Local interconnect network

Main Features:

- Single master with multiple slaves concept.
- Self Synchronization.
- Single wire.
- Low baud rate.
- Low speed application (Less than 20kps).
- Max 40 m wire length
- The LIN is a SCI/UART-based serial

LIN characteristics :

- The LIN protocol is byte oriented.
- data is sent one byte at a time.
- One byte field contains a start bit (dominant), 8 data bits and a stop bit (recessive).
- The data bits are sent LSB first.
- In automotive application, the LIN bus is connected between
  - smart sensor
  - actuators
  - Electronic Control Unit (ECU)

LIN characteristics :

- Broadcast type serial network.
- Single wire 12V bus connection.
- Has the synchronization mechanism that allows the clock recovery by slave nodes.
- Only the master node will be using the oscillating device.
- Nodes can be added to the LIN network without requiring HW/SW changes in other slave nodes.

LIN History:

- In 1996, Volvo and Volcano Communication Technologies (VCT) developed a UART based protocol for the Volvo S80 series, called Volcano Lite.

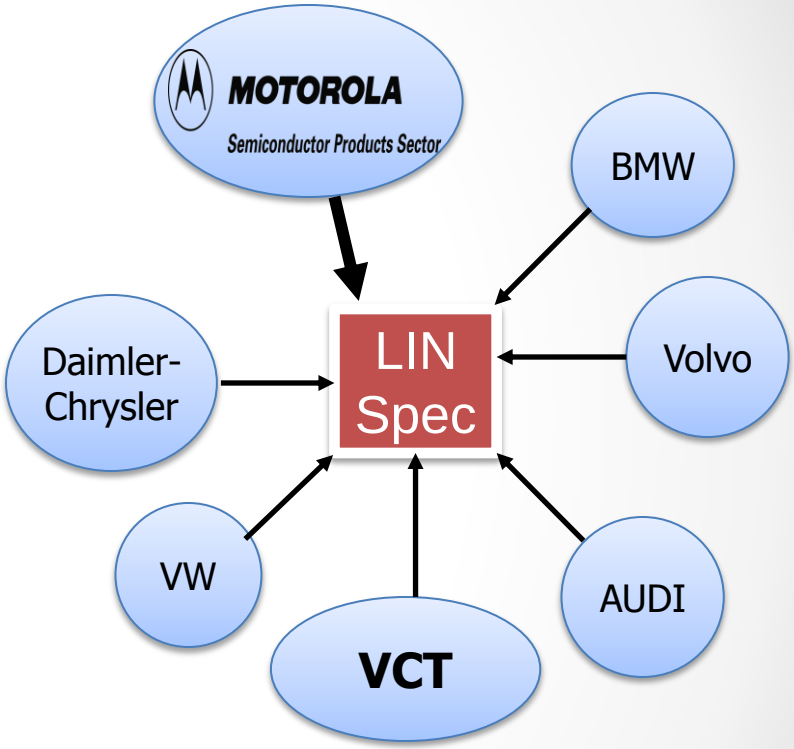
  This protocol was an integral part of the vehicle communication system.

- In 1997, Motorola joined Volvo and VCT in improving the Volcano Lite protocol .

  - **Self-synchronization of the slave**
  - **Form an open standard**

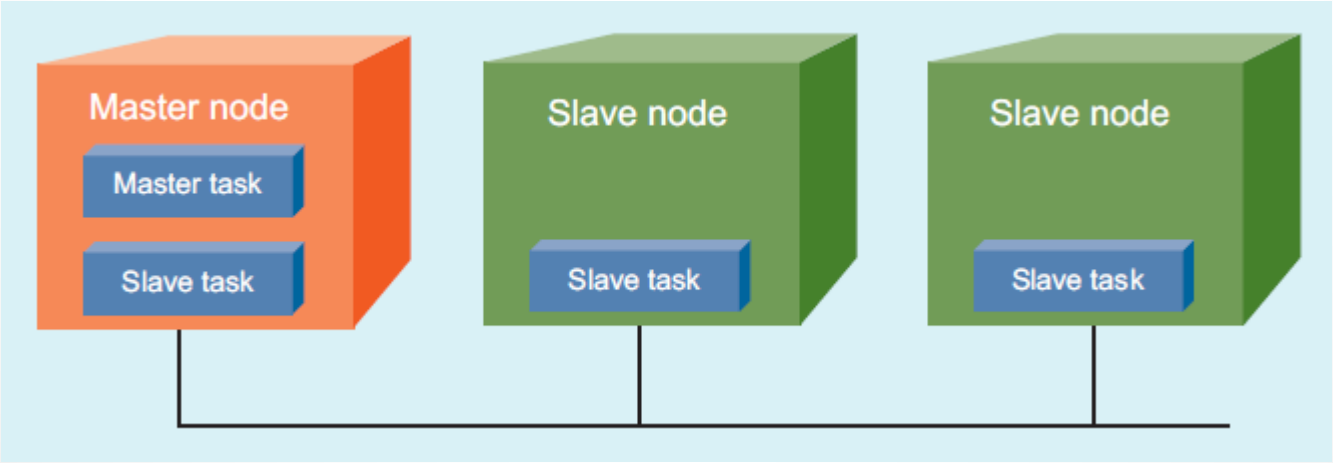
LIN History:

- In December 1998 , Audi, BMW, DaimlerChrysler and VW joined the activities and formed to set up the LIN communication protocol.
- September , LIN API specification draft was released (Rev. 0.1).
- In November 2002, LIN 1.3 was released.
- The latest version LIN 2.0 released in 2003.



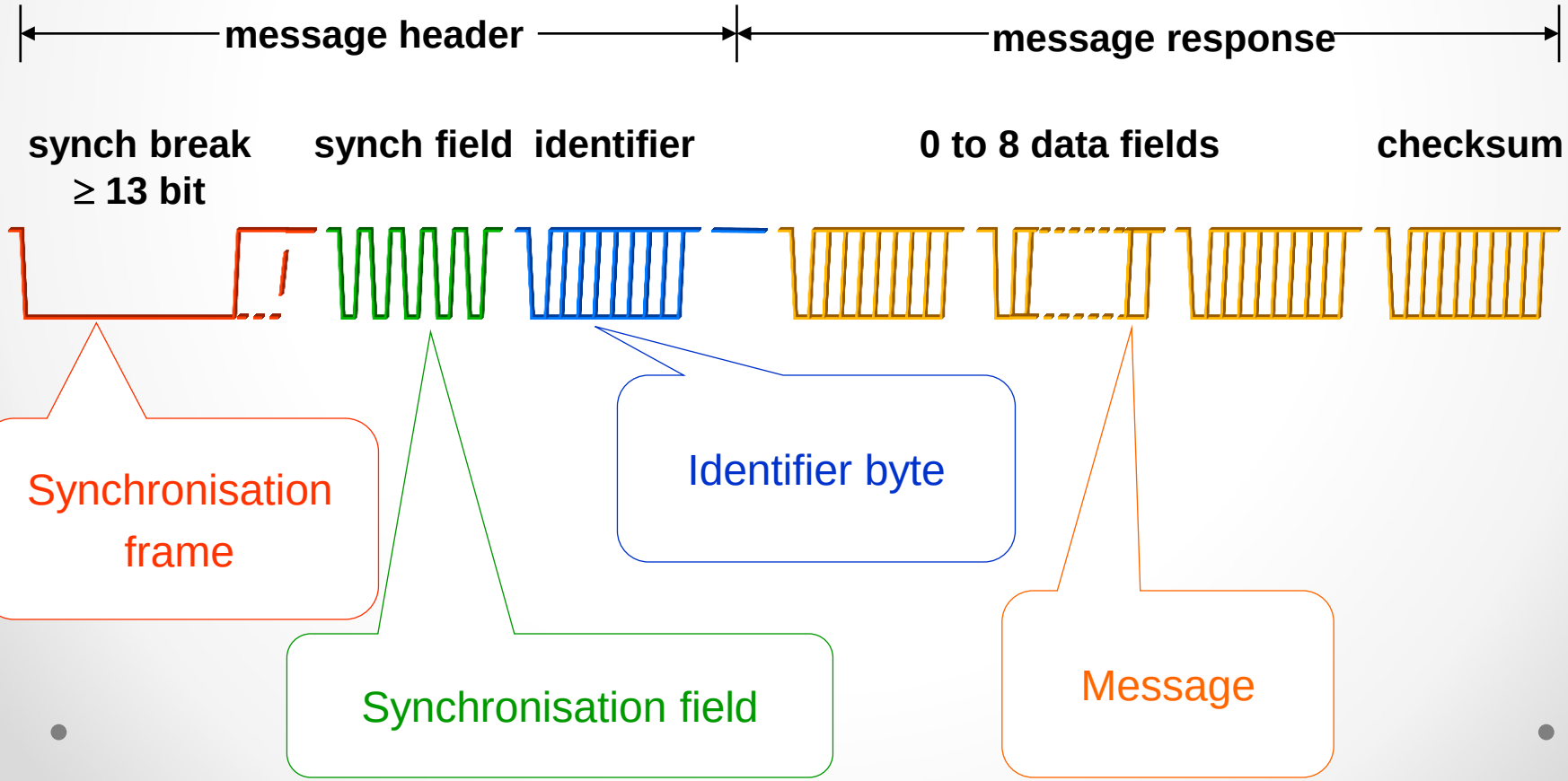
Concept of Operation

- The LIN bus is a single master device and multi slave devices.
- The master device contains both a master task and a slave task.
- Each slave device contains only a slave task.
- Communication over the LIN bus is controlled by master task.



LIN Message Frame

- The basic unit of transfer on the LIN bus is the frame.
- Divided into a header and a response.

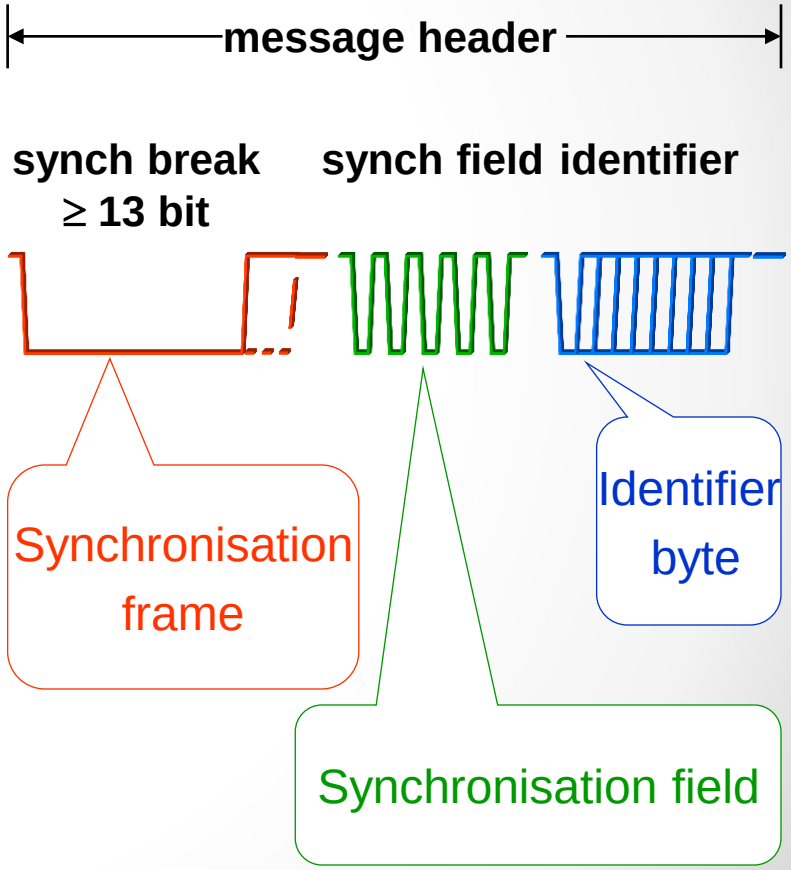


LIN Message Frame

Master Task:

- Control over the whole Bus.
- Controls which message at what time is to be transferred over the bus.
- Send Header:

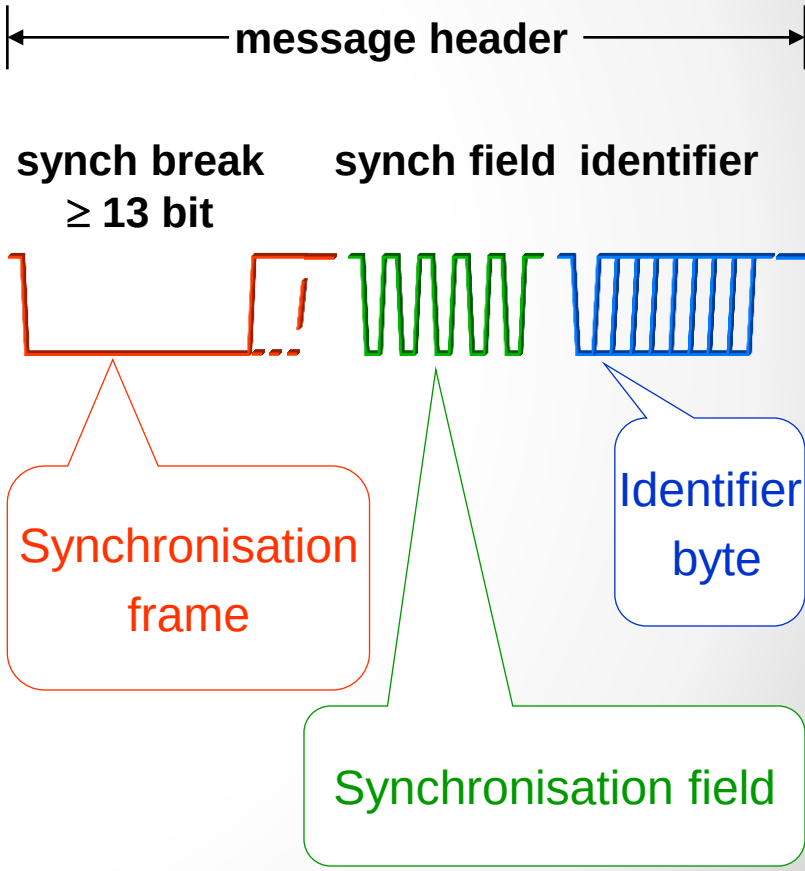
  Sync Break, Sync Byte ,ID-Field.



LIN Message Frame

Master Task:

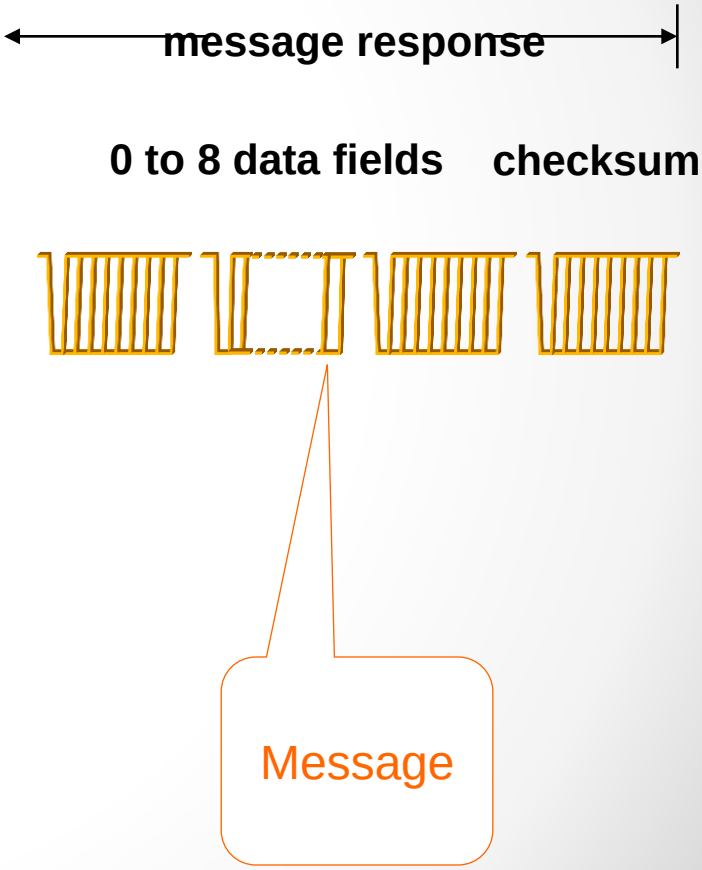
- Error handling.
  - monitors Data Bytes and Check Byte, and evaluates them on consistence
- Receives Wakeup Break from slave nodes when the bus is inactive.
- Defines the transmission speed.
- Switching slave nodes to sleep/wake up mode.



LIN Message Frame

Slave Task:

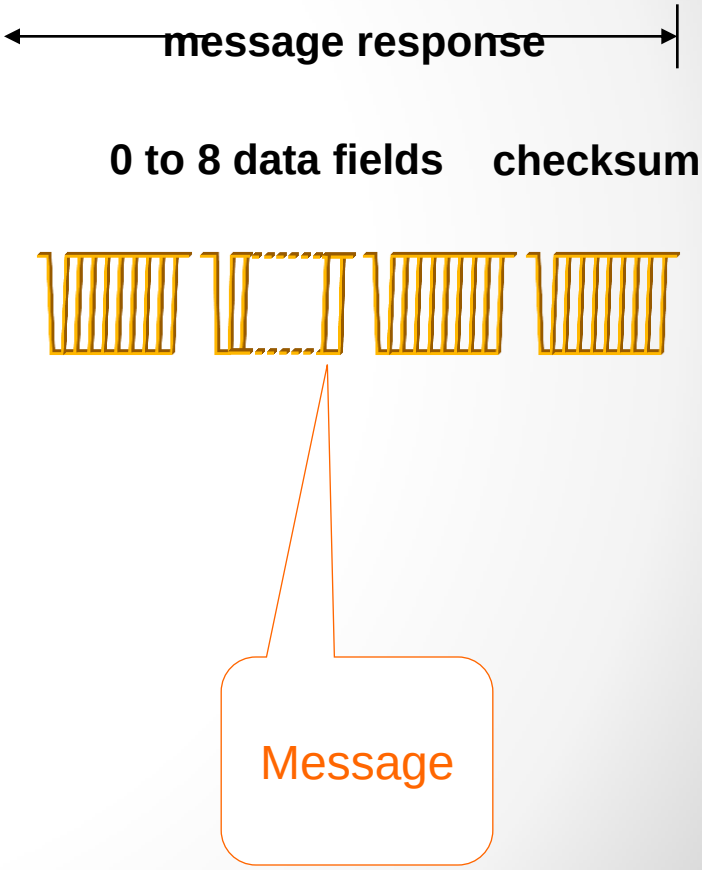
- One of 2-16 Members on the Bus.
- Receives or transmits Data when appropriate ID is sent .
- Slave snoops for ID.



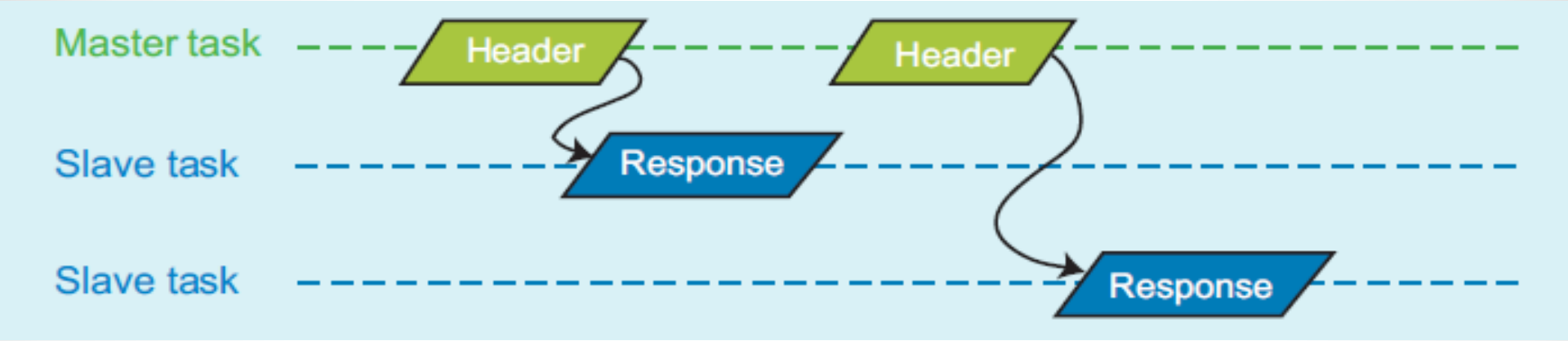
LIN Message Frame

Slave Task:

- According to ID, slave determines:
  - receive data, transmit data ,do nothing.
- When transmitting :
  - sends 1, 2, 4, or 8 Data Bytes + Check-Byte
- The node serving as a master can be slave.



Master and Slave Communication



T_1 : Master task transmits header for ID 0;

Break	Sync	ID=0

T_2 : Upon receiving the header, the slave task configured to publish data for ID 0 transmits a response.

Data Bytes	Checksum

Break	Sync	ID=0	Data Bytes	Checksum

The inter-frame delay for each ID is specified in the schedule table.

T_3 : Master task transmits header for ID 1;

Break	Sync	ID=1

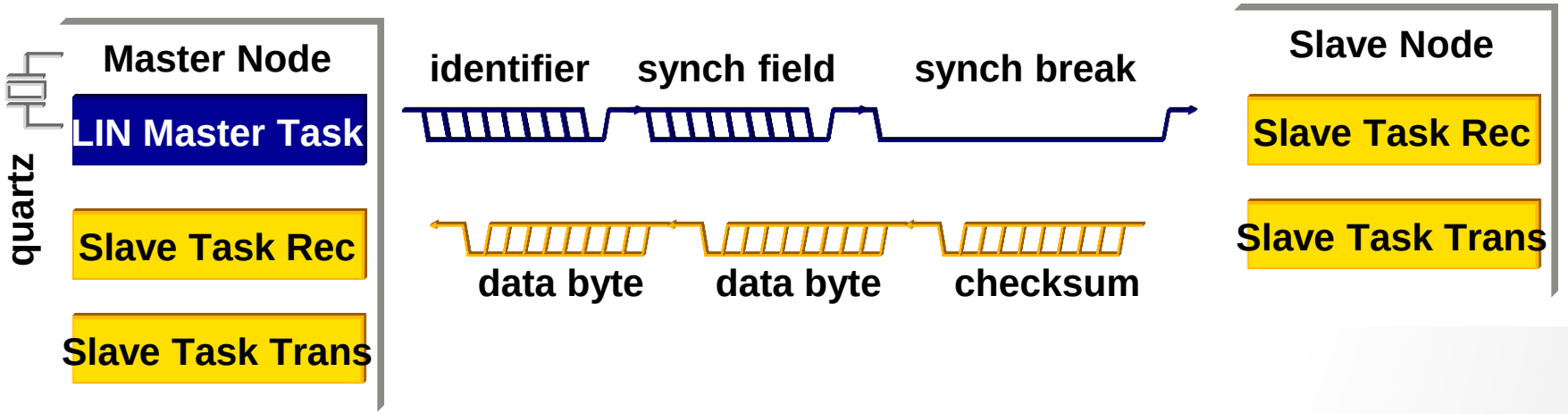
T_4 : Upon receiving the header, the slave task configured to publish data for ID 1 transmits a response.

Data Bytes	Checksum

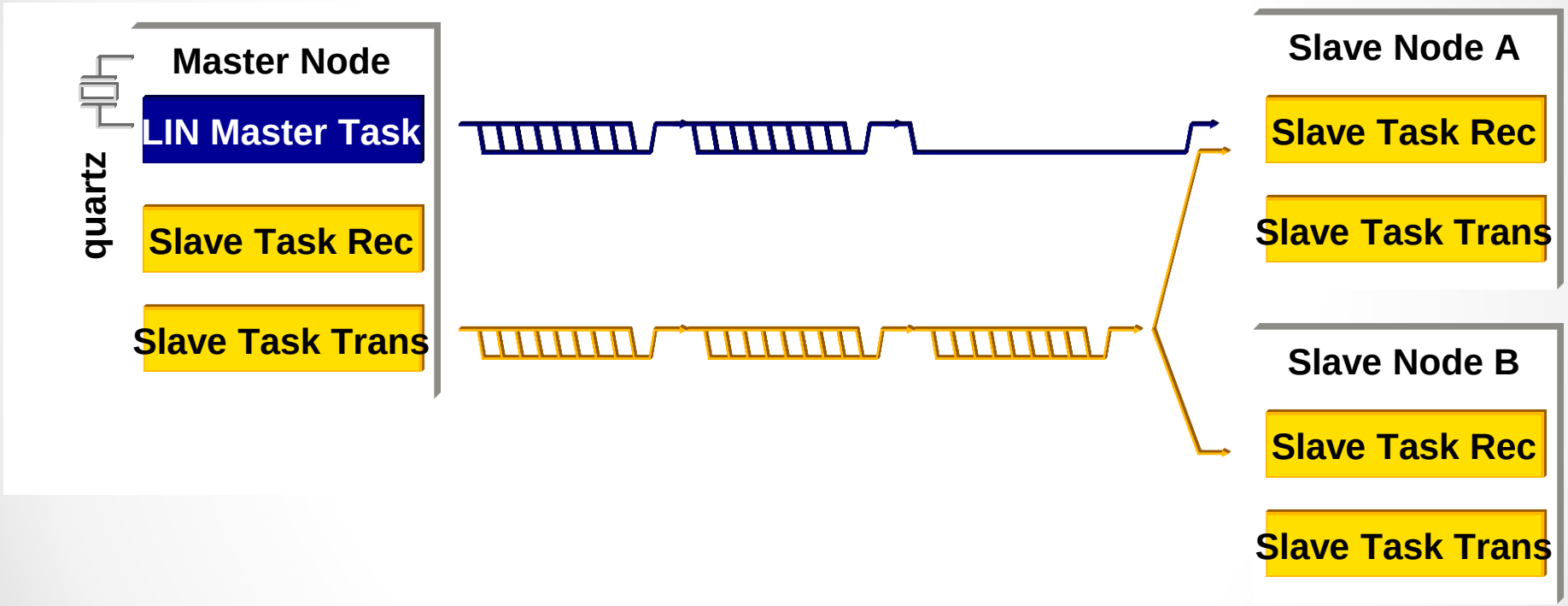
Resulting full LIN frame:

Break	Sync	ID=1	Data Bytes	Checksum

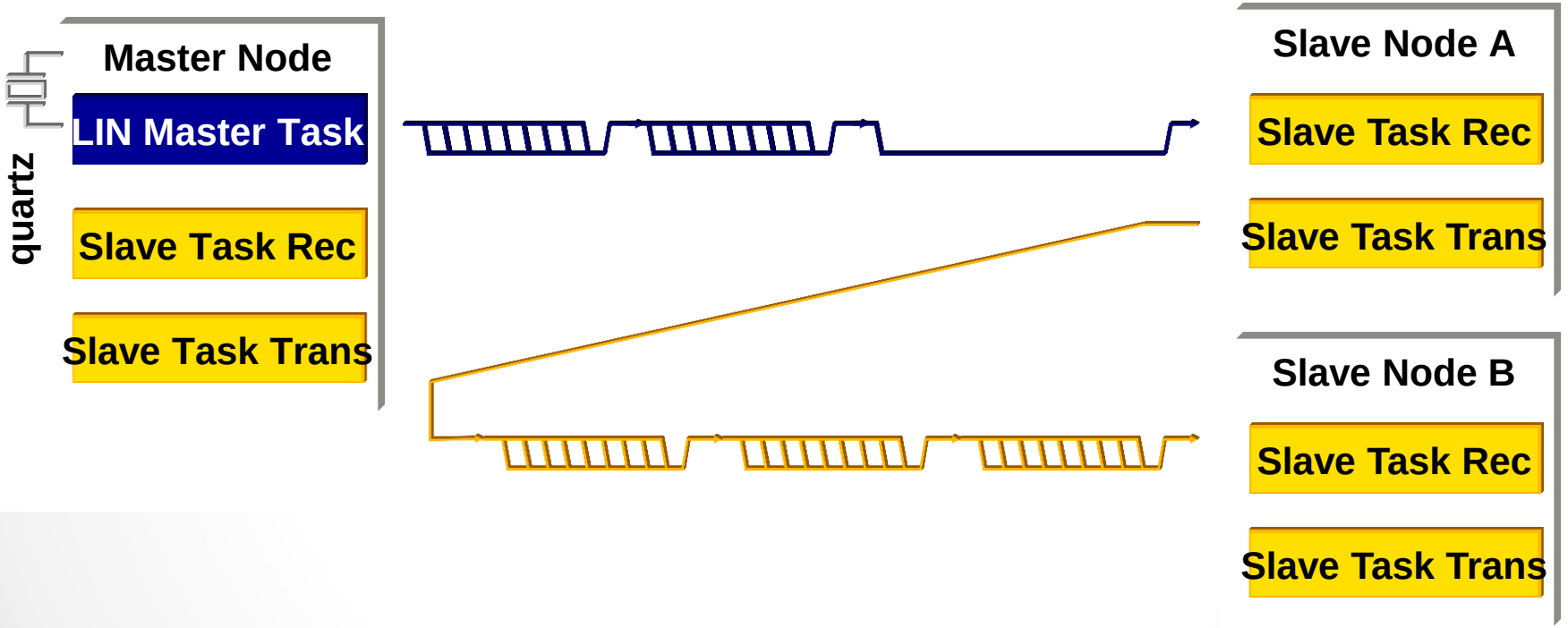
Master and Slave Communication



Master and Slave Communication



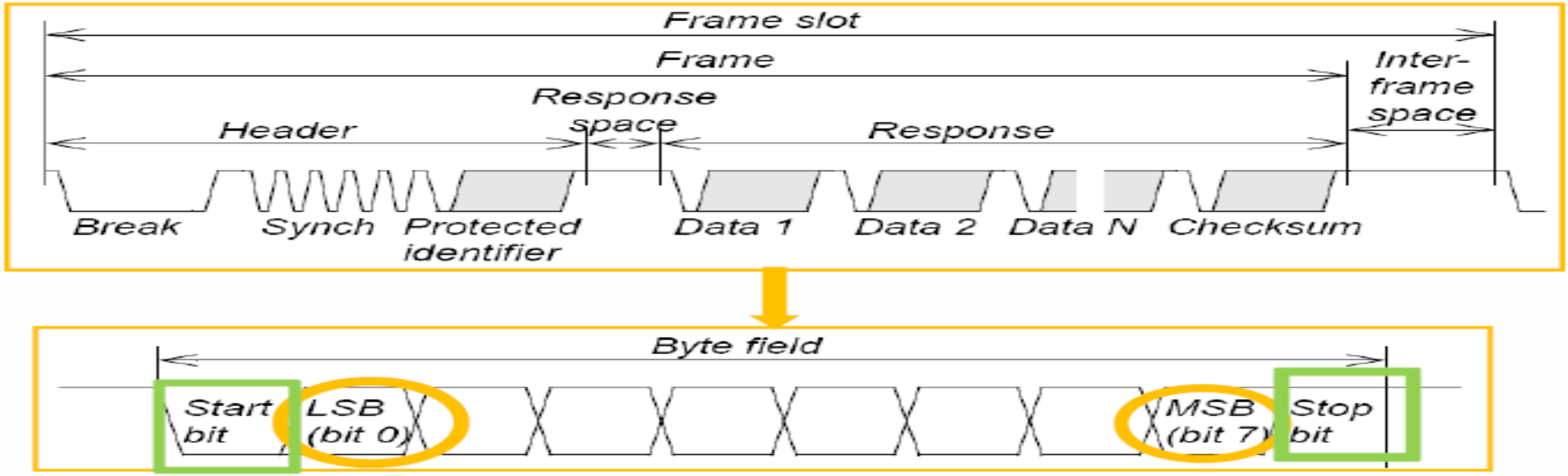
Master and Slave Communication



Frame Structure

Structure of a Byte field:

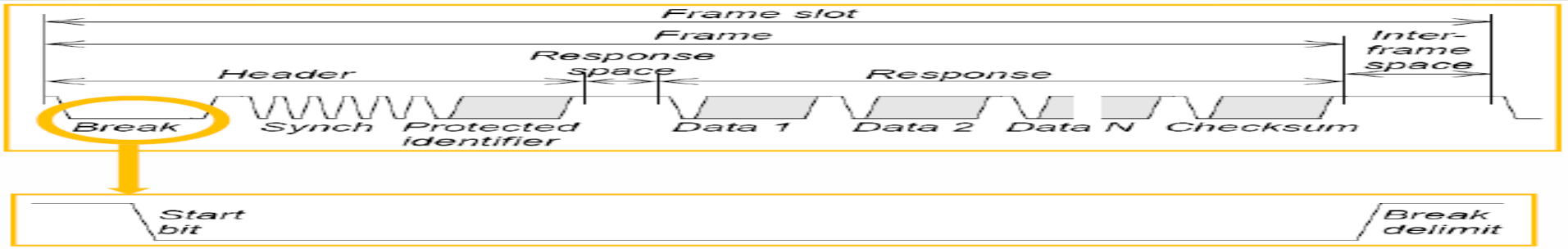
- The LSB of the data is sent first and the MSB last.
- The start bit is encoded as a bit with value zero (dominant) &the stop bit is encoded as a bit with value one (recessive).



Frame Structure

Break

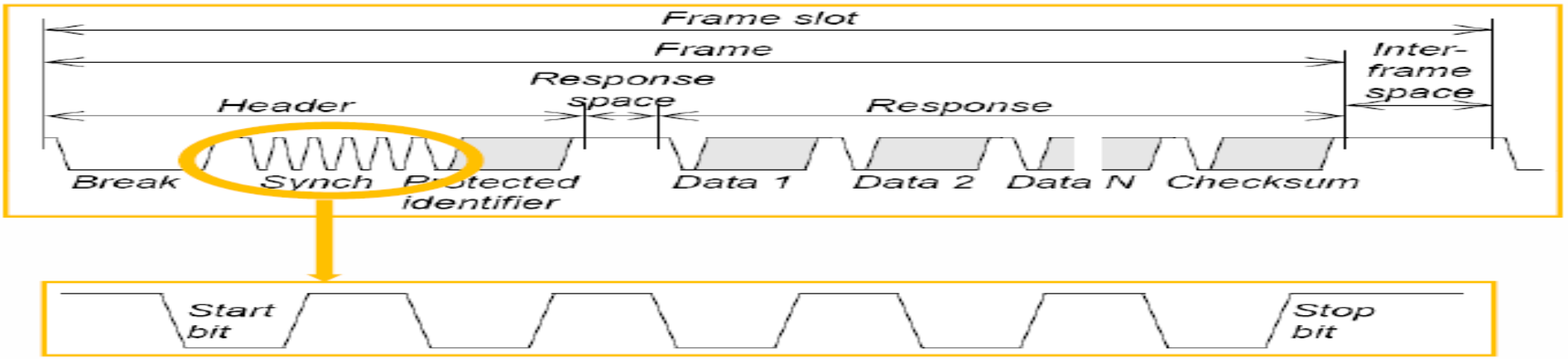
- The break symbol is used to signal the beginning of a new frame.
- A break is always generated by the master task and it shall be at least 13 bits of dominant value, including the start bit, followed by a break delimiter
- Synch break ends with a “break delimiter” which should be at least one recessive bit.



Frame Structure

Synch Byte

- Synch is a byte field with the data value 0x55.
- A slave task shall always be able to detect the break/synch symbol sequence.
- Synch byte is sent to decide the time between two falling edges and thereby determine the transmission rate.
- The bit pattern is 0x55 (01010101, max number of edges).



Frame Structure

Protected Identifier

1. Identifier:

- Six bits are reserved for the identifier (ID).
- Values in the range 0 to 63 can be used.
- The identifiers are split in four categories:
  - Values 0 to 59 (0x3b) are used for signal carrying frames.
  - 60 (0x3c) and 61 (0x3d) are used to carry diagnostic data.
  - 62 (0x3e) is reserved for user defined extensions.
  - 63 (0x3f) is reserved for future protocol enhancements.

Frame Structure

Protected Identifier

1. Identifier:

- Contains information about sender and receiver and the number of bytes which is expected in the response.

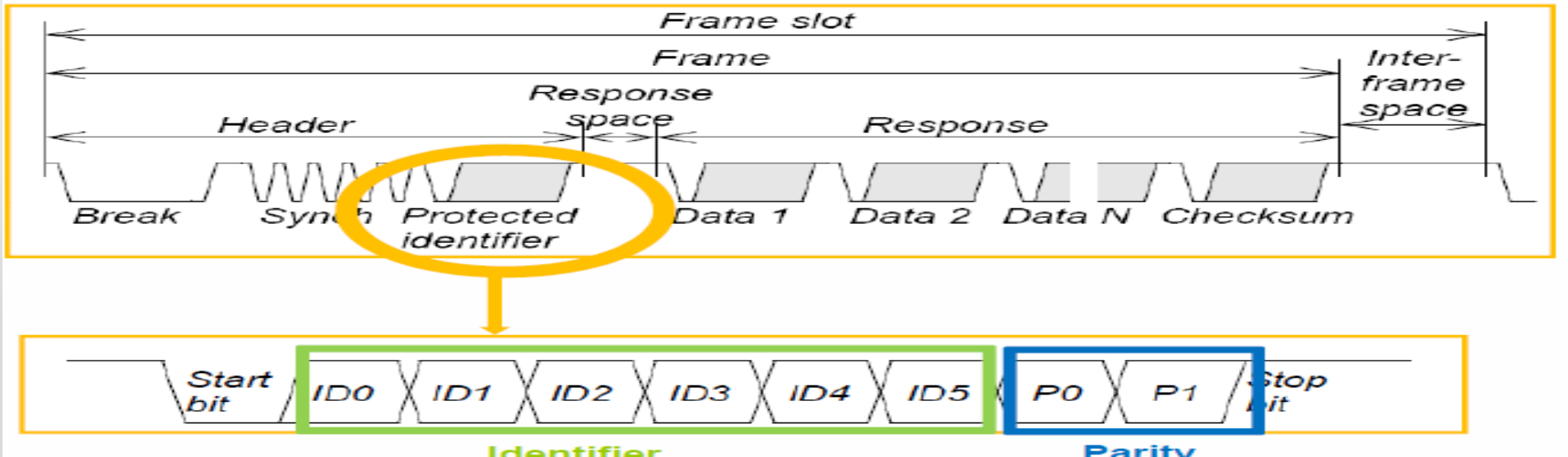
ID range		Frame length
0-31	0x00 – 0x1f	2
32-47	0x20 – 0x2f	4
48-63	0x30 – 0x3f	8

Frame Structure

Protected Identifier

2. Parity:

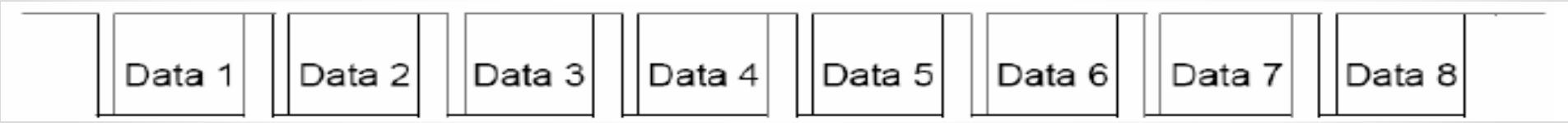
- The parity is calculated on the identifier bits.
- P0 = ID0 .ID1 .ID2 .ID4
- P1 = ⊙(ID1 .ID3 .ID4 .ID5)



Frame Structure

Data

- A frame carries between one and eight bytes of data
- A data byte is transmitted in a byte field
- The data bytes field is transmitted by the slave task in the response.
- Can be 2, 4 or 8 bytes long depending on the two MSB (Most Significant Byte) of the identifier sent by the master.
- This ability came with LIN 2.0, older versions have a static length of 8 bytes.

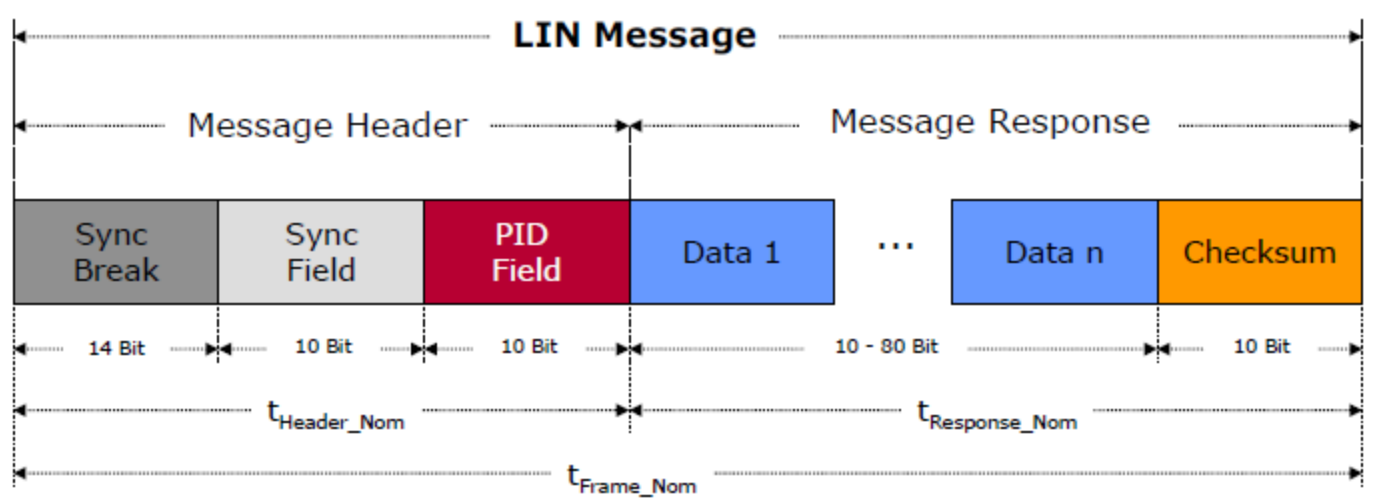


Frame Structure

Checksum:

- The LIN bus defines the use of one of two checksum algorithms to calculate the value in the eight-bit checksum field:
  - Classic checksum is calculated by summing the data bytes alone.(V1.3)
  - Enhanced checksum is calculated by summing the data bytes and the protected ID.(V2.0)

LIN Bus Timing



$$t_{Header\_Nom} = (N_{Sync\_Field} + N_{Sync\_Byte} + N_{PID\_Byte}) \cdot t_{Bit} = 34 \cdot t_{Bit}$$

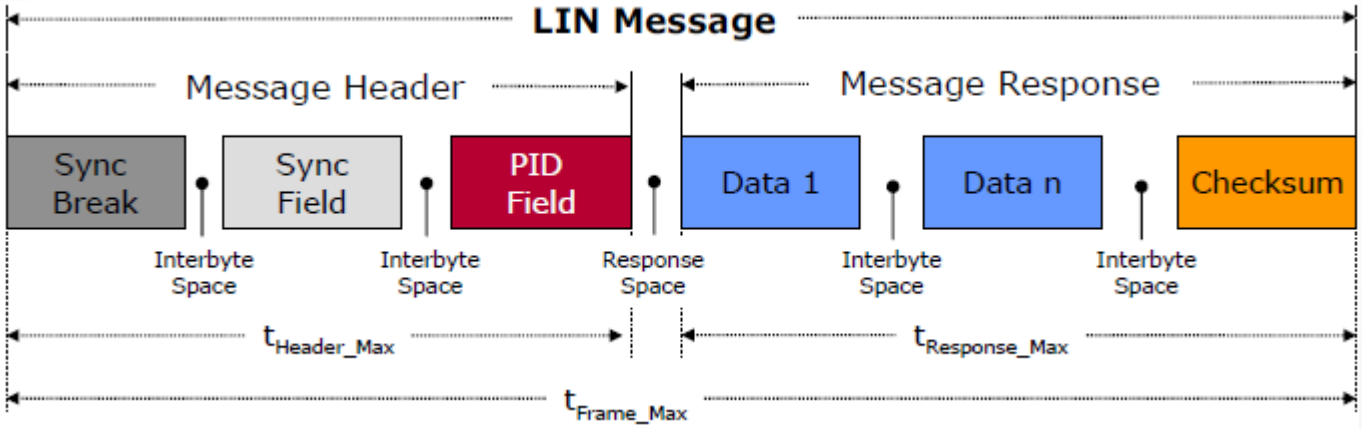
$$t_{Response\_Nom} = 10 \cdot (N_{Data} + 1) \cdot t_{Bit}$$

$$t_{Frame\_Nom} = t_{Header\_Nom} + t_{Response\_Nom}$$

LIN Bus Timing

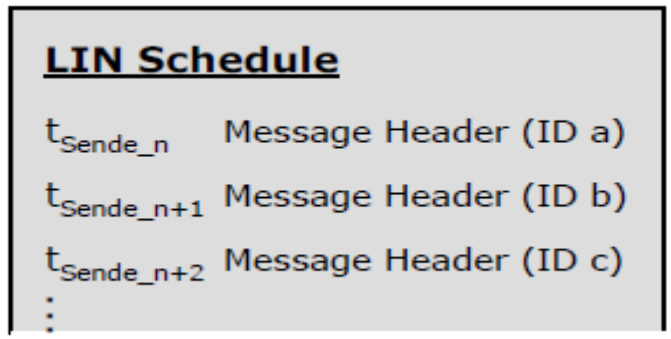
- A time reserve of up to 40% is given for transmission of a LIN message

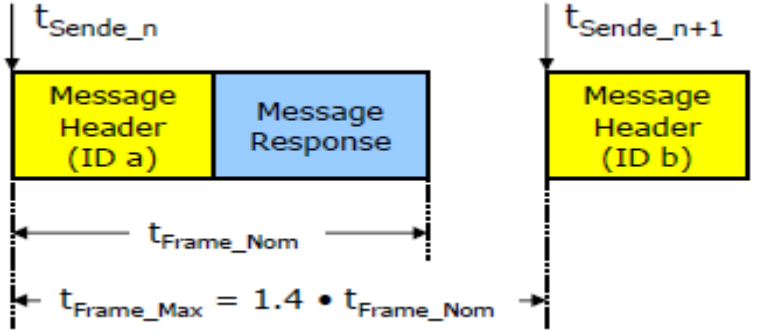
$$t_{Frame\_Max} = t_{Header\_Max} + t_{Response\_Max} = \mathbf{1.4} \bullet t_{Frame\_Nom}$$



Schedule Table

- The master task (in the master node) transmits frame headers based on a schedule table.
- The schedule table specifies the identifiers for each header and the interval between the start of a frame and the start of the following frame.
- The master application may use different schedule tables and select among them.

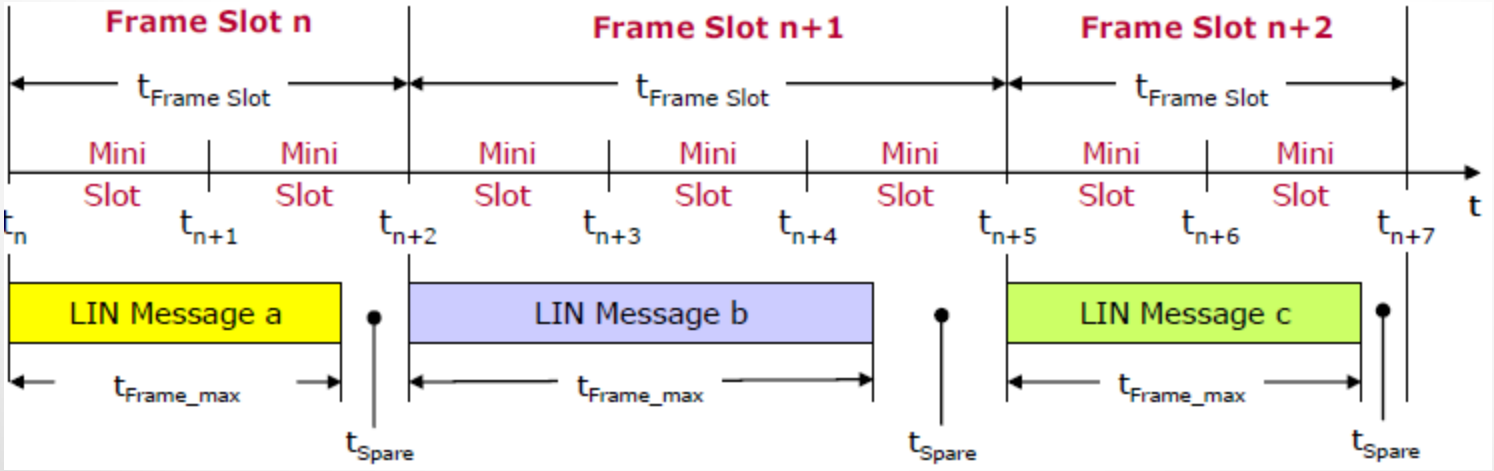




Schedule Table

- The LIN Schedule is organized in Mini Slots (tMini Slot = tTime-Base)
- An adequate number of Mini Slots must be provided to guarantee transmission of a LIN message

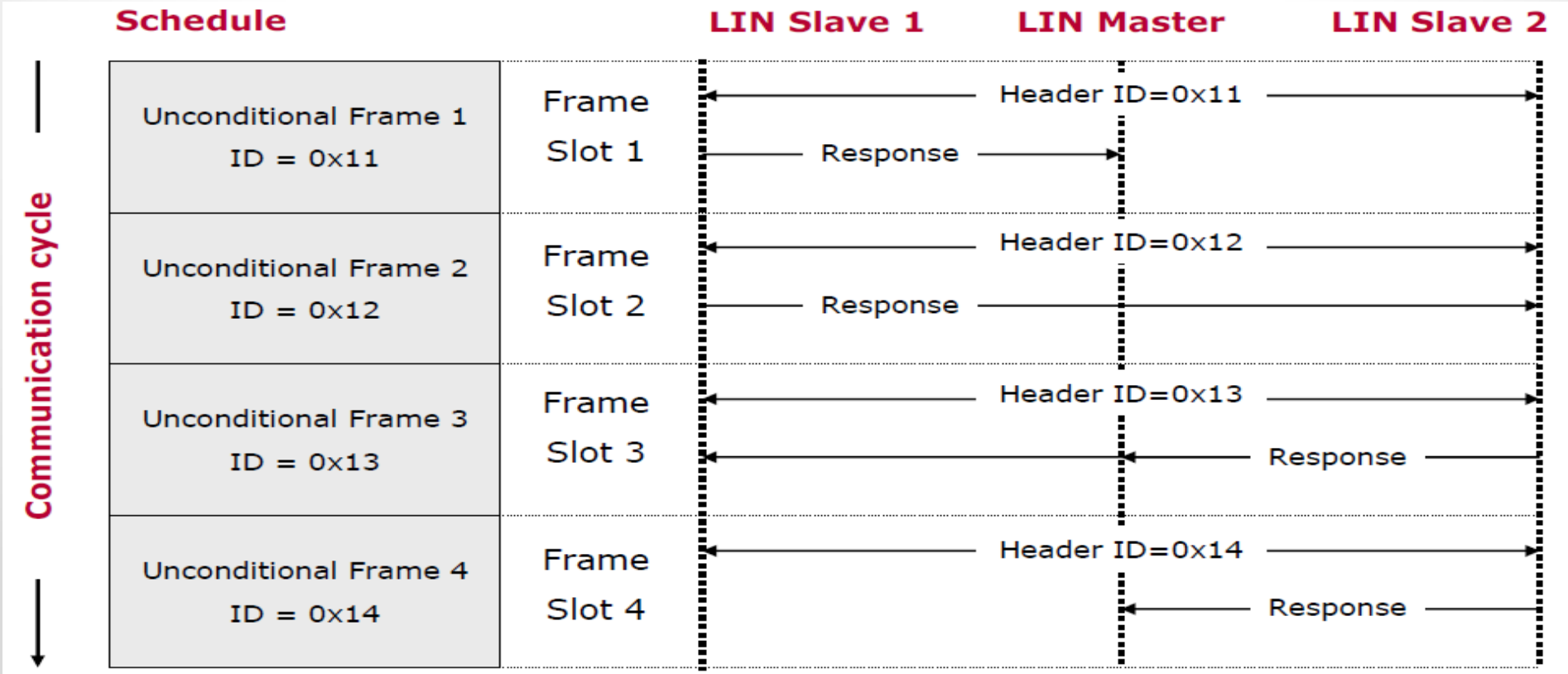
LIN Schedule ($t_{Time\text{-}Base}$)	
t_n	Mini Slot
t_{n+1}: $t_n + t_{Time\text{-}Base}$	Mini Slot
t_{n+2}: $t_{n+1} + 2 \bullet t_{Time\text{-}Base}$	Mini Slot
t_{n+3}: $t_{n+2} + 3 \bullet t_{Time\text{-}Base}$	Mini Slot
⋮	



Frame Types

1.Unconditional Frame

- Characterized in that there is exactly one sender of the Message Response.



Frame Types

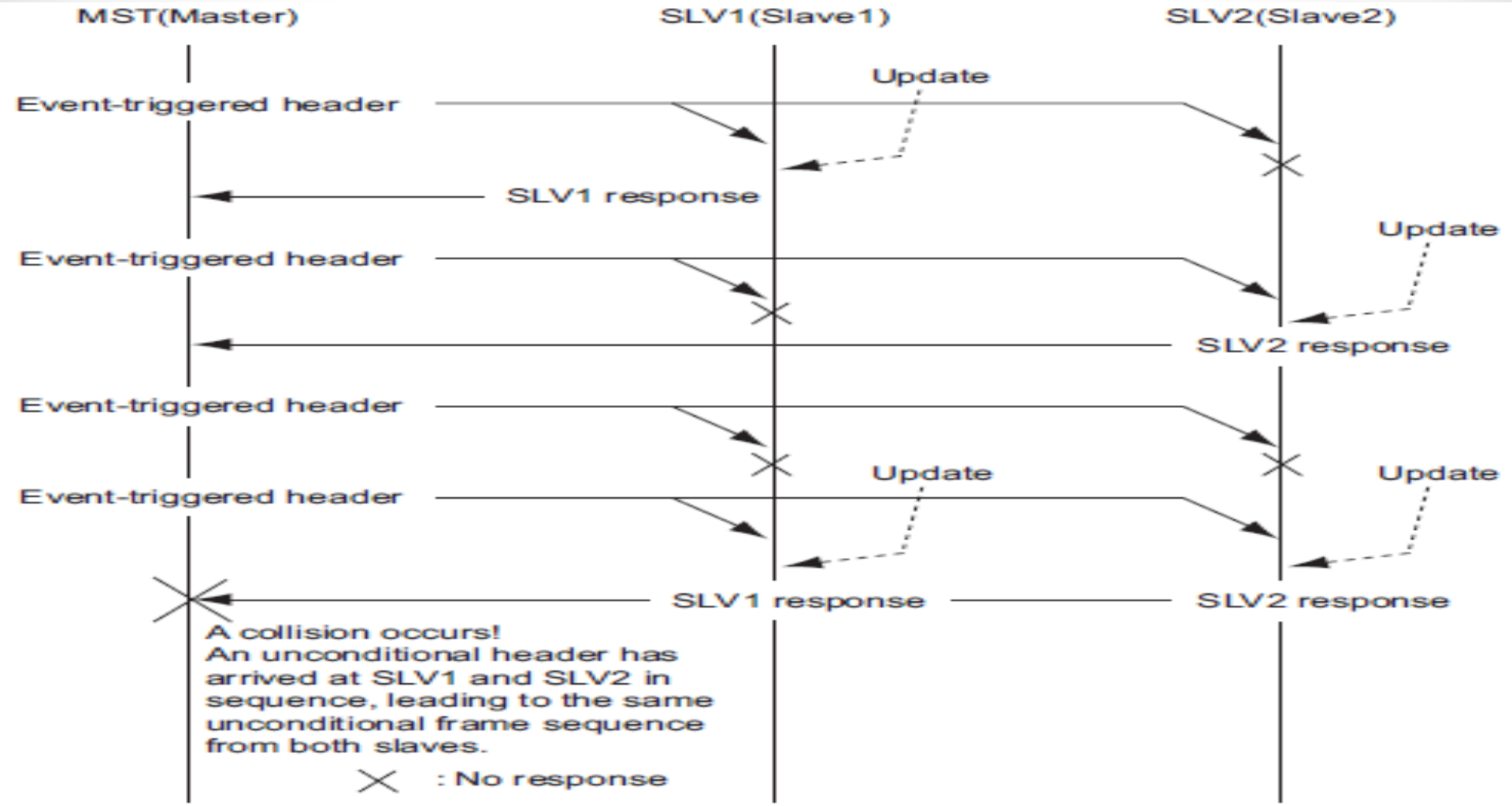
2.Event-triggered frame

- Confirm the availability of an update to the value of a signal.
- Only slave nodes with updated signal values transmit responses to the header.
- The transmission of responses by several slave nodes may lead to a collision.
- When a collision occurs:

  the master node sends requests for the confirmation of signal values to all of the slave nodes via an unconditional frame.

Frame Types

2.Event-triggered frame



Frame Types

2.Event-triggered frame

- A typical use for the event triggered frame is to monitor the door knobs in a four door central locking system.
  - By using an event triggered frame to poll all four doors the system shows good response times.

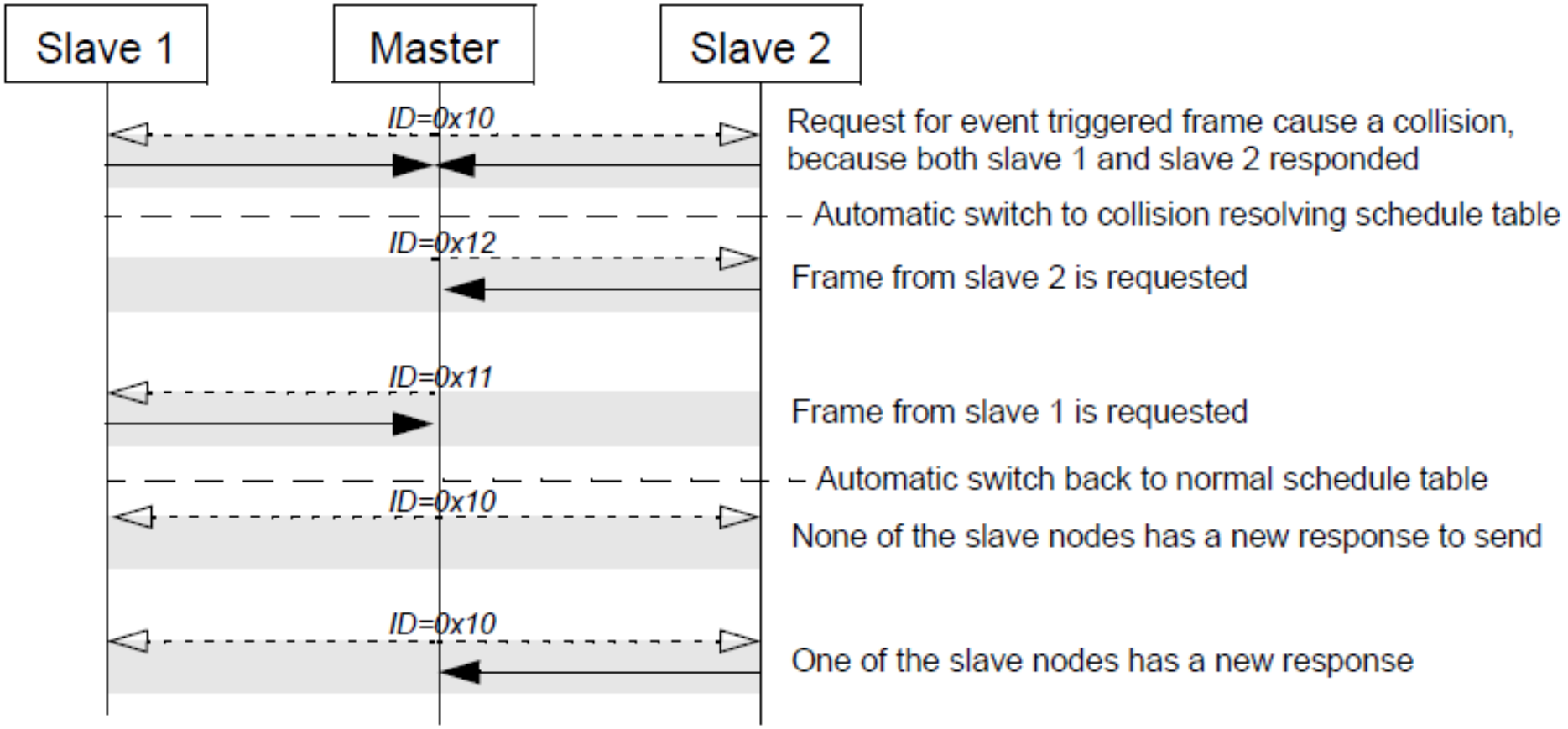
    while still minimizing the bus load.
  - In the rare occasion that multiple passengers press a knob each

    the system will not lose any of the pushes, but it will take some additional time.

Frame Types

2.Event-triggered frame

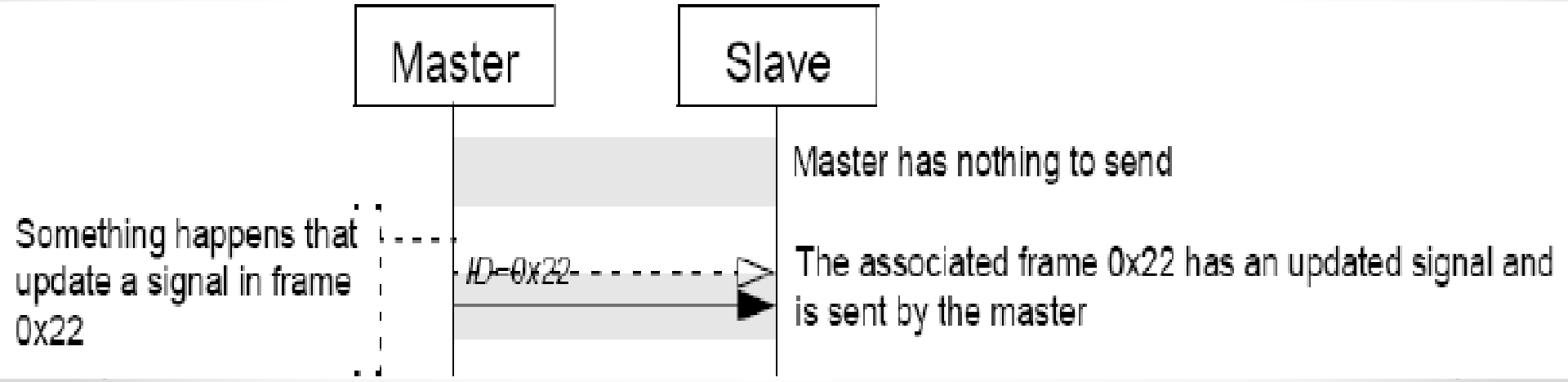
A schedule table contains one event-triggered frame (ID=0x10).



Frame Types

3.Sporadic frames

- Used to inform all relevant slave nodes of the updating of a signal value.
- Managed by the master node.
- Only the master node sends out a response to the header.



Frame Types

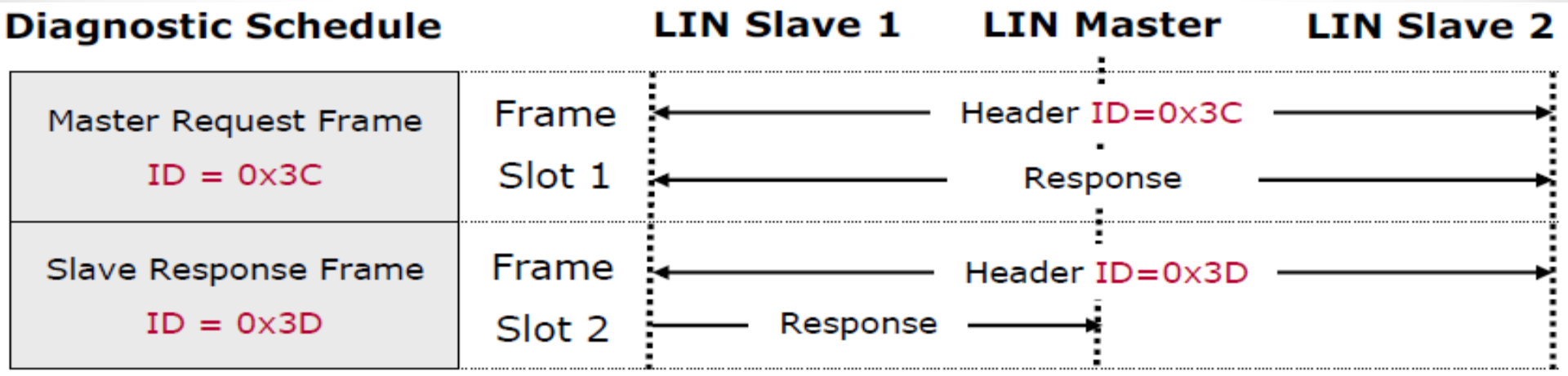
4.User-defined frames

- have an ID of 62.
- carry any type of information.

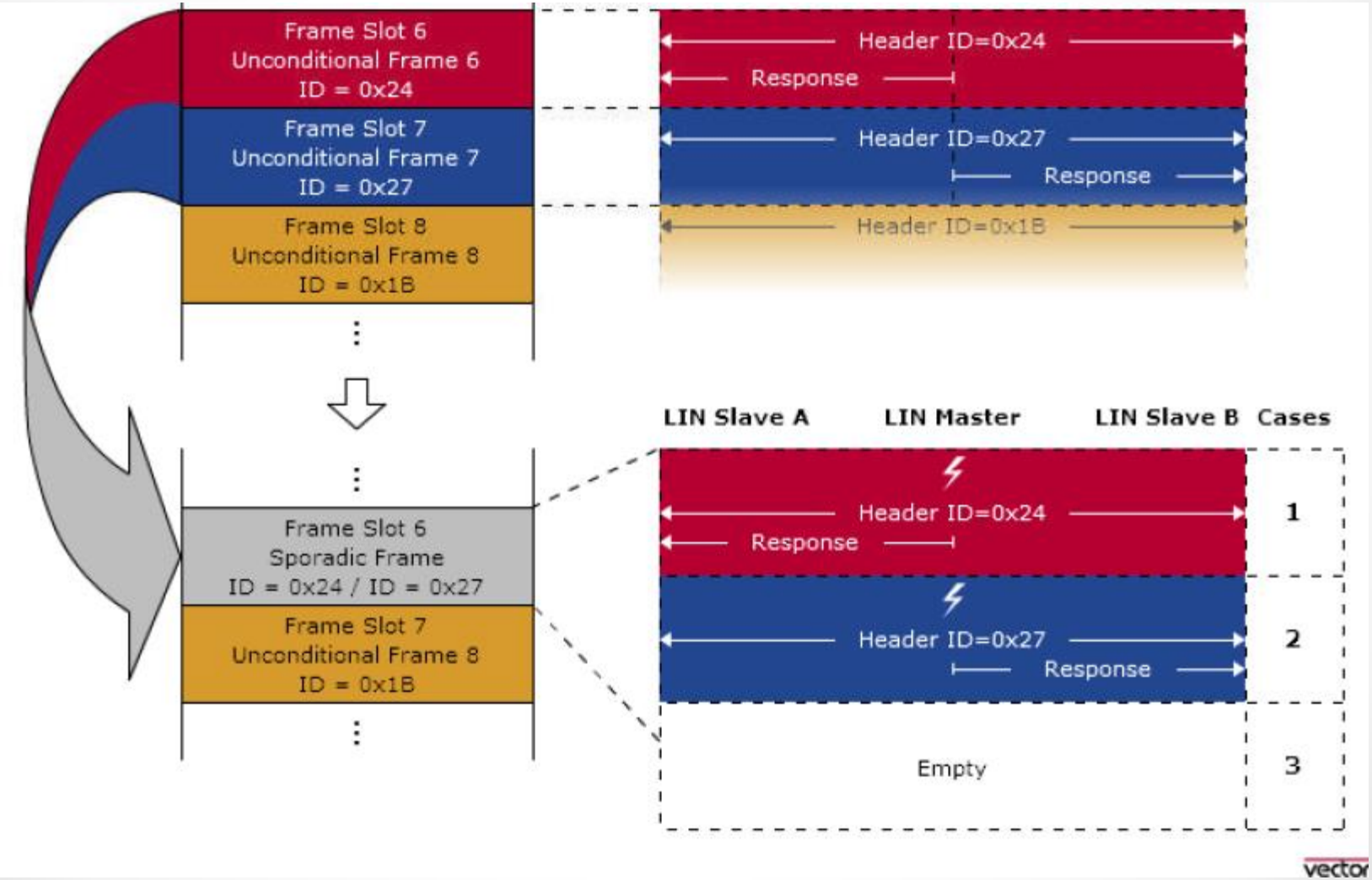
Frame Types

5.Diagnostic frames

- Eight data bytes in length
- Carry diagnostic or configuration data.
- Their IDs are :
  - 60 for a master request frame.
  - 61 for a slave response frame.



Frame Types


Frame Slot 6
Unconditional Frame 6
ID = 0x24
Frame Slot 7
Unconditional Frame 7
ID = 0x27
Frame Slot 8
Unconditional Frame 8
ID = 0x1B
Header ID=0x24
Response
Header ID=0x27
Response
Header ID=0x1B
Frame Slot 6
Sporadic Frame
ID = 0x24 / ID = 0x27
Frame Slot 7
Unconditional Frame 8
ID = 0x1B
LIN Slave A
LIN Master
LIN Slave B
Cases
Header ID=0x24
Response
1
Header ID=0x27
Response
2
Empty
3

Diagnostic

- The Diagnostic Schedule is used for diagnostics.
- It must contain two frame slots:
  - The Master Request Frame (Diagnostic Request)
    - LIN Master sends both the Message Header and the Message Response.
  - The Slave Response Frame (Diagnostic Response)
    - LIN Master sends the Message Header, and a LIN Slave sends the Message Response.
- The number of repeats depends on the diagnostic implementation itself.

Diagnostic

- A diagnostic frame is called a PDU (Packet Data Unit) :
  - Starts with a NAD :
    - Addresses a certain node.
    - The value ranges 1-127, 0 is reserved,128-255 are for free usage.
  - Follows a PCI (Protocol Control Information)
    - Handles the flow control.
  - A Service Identifier (SID) specifies the request and which data bytes to follow.

Diagnostic

- If the PCI-type is a Single Frame (SF) the whole diagnostic request command will fit into a single PDU.

NAD	PCI	SID	Data1	Data2	Data3	Data4	Data5

Request frame PCI-type = SF

Diagnostic

- If the PCI-type is First Frame (FF) the next byte (LEN) will describe the number of bytes to come.
- The data bytes that do not fit into the first frame will be sent in the following frames with the PCI-type of Continuation Frames (CF).

NAD	PCI	LEN	SID	Data1	Data2	Data3	Data4

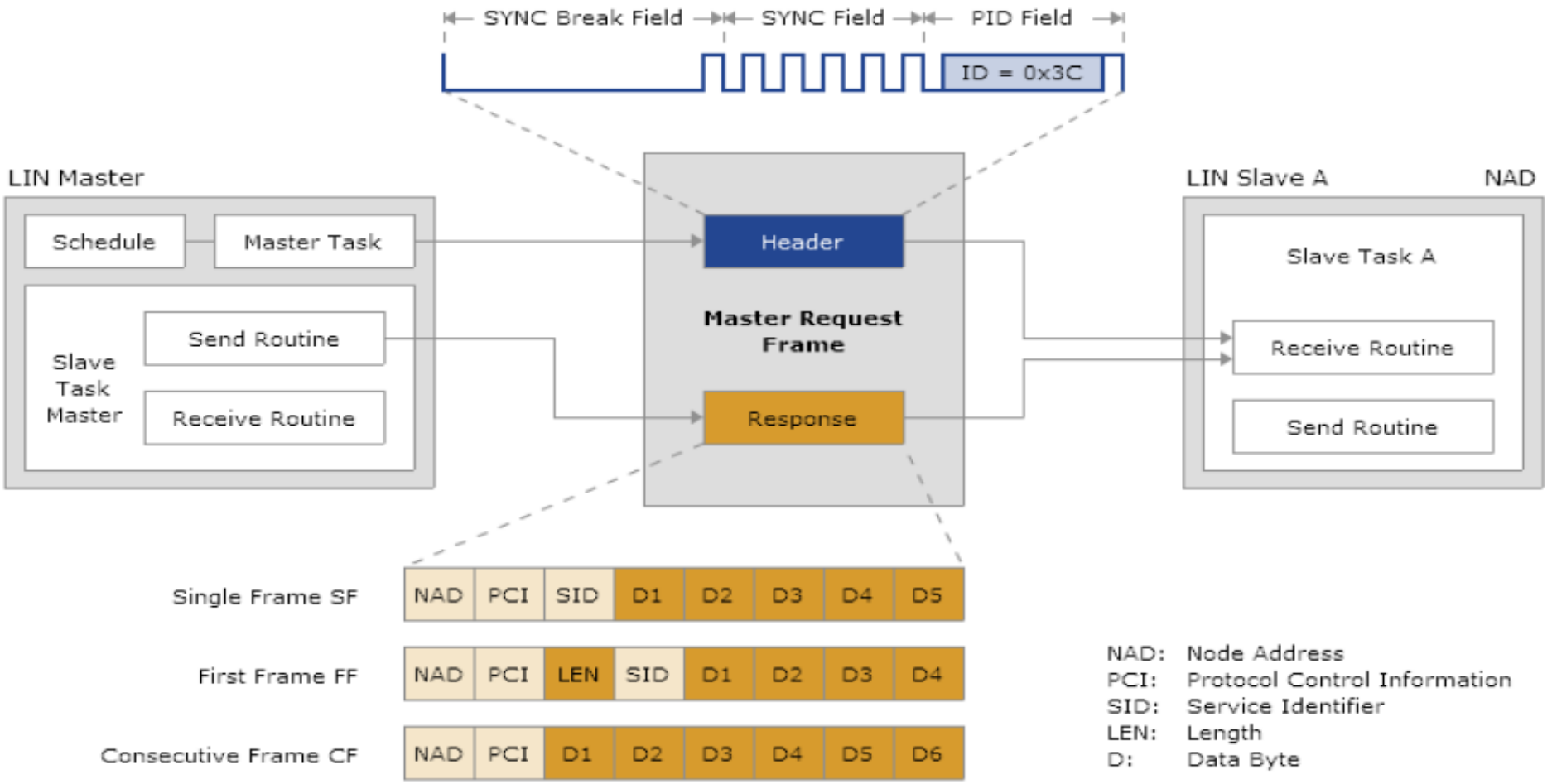
Request frame PCI-type = FF

NAD	PCI	Data	Data2	Data3	Data4	Data5	Data6

Request frame PCI-type = CF

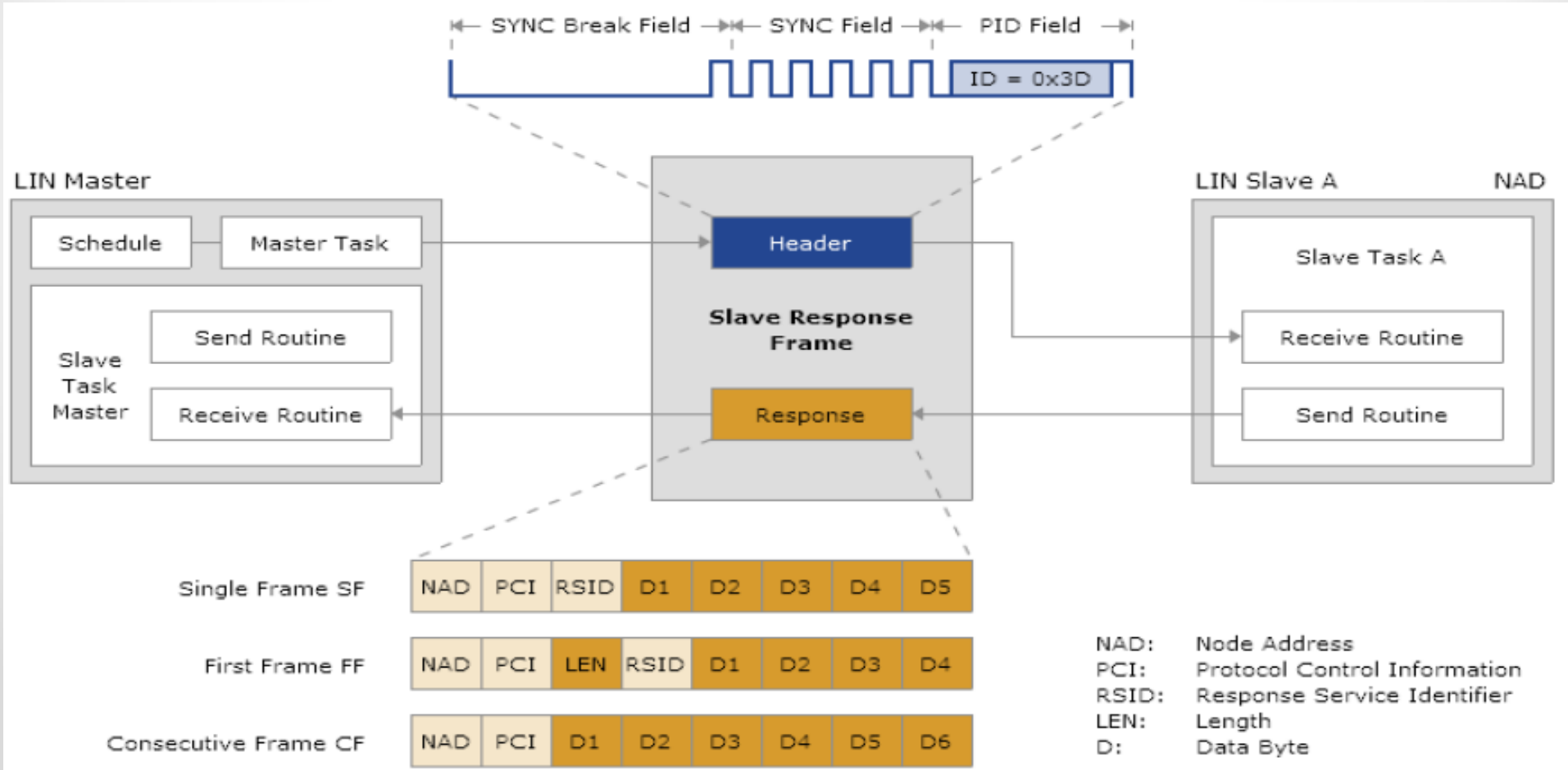
Diagnostic

LIN Master sends both the Message Header and the Message Response.



Diagnostic

LIN Master sends the Message Header, and a LIN Slave sends the Message Response.



Diagnostic

- Methods for diagnostics:
  1. Signal based diagnostic.
  2. User defined diagnostic.
  3. Diagnostic transport layer.

Diagnostic

Methods for diagnostics

1. Signal based diagnostic:
   - The simplest method and uses standard signals in ordinary frames which represent:
     - ✓ Low overhead in slave nodes.
     - ✓ A standardized concept.
     - ✓ Static with no flexibility.
2. User defined diagnostic:
   - designed to fit the needs for a specific device.
   - uses NADs in the range 128-255.

Diagnostic

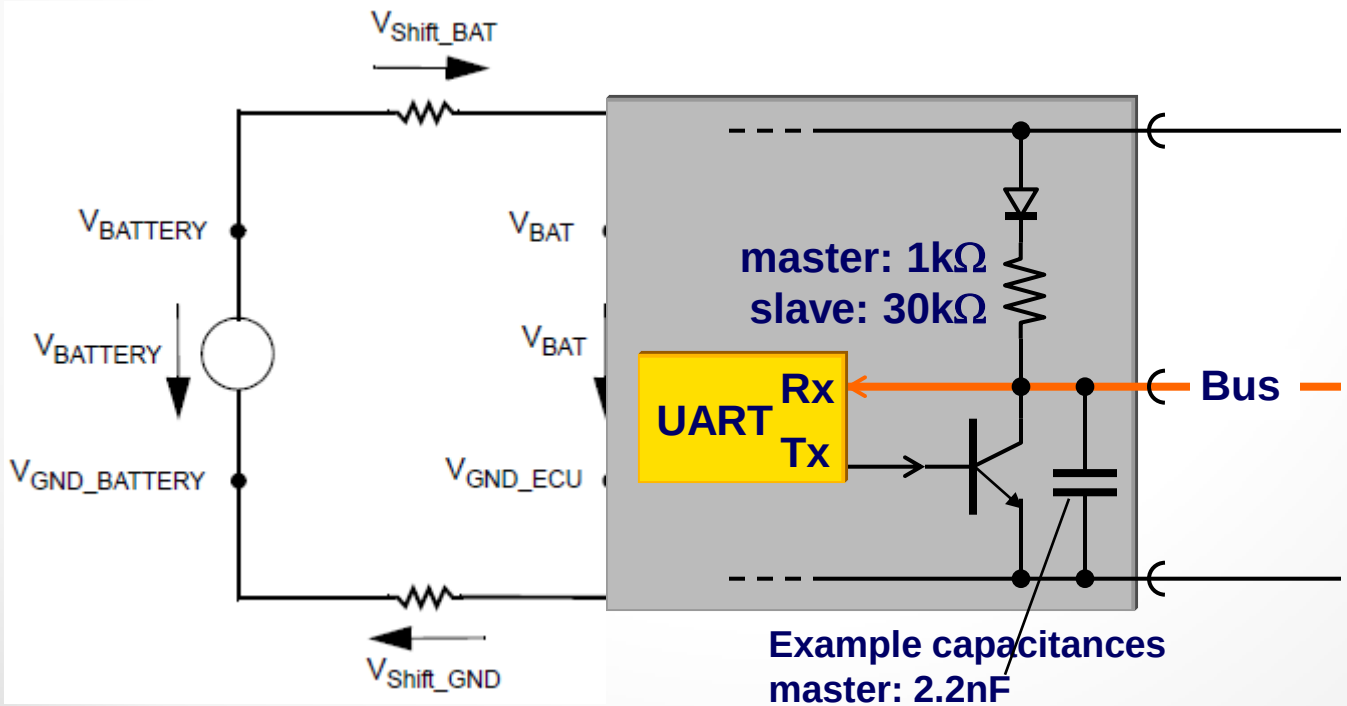
Methods for diagnostics

3. Diagnostic transport layer:

- Useful for a LIN network which is built on a CAN-based system where ISO diagnostics is used.
- NADs 1-127 are used.
- This method represents:
  - ✓ Low load on the master device.
  - ✓ Provides ISO diagnostics for LIN slaves.
  - ✓ Intended for more complex and powerful LIN nodes.

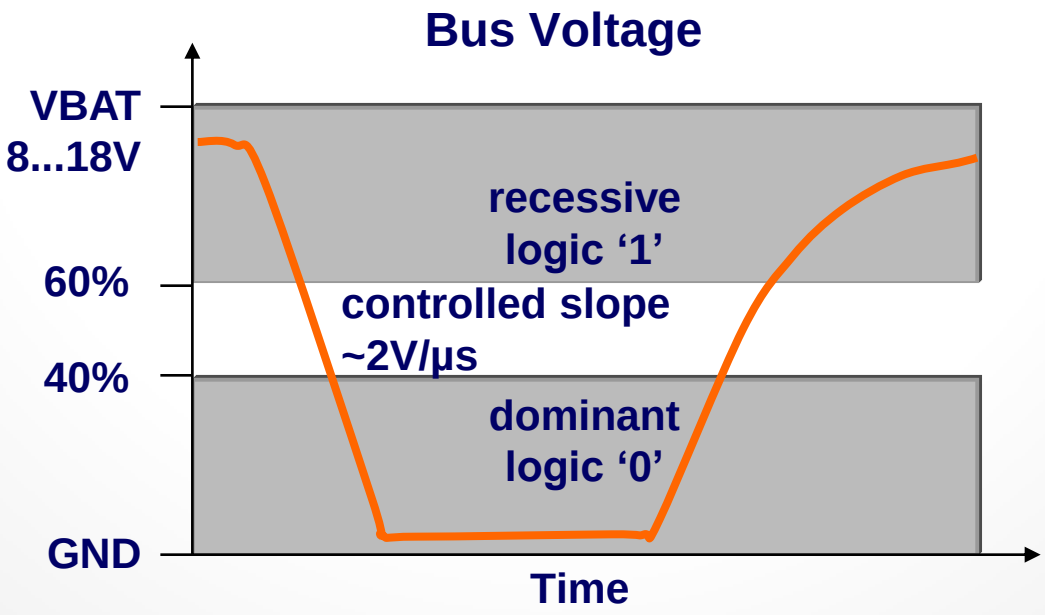
Physical properties

- The LIN-bus transceiver is a modified version of the transceiver used by the ISO 9141 standard.
- The bus is bidirectional and connected to the node transceiver,
- Also via a termination resistor and a diode to Vbat .



Physical properties

- On the bus:
  - Logical low level (0) is dominant
  - Logical high level (1) is recessive.
- Voltage supply (Vsup) for an ECU should be between 7 V and 18 V.



LIN Error Handling

- Each LIN Slave monitors its operating state and creates a status report.
- The status report is sent periodically to the LIN Master (LIN 2.0).
- Monitoring by error detection mechanisms
  - Parity check
  - Checksum
- LIN messages detected as corrupt are rejected
- Error handling is not part of the LIN specification and must be defined separately

LIN Power Management (V2.0)

- It contains "wake up" and "go-to sleep".
- All the slave nodes in an active LIN cluster can be changed into sleep mode by:
  - ✓ Sending a diagnostic master request frame with the first data byte equal to zero.
  - ✓ This special use of a diagnostic frame is called a go-to-sleep-command.
- Slave nodes can automatically enter a sleep mode if the LIN bus is inactive for more than 4 seconds.

LIN Power Management (V2.0)

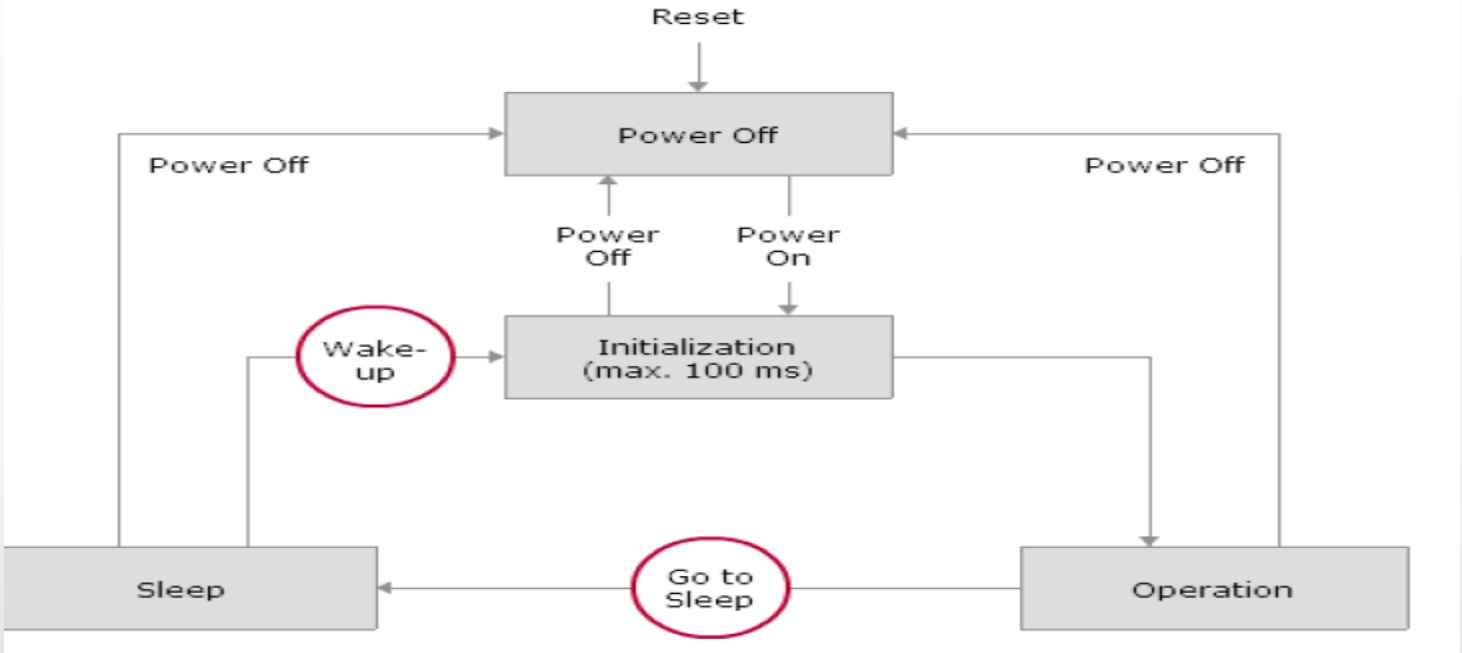
- Any node in a sleeping LIN cluster can send a request for wake up cluster.
- wakeup request is issued by forcing the bus dominant for 250 μs to 5 ms.
- Every slave node can detect the wake-up request (a dominant pulse longer than 150 ms) and be ready to listen to bus commands within 100 ms, measured from the ending edge of the dominant pulse.

LIN Power Management (V2.0)

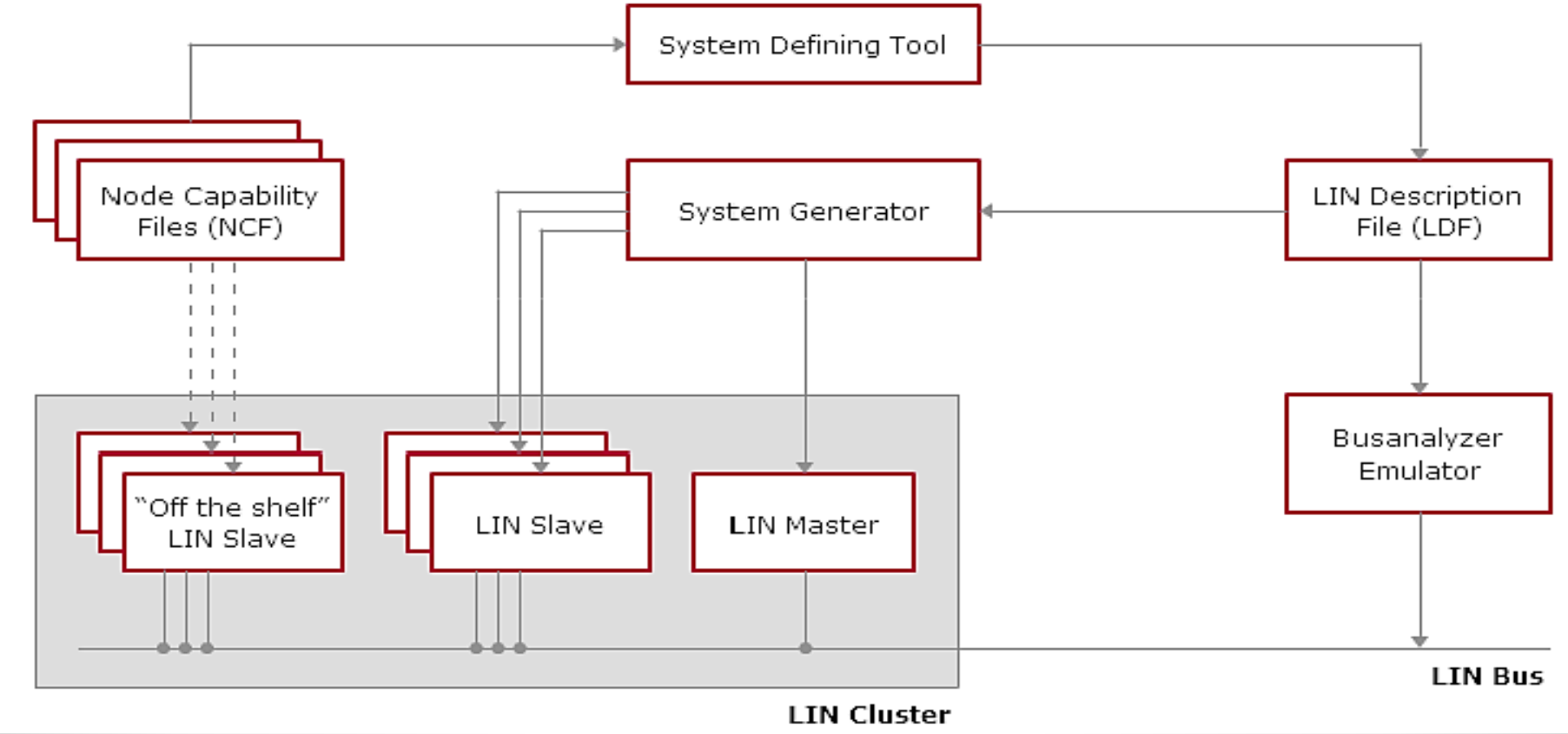
- The master node can wake up .
- When the slave nodes are ready, start sending frame headers to find out the cause of the wake up.
- If the master does not issue headers within 150 ms after receiving the first wakeup request, then the slave requesting wakeup may try issuing a second wakeup request (and waiting for another 150 ms).

LIN Power Management (V2.0)

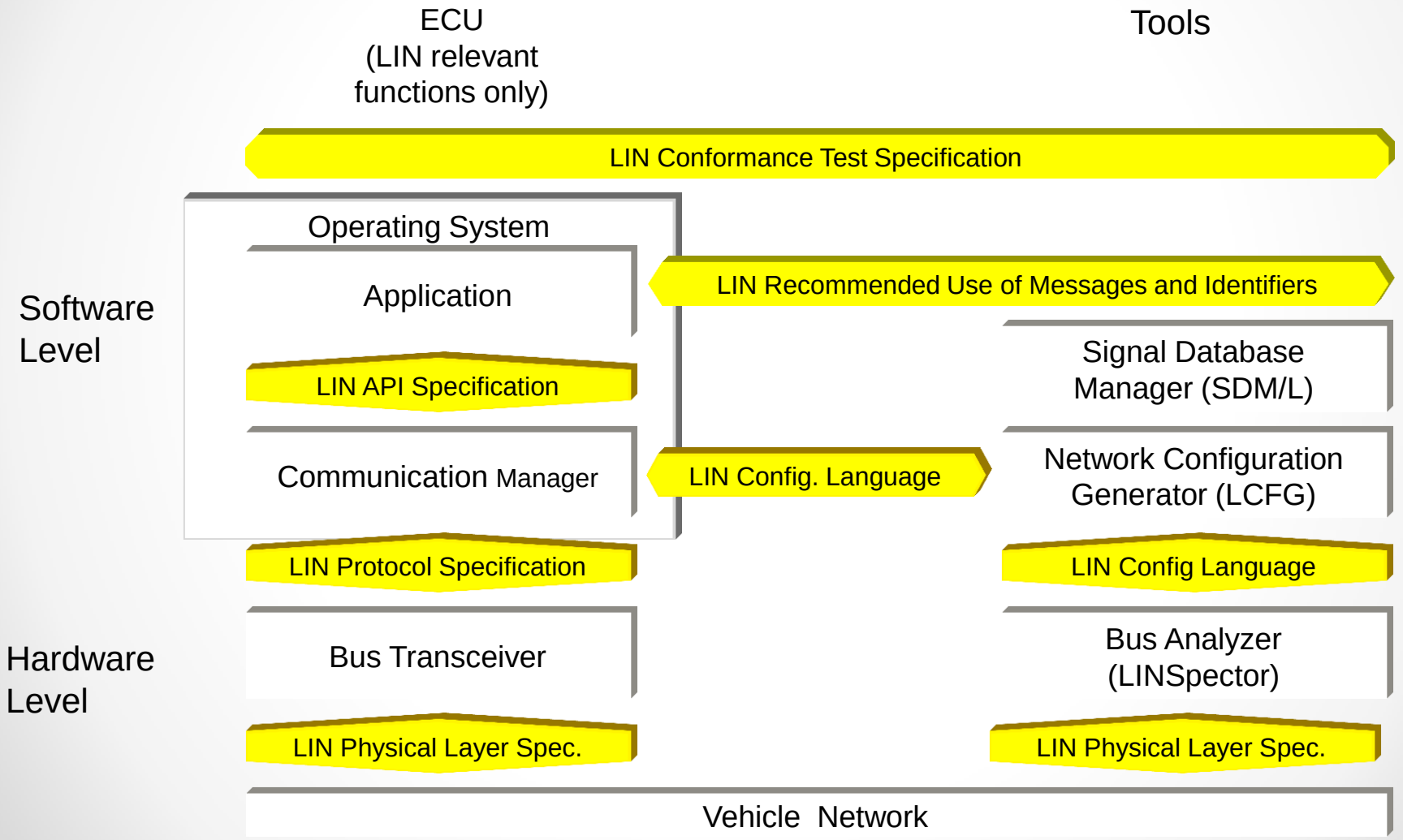
- If the master still does not respond, the slave issue the wakeup request and wait 150 ms a third time.
- If there is still no response, the slave must wait for 1.5 seconds before issuing a fourth wakeup request.



LIN Workflow



LIN Workflow

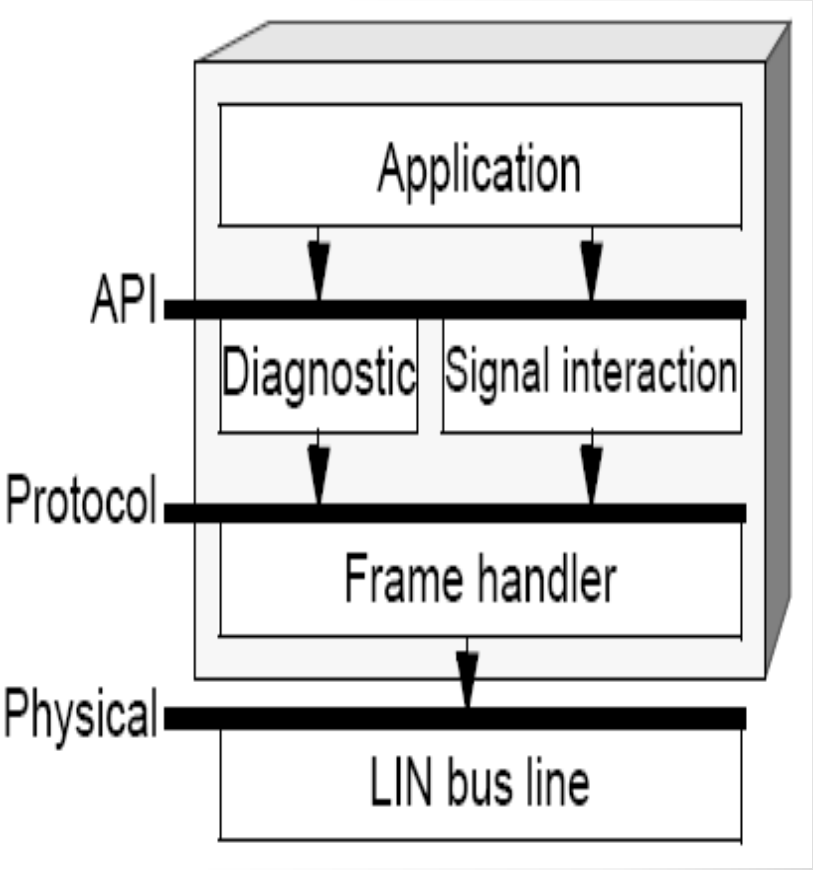


LIN Workflow

- The LIN bus connects a single master device (node) and one or more slave devices (nodes) together in a LIN cluster.
- The behavior of each node is described by its own node capability file(NCF).
- The node capability files are inputs to a system-defining tool
- It generates a LIN description file (LDF) that describes the behavior of the entire cluster.
- The LDF is parsed by a system generator to automatically generate the specified behavior in the desired nodes.

LIN Workflow

- A node interfaces to the physical bus wire using a frame transceiver.
- The frames are not accessed directly by the application.
- A signal based interaction layer is added in between.
- A diagnostic interface exist between the application and the frame handler, as depicted below.



References

www.vector.com

ESC automotive sessions

http://www.ixxat.com/introduction_lin_en.html

http://www.kvaser.com/zh/about-can/related-protocols-and-standards/46.html

http://www.eeherald.com/section/design-guide/esmod10.html