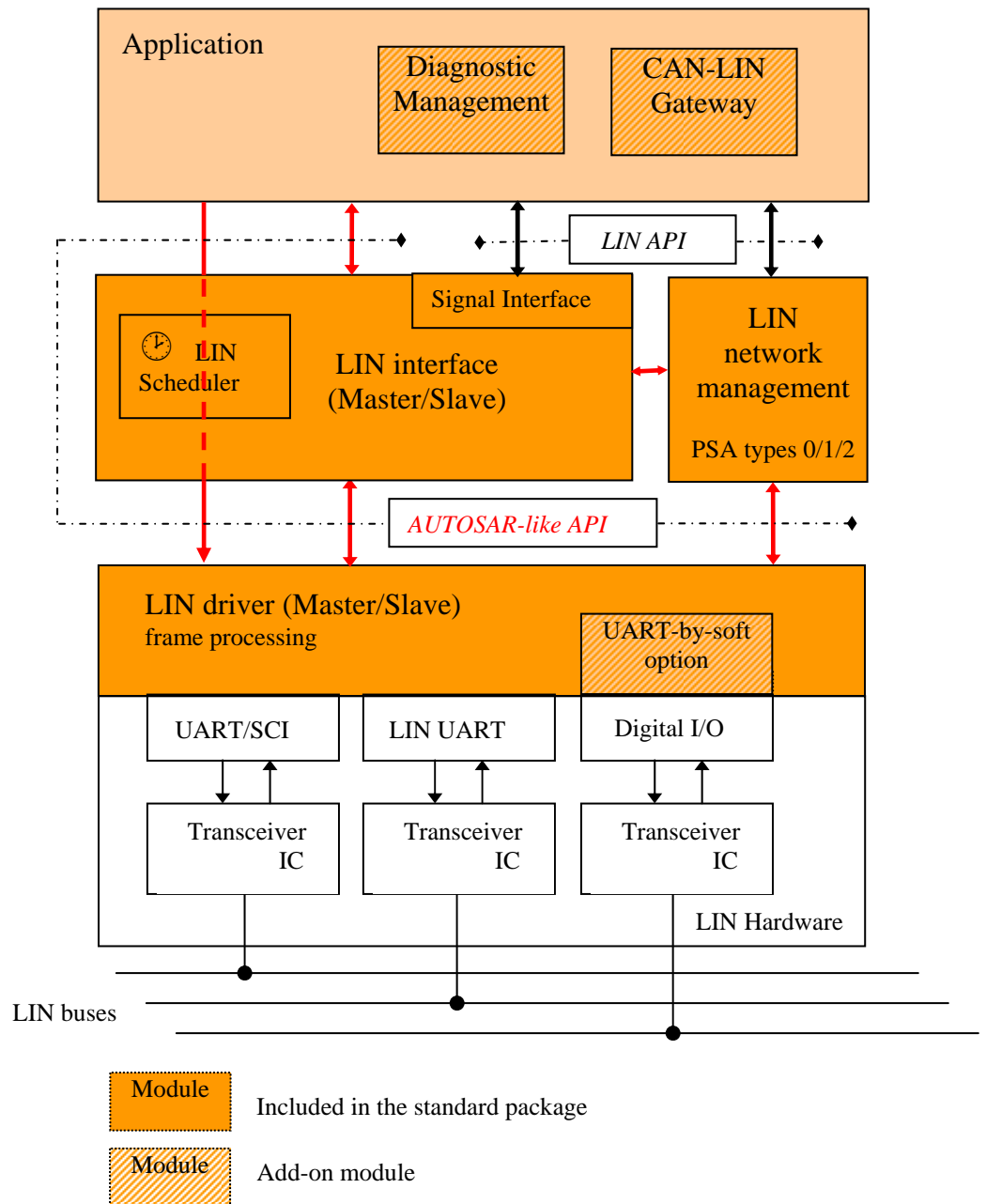
### Standard package :

- Re-usable LIN protocol stack
- Based on the LIN specification rev1.3 (roadmap for LIN2.0) and the PSA LIN specifications.
- Multichannel handling, may be configured as Master or Slave for each bus
- Support for PSA and non-PSA networks
- Written in ANSI 'C' language compliant to the MISRA rules
- AUTOSAR-compatible Programming Interfaces (API)
- Inter-operable with NSI CAN stacks (for CAN-LIN gateway option)
- Optimized for optimal use of microcontroller resources – with or without LIN UART.
- Already validated stacks

### Add-on modules :

- Slave diagnostic session management
- CAN-LIN gateway (CAN stack is a prerequisite)
- UART-by-soft : UART emulation using digital I/O (performance requirement to be checked)

Introduction of the LIN buses in the car embedded networks architectures led the car manufacturer to specify communication rules for ECU connected to vehicle domain LIN networks beyond the LIN consortium specifications in order to guaranty interoperability between all ECU. The LIN communication server has been developed by NSI whose know-how in the integration of communication layers is well recognized, especially to answer these requirements. The use of this standard stack allows to save development costs and to reduce the ECU delivery time. The software components are delivered as source files written in ANSI C language to be linked with the application. The modular architecture of the stack helps easy and error-free integration into the application. Moreover, the LIN interface provides AUTOSAR-compatible programming interfaces making it possible to re-use it in the AUTOSAR environment with almost no modification on application side.



# LIN communication server for embedded electronic control units

## Functional specifications

<b>LIN Driver – Hardware abstraction layer</b>	<b>Master</b>	<b>Slave</b>
✓ Hardware initialization: microcontroller UART & IO ports, LIN transceivers	X	X
✓ Service for header transmission	X	
✓ Service for response transmission	X	X
✓ Services for sleep/wake-up management & transmission of wake-up signals on the bus	X	X
✓ Header transmission confirmation <sup>(1)</sup>	X	
✓ Response transmission confirmation <sup>(1)</sup>	X	X
✓ Header reception indication <sup>(1)</sup>		X
✓ Response reception indication <sup>(1)</sup>	X	X
✓ Configurable checksum (per frame basis) : Classic (LIN 1.2, 1.3) / Enhanced (LIN 2.0)	X	X
✓ Frame errors detection and notification <sup>(1)</sup> : framing error (monitoring & stop bit error), synchronization error, parity error, checksum error, response time-out, bus forced at dominant state, bus physical error	X	X
<b>LIN Interface – LIN 1.3 protocol handler</b>		
✓ Management of the schedule tables according to the ECU message set	X	
✓ Service for activating/switching schedule tables	X	
✓ Signal management according to the ECU message set Signal-oriented interface (LIN API)	X	X
✓ Detection & handling of communication errors : bus idle, ECU message not transmitted		X
<b>LIN Network Management (for PSA networks)</b>		
✓ Service for mode change request	X	X
✓ Notification of network mode changes <sup>(1)</sup>	X	X
✓ Management of the state diagram for a type 0 organ	X	
✓ Management of the state diagram for type 1 and type 2 organs		X
✓ Diagnostic of the network status (error confirmation filtering)	X	
✓ Notification of a network status change <sup>(1)</sup>	X	
<b>LIN Diagnostic Management</b>		
✓ Decoding of the diagnostic request received on the LIN with application indication for each KWP2000 service granted <sup>(1)</sup>		X
✓ Encoding of the diagnostic responses frames to be sent over the LIN		X
✓ Management of the diagnostic session (open/sustain/close) and request/response timings		X
<b>CAN/LIN Gateway</b>		
✓ Signal gateway : LIN frames encoding with data received from the CAN and CAN frames encoding with data received from the LIN	X	
✓ Diagnostic bridge : Diagnostic requests transportation from CAN network to LIN, diagnostic responses transportation from LIN to CAN network	X	
✓ Management of the schedule tables according to the current CAN mode	X	

<sup>(1)</sup> Notifications and Indications are realized by *CallBack* functions implemented in the upper layer.

All transmission parameters (baudrate, syncho break and syncho break delimiter duration, inter-byte delay...etc) are preconfigured to fulfill PSA requirements but may be adapted to a specific need by changing the driver static configuration for each network. Application may also by-pass the LIN scheduler to manage by itself all LIN transmissions by direct call of the driver services.

Complementary developments based on customer specifications may also be done by the NSI software department using the standard modules or part of them : please contact our representative.