SURFACE VEHICLE INFORMATION REPORT

J3076™	OCT2015
Issued 2015-10	

Clock Extension Peripheral Interface (CXPI)

RATIONALE

The CXPI 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 CXPI 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 information in the implementation of low speed vehicle serial data network communications using the Clock Extension Peripheral Interface (CXPI) protocol.

The goal of this document is to provide an overview of the CXPI protocol describing the features and functions of the serial data physical layer, data link layer and application layer for use in the 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 the knowledge of the JASO Clock Extension Peripheral Interface (CXPI) specification.

TO PLACE A DOCUMENT ORDER: Tel: 877-606-7323 (inside USA and Canada)
Tel: +1 724-776-4970 (outside USA)
Fax: 724-776-0790
Email: CustomerService@sae.org
SAE WEB ADDRESS: http://www.sae.org

SAE values your input. To provide feedback on this Technical Report, please visit
http://www.sae.org/technical/standards/J3076_201510

TABLE OF CONTENTS

1. SCOPE

1.1 Mission/Theme

This document is an information report and intended to provide an overview of the Clock Extension Peripheral Interface (CXPI) protocol.

1.2 Overview

This document describes the features and functions of the CXPI protocol. The CXPI protocol provides some selected features of the Controller Area Network (CAN) protocol implemented on a UART-based data link for mainly HMI (Human Machine Interface) of road vehicles electric systems.

1.3 Relationship to JASO D015 CXPI Specification

This information report is a description of the CXPI protocol, which is specified in the JASO D015 CXPI document published by JASO. The JASO D015 CXPI specification is the normative reference for the CXPI protocol. The CXPI specification is maintained by JSAE (Society of Automotive Engineers of Japan, Inc.).

This information report does not supersede any information contained in the JASO D015 CXPI specification. It has the sole purpose of providing textual description and graphical illustrations to ease reading and interpretation of the CXPI protocol.

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 Documents

Available from SAE International, 400 Commonwealth Drive, Warrendale, PA 15096-0001, Tel: 877-606-7323 (inside USA and Canada) or +1 724-776-4970 (outside USA), www.sae.org.

SAE J2602/1 LIN Network for Vehicle Applications

SAE J2602/2 LIN Network for Vehicle Applications Conformance Test

SAE J2602/3 File Structures for a Node Capability File (NCF)

2.1.2 ISO Documents

Copies of these documents are available online at http://webstore.ansi.org/

ISO 11898-1 Road Vehicles — Controller Area Network (CAN) — Part 1: Data Link Layer and Physical Signaling

ISO 14230 Road Vehicles — Diagnostic Communication over K-Line (DoK-Line)

ISO 17987 Road Vehicles — Local Interconnect Network (LIN)

2.1.3 JASO Documents

Copies of these documents are available online at http://www.jsae.or.jp/index_e.php

JASO D015 (all parts), Automobiles —Clock Extension Peripheral Interface (CXPI)

3. DEFINITION OF TERMS

3.1 GLOSSARY

3.1.1 Controller

portion which controls processing of a data link layer

3.1.2 Inter Byte Space

time between bytes on the communication bus

3.1.3 Inter Frame Space

time between frames on the communication bus

3.1.4 Master Node

node in the system that has the function of schedule management and primary clock master

3.1.5 Clock Master

node that transmits the clock to communication bus

3.1.6 Primary Clock Master

node that becomes clock master

3.1.7 Secondary Clock Master

node that transmits the clock when primary clock master doesn't transmit the clock

3.1.8 Slave Node

each node other than master node connected within the CXPI communication system

3.1.9 Schedule

information specifying which frame is transmitted at which timing that becomes the origin of periodic transmission

3.1.10 Sequence

procedure of transmission and reception of the data among two or more nodes

3.1.11 Idle State

existence of the frame is not recognized on the communication bus. Only the clock exists, and it is possible to transmit frames

3.1.12 PTYPE Field

includes the parity bit of “1 bit” and the frame TYPE of 7 bits.

3.1.13 PID Field

includes the parity bit of “1 bit” and the frame ID of 7 bits.

3.1.14 Nominal Baud Rate

baud rate within the protocol specification range and target value of baud rate actually used

3.1.15 Operational State

Supported wakeup/sleep; in the state of normal mode, non-supported wakeup/sleep; in the state of power-on

4. ACRONYMS, ABBREVIATIONS, AND SYMBOLS

4.1 Abbreviations

AP	Application Layer
CAN	Controller Area Network
CR	Collision Resolution
CRC	Cyclic Redundancy Check
CSMA	Carrier Sense Multiple Access
CT	Counter
CXPI	Clock eXtension Peripheral Interface
DLC	Data Length Code
DLL	Data Link Layer
ECU	Electronic Control Unit
FET	Field Effect Transistor
HI	High
HMI	Human Machine Interface
IBS	Inter Byte Space
ID	Identifier
IFS	Inter Frame Space
Ind	Indicator
JSAE	Society of Automotive Engineers of Japan, Inc.
LIN	Local Interconnect Network
LO	Low
LSB	Least Significant Bit
MSB	Most Significant Bit
NRZ	Non Return to Zero
NM	Network Management

OP Option

OSI Open Systems Interconnection

PID Protected ID

PL Physical Layer

PTYPE Protected TYPE

PWM Pulse Width Modulation

RX Receive (data)

RXD RXD pin of the Transceiver

TH Threshold

TTL Transistor-Transistor-Logic

TX Transmit (data)

TXD TXD pin of the Transceiver

UART Universal Asynchronous Receiver Transmitter

4.2 Symbols

kbit/s Kilobit per second

$RX_{(bit)}$ Receive (bit)

T_{bit} Nominal bit time

$TX_{(bit)}$ Transmit (bit)

V/µs Voltage per microsecond

5. CXPI PROTOCOL PRINCIPLES

5.1 Features

The CXPI protocol has been defined with the objective to combine significant features of CAN onto a UART-based data link with a Carrier Sense Multiple Access (CSMA) and Collision Resolution (CR) physical layer transceiver to improve the data exchange communication between ECUs compared to typical UART-based communication e.g., RS 232, ISO 14230 K-Line, ISO 17987 Local Interconnect Network (LIN).

The following is an overview of the CXPI protocol features:

- Carrier Sense Multiple Access (CSMA): the CXPI UART transceiver performs bit level arbitration,
- Collision Resolution (CR): the CXPI UART transceiver performs bit level collision resolution (avoidance),
- Bus master clock synchronization (Pulse Width Modulation (PWM) bit representation) of all nodes on the CXPI communication bus to trigger the slave nodes communication,
- Cyclic Redundancy Check (CRC) at the end of each frame,

- Single Master and multiple Slave Node Polling Schedules,
- Polling method to support periodic schedules,
- Event method to support the responsiveness of slave node communication,
- Sleep and Wakeup support,
- Maximum baud rate of 20 kbit/s,
- Up to 16 nodes connected to a CXPI communication bus,

In addition to above feature list very small CXPI devices (nodes) may be equipped without a UART and microcontroller. Those devices can be designed to support the CXPI protocol with a Pulse Width Modulation (PWM)-based communication which is compatible with the CXPI UART-based communication.

5.2 OSI Layer Structure

The CXPI protocol is compliant to the JIS X 0026: 1995, Glossary of terms used in information processing (Open Systems Interconnection). Figure 1 shows the CXPI layer structure. The Presentation, Session, Transport and Network Layers are not part of this Information Report.

Application Layer	**Frame Transfer Management** **Wakeup / Sleep** **Frame ID Assignment**
Presentation Layer	**Not Applicable**
Session Layer	**Not Applicable**
Transport Layer	**Not Applicable**
Network Layer	**Not Applicable**
Data Link Layer	**Management of Retransmission** **Transmission Processing** **Reception Processing**
Physical Layer	**Encoding, Decoding** **Switching Signal Encoding/Decoding** **Clock Synchronization** **Clock Output**

Figure 1 - CXPI layer structure

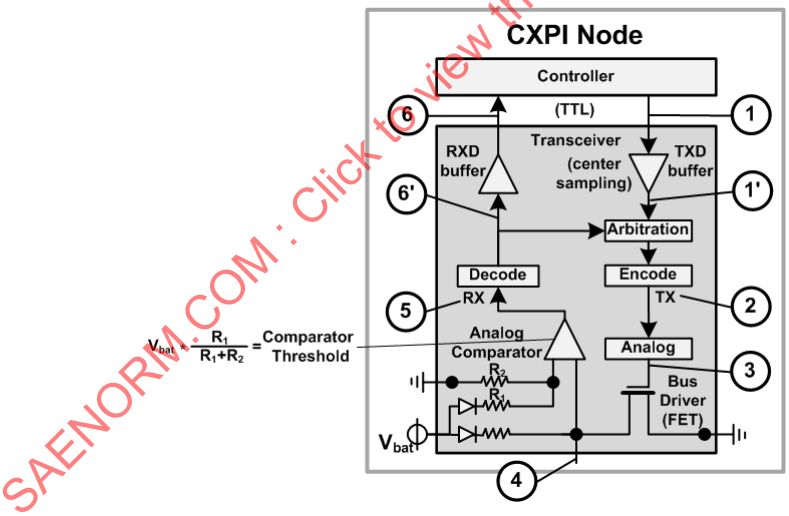
5.3 CXPI Transceiver Functionality

5.3.1 Transceiver Block Diagram and Signal References

The block diagram of the CXPI Transceiver with the signal references is used as a reference of all timing diagrams in the following chapters. Each CXPI node implements a CXPI Transceiver which consists of the following function blocks:

- (1): TXD signal from Controller: TXD buffer buffers a sending bit sent from the Controller (TTL level signal);
- Arbitration Unit: receives the sending bit from the TXD buffer and performs an arbitration (low bit wins) according to the Carrier Sense Multiple Access (CSMA) and Collision Resolution (CR) method;
- (2) TX Encode Unit: TTL signal after arbitration generated as PWM signal which will be transmitted;
- (3) TX Analog Unit with bus driver (FET): TX analog signal as input to the bus driver (FET);
- (4) CXPI bus signal: Physical analog waveform of the CXPI bus;
- (5) RX Decode Unit: TTL signal generated as output from the comparator as PWM signal;
- (6') Decoded RX signal: TTL signal generated from PWM signal as input for Arbitration Unit;
- (6) RXD signal to Controller: RXD buffer buffers the received bit (none arbitrated bits) and transmits to the Controller for software comparison with original transmitted byte;

Figure 2 shows the block diagram and signal references.



No	
1	CXPI Node Controller TXD line which transmits bytes from the UART in TTL level
1'	CXPI Node Transceiver TXD line which shows ½ x T_{bit} delay because of center sampling at (1)
2	Node Encoder logic which performs conversion of NRZ signal from TXD buffer to PWM signal
3	Node Transceiver analog circuit shapes the TTL into an analog signal
4	CXPI communication bus (analog signal waveform is interpreted by all connected nodes)
5	Actual CXPI communication bus analog RX signal waveform received by all nodes to be decoded
6	Controller RXD line which receives bytes from the RXD buffer in TTL level
6'	Controller RXD line which receives bytes from the Decoder logic

Figure 2 - Block diagram and signal references

The block diagram shown in Figure 2 is common for all nodes. But there are two differences between a master and a slave node transceiver which are not illustrated in Figure 2.

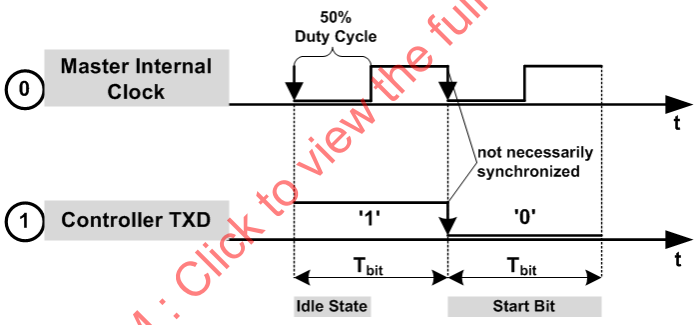
- The master node transceiver operates based on the clock master (the clock supplied by a controller which is called master internal clock). On the other hand, the slave node transceiver operates based on the clock generated by detecting the falling edge of the CXPI bus signal (see RX signal (5) in Figure 2). Therefore, all slave transceivers operate synchronously, but slightly delayed from the master internal clock.

- In the master node transceiver, the TX signal (2) is generated to drive the CXPI bus for both, logic '1' and logic '0' states. In the slave node transceiver, the TX signal (2) is generated to drive the CXPI bus only when the CXPI bus is to be driven to dominant level (logic '0') to overwrap the slave signal on the CXPI bus master clock (logic '1' PWM signal generated by the master node). The reason is that only the master node outputs the bus master clock with a TXD signal (1).

5.3.2 Start Bit Falling Edge Timing

The master internal clock is generated by the master node. The master internal clock signal is a TTL signal with a 50% duty cycle (25 µs logic '0' and 25 µs logic '1', e.g. @20 kbps baud rate). The master internal clock triggers the bitwise transmission of the Controller transmitted frame bytes. On each falling edge of the master internal clock signal the logic state of the Controller TXD transmitted bit is started. Based on the physics of the components in the transceiver there might be a very small delay of the Controller TXD start bit timing compared to the master internal clock falling edge which is, for simplicity, not shown in the timing diagram.

Figure 3 shows the start bit falling edge timing.



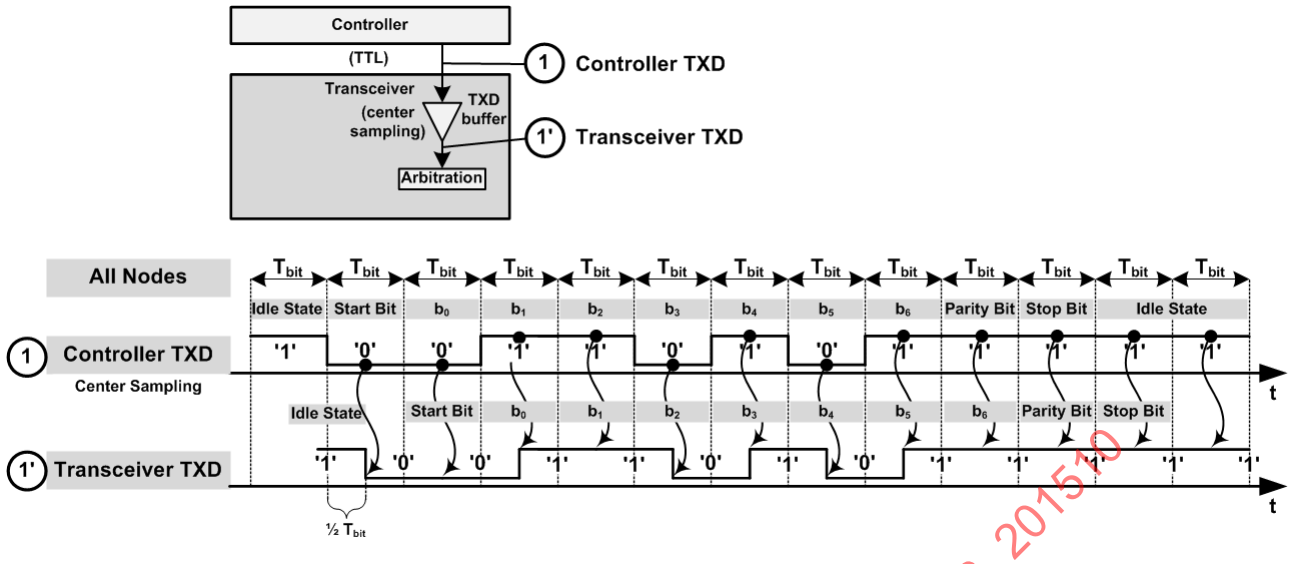
No	
0	CXPI Master Node Clock
1	CXPI Node Controller TXD line which transmits bytes from the UART in TTL level

Figure 3 - Start bit falling edge timing

5.3.3 Internal TXD Signal Delay (relevant to all nodes)

The Controller TXD signal (1) is not directly connected to the Arbitration Unit but latched (sampled and hold) to become a Transceiver TXD signal (1'). The Transceiver TXD signal (1') is sampled around the center of the UART bit. Be aware of that it is not the center of the master internal clock bit. Thus the Transceiver TXD signal (1') is delayed from the Controller TXD signal (1) by (1/2) x T_{bit}.

Figure 4 shows the internal TXD signal delay.



No

1 CXPI Node Controller TXD line which transmits bytes from the UART in TTL level

1' CXPI Node Transceiver TXD line which shows ½ x T_{bit} delay because of center sampling at (1)

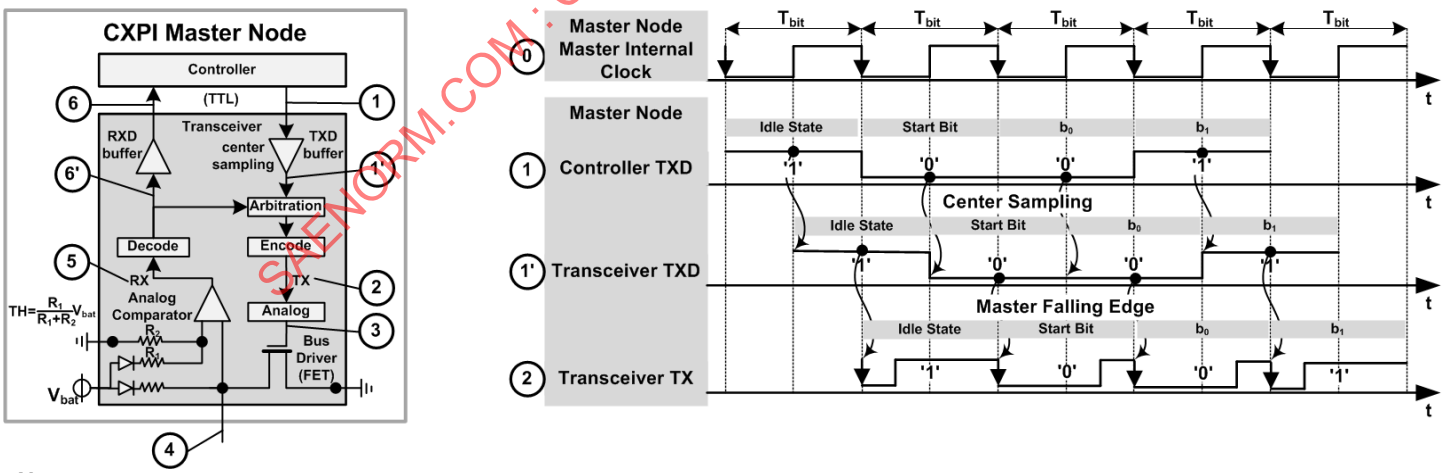
Figure 4 - Internal TXD signal delay

5.3.4 TX signal Generation

5.3.4.1 Master Node

The Transceiver TXD signal (1') bit is encoded into the Transceiver TX signal (2) at the next falling edge of the master internal clock signal after the Controller TXD signal (1) is latched. The falling edge of the Transceiver TX signal (2) is triggered by the sampling point of the Transceiver TXD signal (1') which is triggered by the falling edge of the master internal clock signal.

Figure 5 shows the internal TX signal generation (Master Node).



No

0 CXPI Master Node Clock

1 CXPI Master Node Controller TXD line which transmits bytes from the UART in TTL level

1' CXPI Master Node Transceiver TXD line which shows ½ x T_{bit} delay because of center sampling at (1)

2 Master Node Encoder logic which performs conversion of NRZ signal from TXD buffer to PWM signal

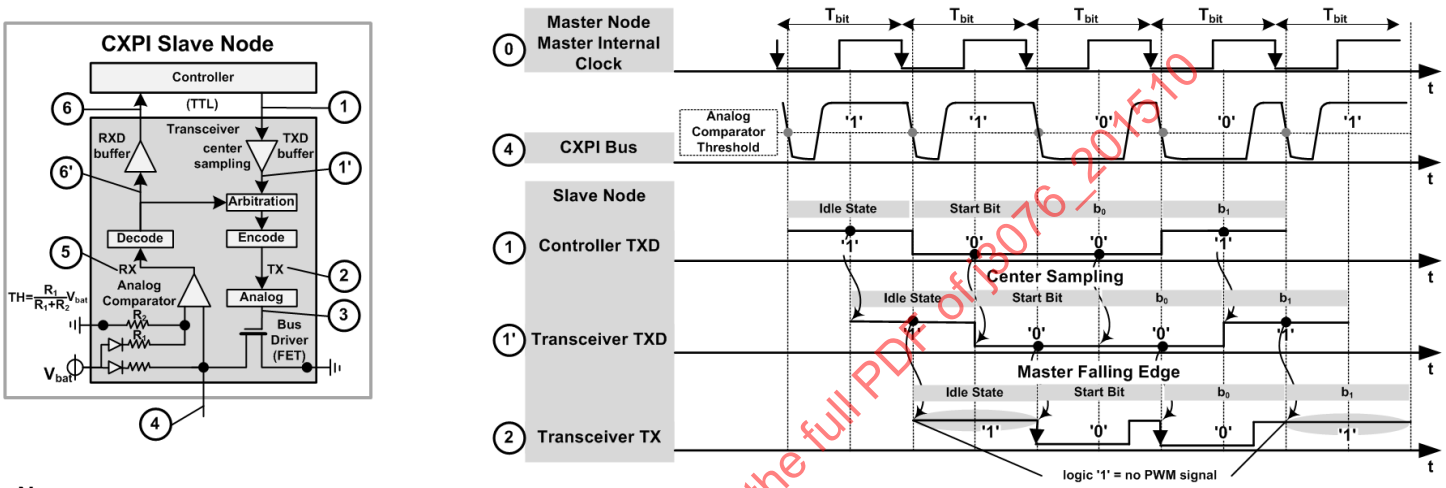
Figure 5 - Internal TX signal generation (Master node)

5.3.4.2 Slave Node

The Transceiver TXD signal (1') bit is encoded into the Transceiver TX signal (2) at the next falling edge of the operation clock signal after the Controller TXD signal (1) is latched. The falling edge of the Transceiver TX signal (2) is triggered by the sampling point of the Transceiver TXD signal (1') which is triggered by the falling edge of the operation clock signal.

Note, the operation clock signal, which is generated inside the slave transceiver is the clock for internal operation detecting the falling edge of the CXPI bus.

Figure 6 shows the Internal TX signal generation (Slave Node).



No

0 CXPI Master Node Clock

1 CXPI Slave Node Controller TXD line which transmits bytes from the UART in TTL level

1' CXPI Slave Node Transceiver TXD line which shows ½ x T_{bit} delay because of center sampling at (1)

2 Slave Node Encoder logic which performs conversion of NRZ signal from TXD buffer to PWM signal

4 CXPI communication bus (analog signal waveform is interpreted by all connected nodes)

Figure 6 - Internal TX signal generation (Slave Node)

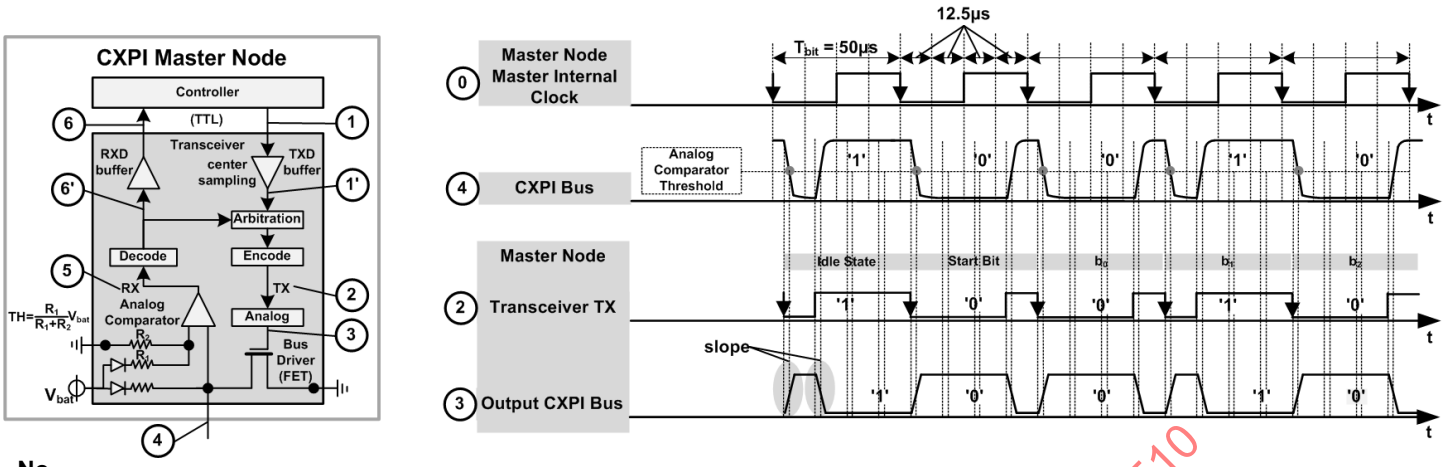
IMPORTANT: During Idle State and when output data is in logic '1' state, the TX signal is not generated (it stays in logic '1' state, not PWM-encoded).

5.3.5 Internal Analog Circuitry Delay

5.3.5.1 Master Node

The analog circuitry delay (delay between Transceiver TX (2) and Output CXPI Bus signal (3)) shall be asserted by the slope of several µs width to stabilize the FET switching (refer to the wave form (3) in Figure 7).

Figure 7 shows the internal analog circuitry delay (Master Node).



No

0 CXPI Master Node Clock

2 Master Node Encoder logic which performs conversion of NRZ signal from TXD buffer to PWM signal

3 Master Node Transceiver analog circuit shapes the TTL into an analog signal

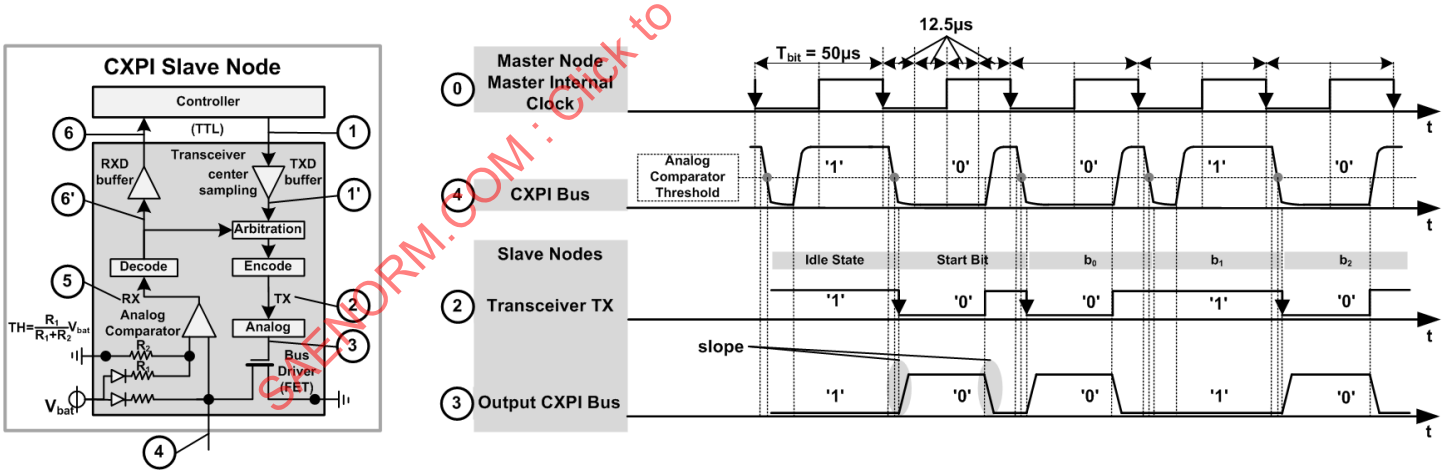
4 CXPI communication bus (analog signal waveform is interpreted by all connected nodes)

Figure 7 - Internal analog circuitry delay (MASTER NODE)

5.3.5.2 Slave Node

The analog circuitry delay (delay between Transceiver TX (2) and Output CXPI Bus signal (3)) shall be asserted by the slope of several µs width to stabilize the FET switching (refer to the wave form (3) in Figure 8).

Figure 8 shows the internal analog circuitry delay (Slave Node).



No

0 CXPI Master Node Clock

2 Slave Node Encoder logic which performs conversion of NRZ signal from TXD buffer to PWM signal

3 Slave Node Transceiver analog circuit shapes the TTL into an analog signal

4 CXPI communication bus (analog signal waveform is interpreted by all connected nodes)

Figure 8 - Internal analog circuitry delay (Slave Node)

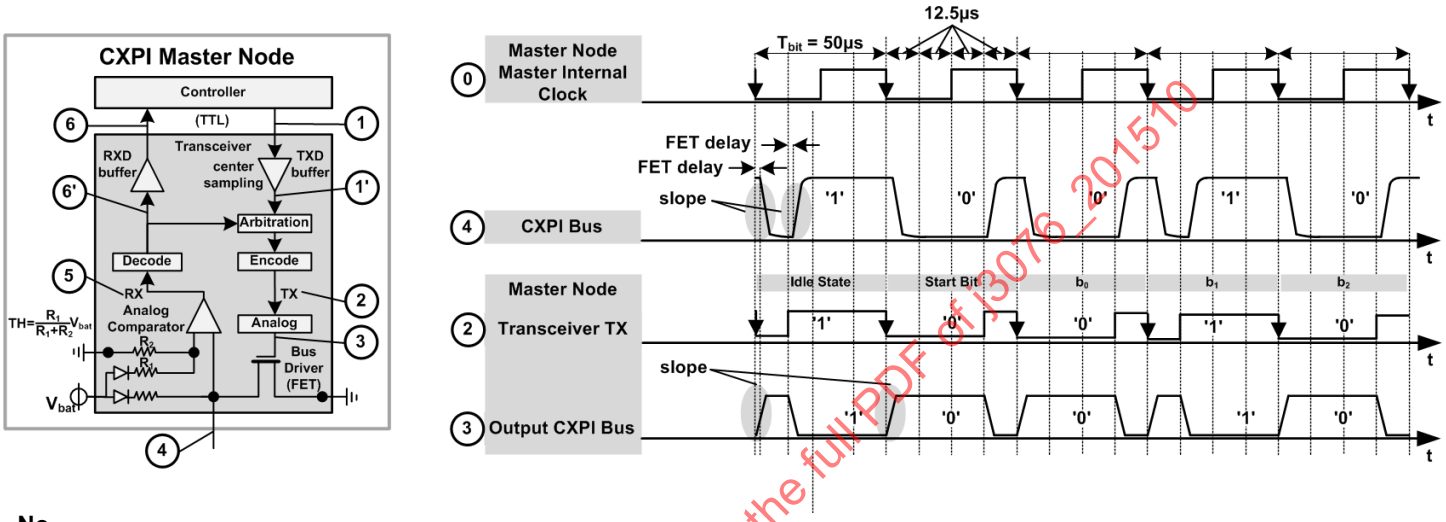
5.3.6 Internal FET Switching Delay

5.3.6.1 Master Node

The FET transistor switching delay (delay between Output CXPI Bus (3) and (CXPI Bus signal 4)) is very small (< 1 µs). This delay is almost not visible in the timing diagrams because of the resolution.

The wave form CXPI Bus signal (4) is affected by the floating capacitance of the CXPI bus.

Figure 9 shows the internal FET switching delay (Master Node).



No

0 CXPI Master Node Clock

2 Master Node Encoder logic which performs conversion of NRZ signal from TXD buffer to PWM signal

3 Master Node Transceiver analog circuit shapes the TTL into an analog signal

4 CXPI communication bus (analog signal waveform is interpreted by all connected nodes)

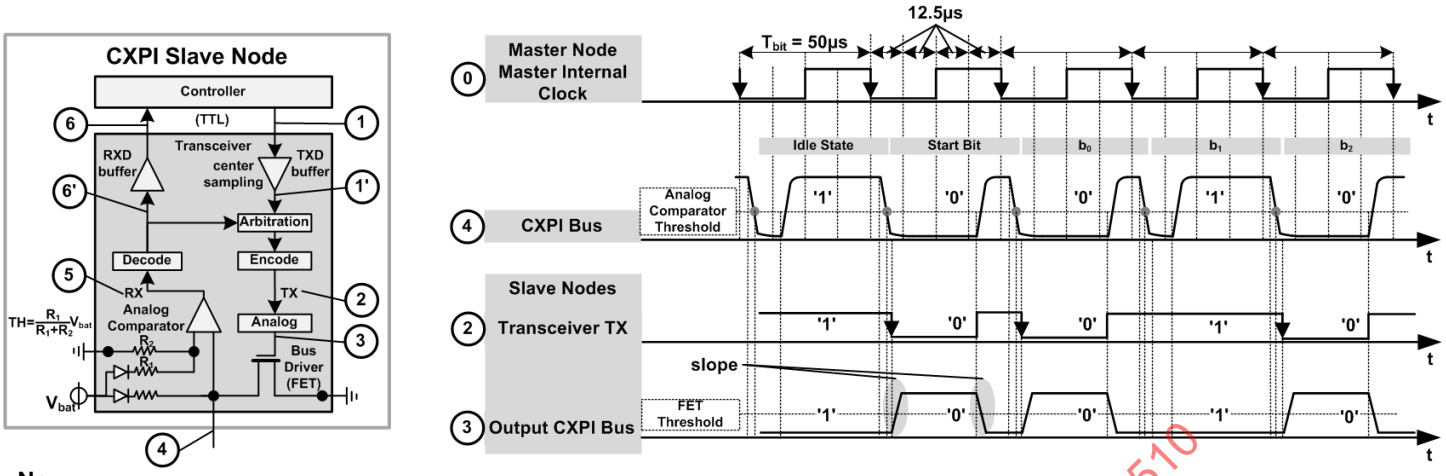
Figure 9 - Internal FET switching delay (Master Node)

5.3.6.2 Slave Node

The FET transistor switching delay (delay between Output CXPI Bus (3) and CXPI Bus signal (4)) is very small (< 1 µs). This delay is almost not visible in the timing diagrams because of the resolution.

The wave form CXPI Bus signal (4) is affected by the floating capacitance of the CXPI bus.

Figure 10 shows the Internal FET switching delay (Slave Node).



No

0 CXPI Master Node Clock
2 Slave Node Encoder logic which performs conversion of NRZ signal from TXD buffer to PWM signal
3 Slave Node Transceiver analog circuit shapes the TTL into an analog signal
4 CXPI communication bus (analog signal waveform is interpreted by all connected nodes)

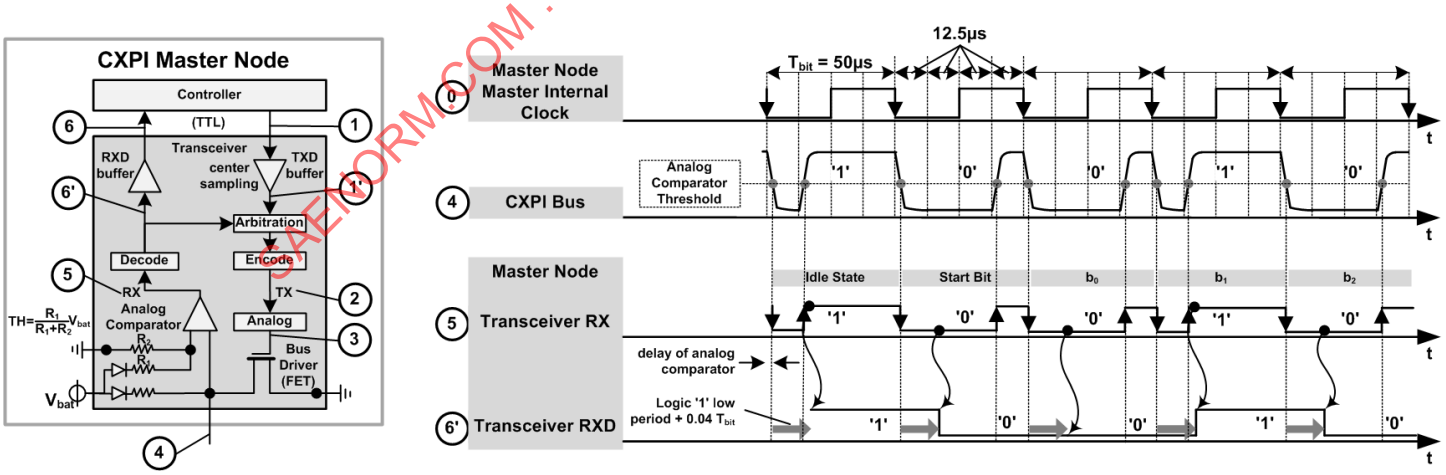
Figure 10 - Internal FET switching delay (Slave Node)

5.3.7 Internal RXD Signal Generation

5.3.7.1 Master Node

The Transceiver RX signal (5) is the PWM TTL signal received from the output of the analog comparator. The RX Decoder Unit converts the PWM encoded bits into non-PWM encoded bits and outputs those as the Transceiver RXD signal (6'). Those bits are used as input to the Arbitration Unit and the RXD buffer. The sampling point to determine a logic '1' or '0' state of the Transceiver RXD signal (6') equals the Logic '1' low period + 0.04 x T_{bit}. An example is included in the JASO D015-3, Automobiles —Clock Extension Peripheral Interface (CXPI) — Part 3: Protocol Specification.

Figure 11 shows the internal RXD signal generation (Master Node).



No

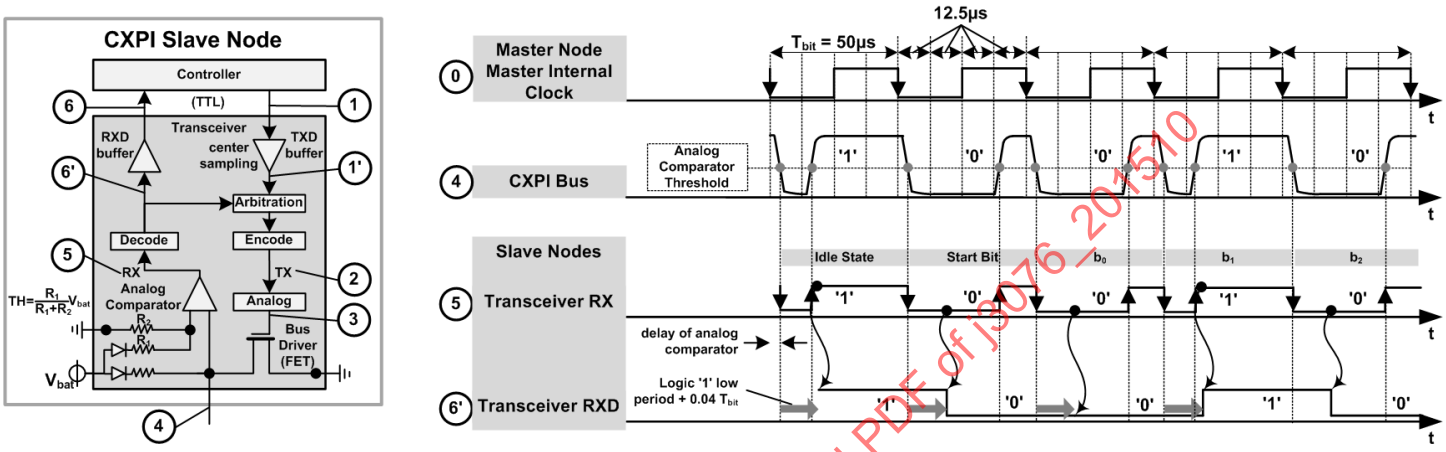
0 CXPI Master Node Clock
4 CXPI communication bus (analog signal waveform is interpreted by all connected nodes)
5 Actual CXPI communication bus analog RX signal waveform received by all nodes to be decoded
6 Controller RXD line which receives bytes from the RXD buffer in TTL level
6' Controller RXD line which receives bytes from the Decoder logic

Figure 11 - Internal RXD signal generation (Master node)

5.3.7.2 Slave Node

The Transceiver RX signal (5) is the PWM TTL signal received from the output of the analog comparator. The RX Decoder Unit converts the PWM encoded bits into non-PWM encoded bits and outputs those as the Transceiver RXD signal (6'). Those bits are used as input to the Arbitration Unit and the RXD buffer. The sampling point to determine a logic '1' or '0' state of the Transceiver RXD signal (6') equals the Logic '1' low period + 0.04 x T_{bit}. An example is included in the JASO D015-3, Automobiles —Clock Extension Peripheral Interface (CXPI) — Part 3: Protocol Specification.

Figure 12 shows the Internal RXD signal generation (Slave Node).



No

0 CXPI Master Node Clock
4 CXPI communication bus (analog signal waveform is interpreted by all connected nodes)
5 Actual CXPI communication bus analog RX signal waveform received by all nodes to be decoded
6 Controller RXD line which receives bytes from the RXD buffer in TTL level
6' Controller RXD line which receives bytes from the Decoder logic

Figure 12 - Internal RXD signal generation (Slave Node)

6. PROTOCOL DESCRIPTION

6.1 Application Layer

6.1.1 Overview

The application layer is an abstraction layer reserved for communications protocols and methods designed for process-to-process communications across the CXPI network. The application layer coordinates frame exchange for the software running on a CXPI node. The CXPI protocol at the application layer handles the requests that the different node software applications are making to the CXPI network.

The CXPI application layer supports the following network functions:

- Master / Slave Communication
- Frame Transfer Management
- Frame ID Assignment
- Wakeup / Sleep

6.1.2 Master / Slave Communication

The CXPI protocol is based on a single master and multiple slave (maximum 15 slave nodes) communication system with a maximum of 40[m] cable length and features like clock synchronization, polling method (PID or PTYPE field request, response) and event frame transmission. Each node uses a CXPI-compliant transceiver which encodes/decodes each bit into a Pulse Width Modulated (PWM) physical signal, typically 0 – 12 V, on a single wire with reference to ground potential.

6.1.3 Frame Transfer Management

6.1.3.1 General

The Frame Transfer Management is one of the core functions of the CXPI protocol. The principle of a frame transfer consists of two (2) steps:

- First step = Request:
  - **Method 1 – Event Trigger Method:**
    The Master Node or Slave Nodes which have internally registered an event occurrence issue a "PID field" request. This action always takes place during a contention / arbitration phase.
  - **Method 2 – Polling Method**:
    Only the Master Node can issue a "PID field" and "PTYPE field" request according to the Master Node's schedule (table). A contention / arbitration phase is only present if the Master Node has previously sent a PTYPE field.
- Second step = Response (or Action): Reaction of the node for requested PID field:
  The response (Frame information, Data, CRC) is issued by the node (Master or Slave Nodes) which is uniquely associated (or related) to generate a response corresponding to the requested PID field.

The CXPI protocol supports two (2) methodologies about how and when data is transferred in a frame. Only one of the two methods can be implemented in all nodes connected to the CXPI communication bus.

- **Method 1 – Event Trigger Method:**
  The Master Node issues a PID field repeatedly and periodically (Polling). The PID field is always issued during a contention / arbitration phase. If the master node wins arbitration then the node which supports the PID field issues a response. Even during master polling, the Master Node and all Slave Nodes are allowed to issue a PID field on the communication bus if they have previously registered an internal event occurrence. The node, associated with the PID field of the node which wins arbitration during the contention phase based on the CSMA/CR access method, will issue the response. See Figure 13 for further details.
- **Method 2 – Polling Method**:
  The Master Node issues a PID field or a PTYPE field repeatedly and periodically (Polling). When the Master Node sends a PID field no contention / arbitration phase is active in the Polling Method. Consequently, if the Master Node has issued a PID field the node, which supports the PID field, issues a response frame.
  When the Master Node sends a PTYPE field the contention / arbitration phase becomes active in the Polling Method within IBS time. All nodes, which have previously registered an event occurrence, will issue a PID field. The node with the highest priority PID field wins the arbitration based on the CSMA/CR access method and the node associated with the PID field issues a response. During the Polling Method the Master Node controls the response exchange on the communication bus. See Figure 14 for further details.

The difference between the Event Trigger Method and Polling Method is, that the Polling Method requires the Master Node to control the event driven response transmission through a previously transmitted PTYPE field request, which allows all nodes to issue an event driven PID field on the communication bus. During a Polling Method a contention / arbitration phase will only become active after the master node has sent a PTYPE field within the specified IBS time. The node, associated with the PID field of the node which wins arbitration based on the CSMA/CR access method, will issue the response.

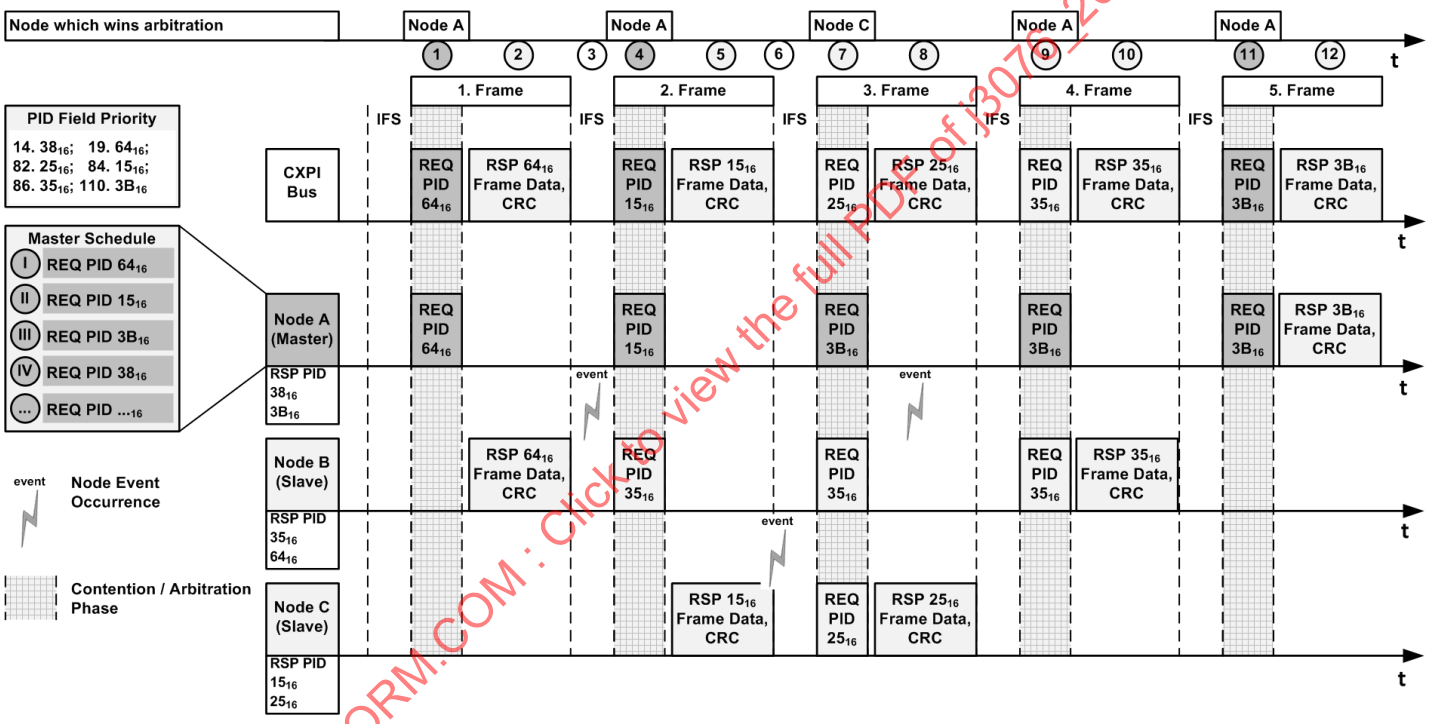
As general information, the following design guidelines shall be followed:

- Either Method 1 or Method 2 shall be chosen at implementation time for all nodes which will be connected to the CXPI communication bus.
- The nodes operating based on Method 1 and Method 2 are not mutually connected to one network domain.

6.1.3.2 Method 1 – Event Trigger Method

The Event Trigger Method is enabled in the master and slave nodes as soon as the idle state of the communication bus is detected. If a node has detected an internal event, which requires a frame to be sent onto the communication bus, it first waits until IFS has expired and then immediately sends a PID field request within the IBS. If the node wins arbitration based on the CSMA/CR access method, the node associated with the PID field will issue the response. The response includes the frame information, data and the CRC.

See Figure 13 for a detailed Method 1 – Event Trigger Method Example.



No

1 Node A (Master) requests PID 64_{16} according to the Master Node Schedule.
2 Node B (Slave) sends 64_{16} Response (Frame information, Data, CRC).
3 Node B (Slave) has internally registered an event trigger.
4 Node A (Master) requests PID 15_{16} according to the Master Node Schedule and Node B (Slave) requests PID 35_{16} (event).
5 Node C (Slave) sends
6 Node C (Slave) has internally registered an event trigger.
7 Node A (Master) requests PID $3B_{16}$ according to the Master Node Schedule, Node B (Slave) requests PID 35_{16} (retransmission, based on event) and Node C (Slave) requests PID 25_{16} (event).
8 Node C (Slave) sends 25_{16} Response (Frame information, Data, CRC) based on the fact that it has won arbitration (see 7). Node B (Slave) has internally registered a new event trigger.
9 Node A (Master) requests PID $3B_{16}$ according to the Master Node Schedule, Node B (Slave) requests PID 35_{16} (retransmission, based on event) (see 7).
10 Node B (Slave) sends 35_{16} Response (Frame information, Data, CRC) based on the fact that it has won arbitration (see 9).
11 Node A (Master) requests PID $3B_{16}$ (retransmission) according to the Master Node Schedule.
12 Node A (Master) sends $3B_{16}$ Response (Frame information, Data, CRC).

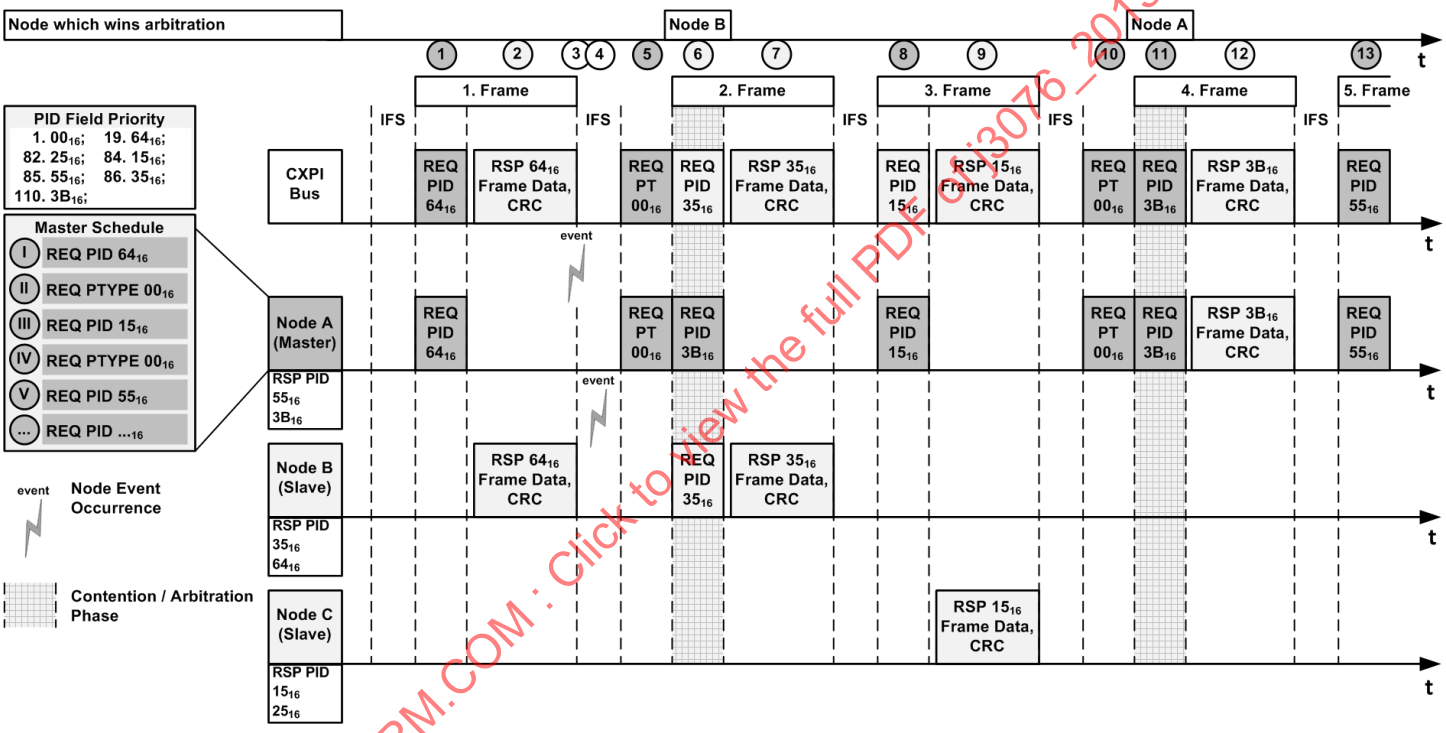
Figure 13 - Event Trigger Method example

6.1.3.3 Method 2 – Polling Method

The Polling Method can only be scheduled by the Master Node. Each time an IFS is detected, the Master Node either issues a PID field or PTYPE field request on the communication bus. Whether the Master Node schedules a PID field or a PTYPE field is the designer's responsibility and therefore part of the beforehand definition of the Master Node Schedule table.

In case the PTYPE field is issued by the Master Node and has been detected by the Master Node and all Slave Nodes, all nodes (including the master node) which have previously registered an internal event occurrence are now authorized (or allowed) to issue their own PID field to place their requests on the communication bus. Therefore more than one node may issue the PID field at the same time (contention / arbitration phase). If a collision occurs arbitration is performed as a natural result. The node (including the Master Node) which is uniquely associated (or related) to generate a response corresponding to the PID field which has won arbitration shall generate the response and issue it on the communication bus.

See Figure 14 for a detailed Method 2 – Polling Method Example.



No

1 Node A (Master) requests PID 64_{16} according to the Master Node Schedule.
2 Node B (Slave) sends 64_{16} Response (Frame information, Data, CRC).
3 Node A (Master) has internally registered an event trigger.
4 Node B (Slave) has internally registered an event trigger.
5 Master Node requests PTYPE 00_{16} according to the Master Node Schedule (all nodes can send response frame based if event has occurred).
6 Node A (Master) requests PID $3B_{16}$ (event) and Node B (Slave) requests PID 35_{16} (event).
7 Node B (Slave) sends 35_{16} Response (Frame information, Data, CRC) based on the fact that it has won arbitration (see 6).
8 Master Node requests PID 15_{16} according to the Master Node Schedule.
9 Node C (Slave) sends 15_{16} Response (Frame information, Data, CRC).
10 Master Node requests PTYPE 00_{16} according to the Master Node Schedule (all nodes can send response frame if an internal event has occurred).
11 Node A (Master) requests PID $3B_{16}$ based on the previously registered event (see 3) and loosing arbitration (see 6).
12 Node A (Master) sends $3B_{16}$ Response (Frame information, Data, CRC).
13 Node A (Master) requests PID 55_{16} according to the Master Node Schedule.

Figure 14 - Polling Method example

6.1.3.4 Master Node Frame Schedule

The purpose of the Master Node Frame Schedule is to manage CXPI communication bus utilization during either the Event Triggered Method (method 1) or the Polling Method (method 2). The chosen implementation method is valid for all nodes. The schedule for requesting and transmitting frames of all nodes on the CXPI communication bus is defined in the Master Node Frame Schedule.

The schedule is partitioned into two (2) time slots:

- Master Node controlled response transmission (Master and Slave Nodes PID field request, Master or Slave Nodes response);
- Master Node controlled polling transmission (Master Node PTYPE field, master or all slave nodes with event triggers transmit PID field);

6.1.4 Frame ID Assignment

Each CXPI frame has a frame identifier (Frame ID) which can be assigned any data without reserved Frame ID. A CXPI frame identifier consists of seven (7) bits (bit 0 through 6) which is the value of the identifier. A CXPI frame identifier is complemented with a Parity Bit (bit 7). Such complemented frame is called the PID field. The Parity Bit is added to provide transmission security between sender and receiver of a PID field.

The PTYPE field is a special PID field with a frame ID of 00_{16} and a Parity Bit of 1_b. The PTYPE field is only used in the Polling Method. The PTYPE field is only used by the CXPI master node when polling all nodes (including itself) simultaneously, for event data. Following a PTYPE request, any node with event data to be transmitted, sends a request PID and corresponding response frame, provided it wins the CXPI bus in arbitration phase. The PTYPE field may be used by the master node repetitively in its polling schedule to request event data from all nodes simultaneously..

The assignment of Frame IDs is specified in the JASO D015-3, Automobiles — Clock Extension Peripheral Interface (CXPI) — Part 3: Protocol Specification:

- Assigned Frame IDs: PTYPE field (Polling Method), Sleep frame, Request and Response frames for Inspection and Diagnostics communication.
- Available Frame IDs (sorted by priority) for CXPI network design usage. Any type of payload data can be assigned to those Frame IDs.

6.1.5 Wakeup / Sleep

6.1.5.1 General

The Wakeup/Sleep function of the CXPI protocol is a feature to control power consumption of each CXPI node. If there is no need to exchange information between CXPI nodes all nodes would be transitioned into the Sleep state. In case information exchange is required CXPI nodes will be woken up.

6.1.5.2 Wakeup / Sleep Processing

The Wakeup/Sleep function of the CXPI protocol supports three (3) operating modes:

a. Sleep mode,

b. Standby mode,

c. Normal mode.

Figure 15 shows the Sleep, Standby and Normal mode management.

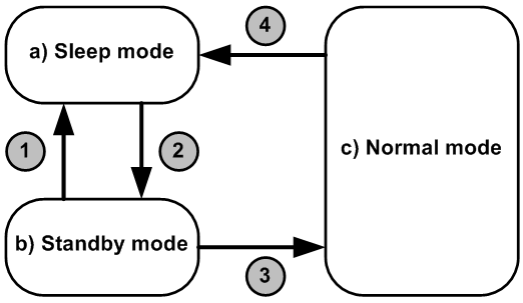


Figure 15 - Sleep, standby and normal mode management

The three (3) operating modes can be described as follows:

a. Sleep mode: No communication, Low power consumption, no decoding and encoding activated;

   1. Standby to Sleep mode: A CXPI node transitions into Sleep mode in case physical bus errors occur.

   2. Sleep to Standby mode: A CXPI node transitions into Standby mode in case a Wakeup pulse has been received or an internal event occurred.

b. Standby mode: Waiting condition to transition to Normal mode, Send a Wakeup pulse (if necessary), no decoding and encoding activated;

   3. Standby to Normal mode: The CXPI master node starts sending the bus master clock to the CXPI bus. The CXPI slave nodes have received the bus master clock.

c. Normal mode: All CXPI nodes enable reception, the CXPI master node enables the transmitter, the CXPI slave node enables the transmitter after receiving the first PID field from the CXPI master node, decoding and encoding activated;

   4. Normal to Sleep mode:

      The CXPI master node transmits the Sleep permission (Sleep frame) and transitions into Sleep mode if the Sleep conditions are satisfied.

      The CXPI slave nodes have received the Sleep frame. All CXPI nodes transition into Sleep mode in case a physical bus error occurs.

6.1.5.3 Network Management (NM)

The 'Frame Information byte' in the CXPI frame includes two (2) 'NM' bits. One bit defines the 'Wakeup.ind' wakeup indication and the other bit defines the 'Sleep.ind' sleep indication. The 'NM' bits are used to control the wakeup and sleep function of the slave nodes in order to save electrical power consumption.

a. Wakeup.ind:

b. At the time of wakeup the master node transmits the bus master clock as a trigger and sets the 'Wakeup.ind' bit in the Frame Information byte to '1' until it sleeps. If the slave node transmits the wakeup pulse by own node at the time of wakeup, it keeps setting '1' until it sleeps. Otherwise it is set to '0'.

c. Sleep.ind:

d. Each node shows whether it is state where it may sleep now. In the state of the sleep prohibition, '0' is set. In the state of the sleep permission, it is set to '1'.

6.1.6 Multi Clock Master Processing

The CXPI protocol features a multi clock master processing to assure reliable communication after a transition from Sleep mode to Standby state and then Operational state. The CXPI multi clock master system consists of a primary and secondary clock master.

In a failure free CXPI communication system the primary clock master starts transmitting a Periodic Request ID (PID or PTYPE field) after a node, which has identified an internal event, transmits a wakeup pulse to wake up the master node. After a specified wakeup recovery time, the master node with the bus master clock function is expected to transmit a Periodic Request ID (PID field or PTYPE) followed by a response frame.

If a node has been assigned the secondary clock master function on the CXPI bus does not detect a Periodic Request ID (PID field) after the wakeup timeout the node assigned the secondary clock master function will automatically transmit the bus master clock and the Periodic Request ID (PID field). The primary clock master and the secondary clock master are pre-assigned during system design.

The primary clock master takes over the bus master clock function from the secondary clock master at every wakeup (transition to operational mode).

During a wakeup period, the primary clock master should send the bus master clock.

The detailed description of multi clock master processing is specified in the JASO D015-3, Automobiles — Clock Extension Peripheral Interface (CXPI) — Part 3: Protocol Specification.

6.2 Data Link Layer

6.2.1 Data Link Layer functional model

6.2.1.1 Overview

The Data Link Layer functional model consists of three (3) major function blocks:

- Transmission Processing Unit
- Reception Processing Unit
- Retransmission Management Unit

Figure 16 shows the Data Link Layer functional model.

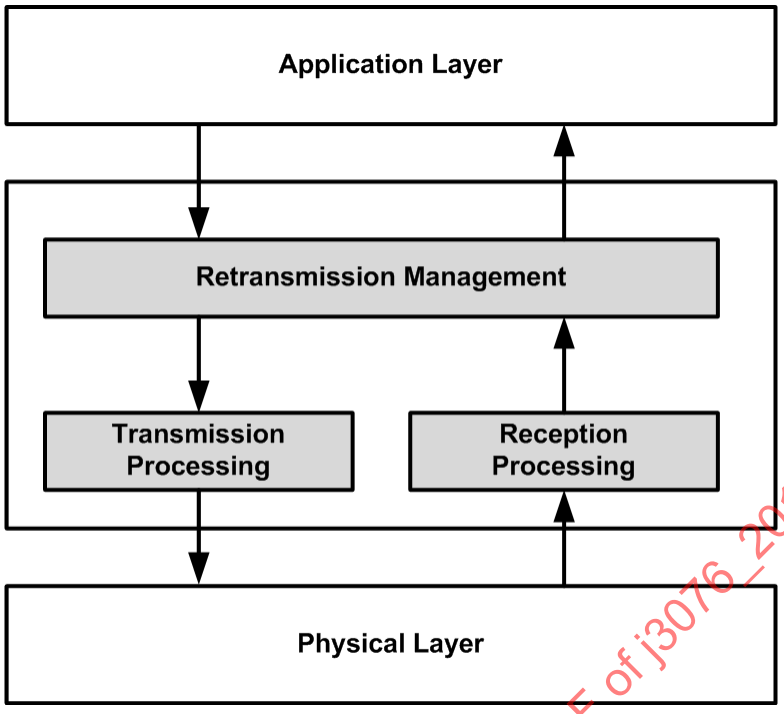


Figure 16 - Data Link Layer functional model

6.2.1.2 Transmission Processing Unit

The Transmission Processing Unit performs the following tasks:

a. Frame structure,

b. Add start and stop bit to each byte before transmission onto the physical layer,

c. Detect complete received frame,

d. Bus access method (according to Event Trigger Method or Polling Method),

e. Error detection function,

f. Optional bus monitoring module (does not send any data to the CXPI bus),

g. Failure management to prohibit sending data if error counter has reached maximum value.

6.2.1.3 Reception Processing Unit

The Reception Processing Unit performs the following tasks:

a. Frame structure,

b. Add start and stop bit to each byte before transmission onto the physical layer,

c. Reception function to determine whether received frame includes a matching PID field value for this node,

d. Clock existence function to determine whether bus master clock signal is on the CXPI bus,

e. Error detection function,

f. Failure management to prohibit sending data if error counter has reached maximum value.

6.2.1.4 Retransmission Management

The Retransmission Management compares each bit of the received data (Reception Processing Unit) with the transmitted data (Transmission Processing Unit). If a bit does not match due to arbitration or error, the frame can be retransmitted after the next IFS.

6.2.2 Frame Types

6.2.2.1 Overview

Three (3) CXPI frame types are defined:

- Normal frame,
- Sleep frame,
- Long frame (optional).

6.2.2.2 Frame Structure

The frame types as described in chapter 6.2.2 are applicable.

A frame consists of the following fields:

- PTYPE field: only used by the master node in Method 2 – Polling Method.
- PID field: mandatory in all master and slave node frames to identify the content of the data bytes.
- Frame Information (FI) field: mandatory in all master and slave node frames to identify
  - the Data Length Code (DLC) of the data byte field,
  - the Network Management (NM) indication field (Wakeup.ind, Sleep.ind),
  - the Counter (CT) field (optional);
- Data field: all master and slave node frames may contain actual information;
- Cyclic Redundancy Check (CRC) field: mandatory in all master and slave node frames to detect erroneous frame content to trigger error handling.

6.2.2.3 Normal Frame

The normal frame format supports both methods, Event Trigger and Polling and is mainly used to communicate compact data either event based or scheduled by the master node.

The normal frame structure is illustrated in Figure 17.

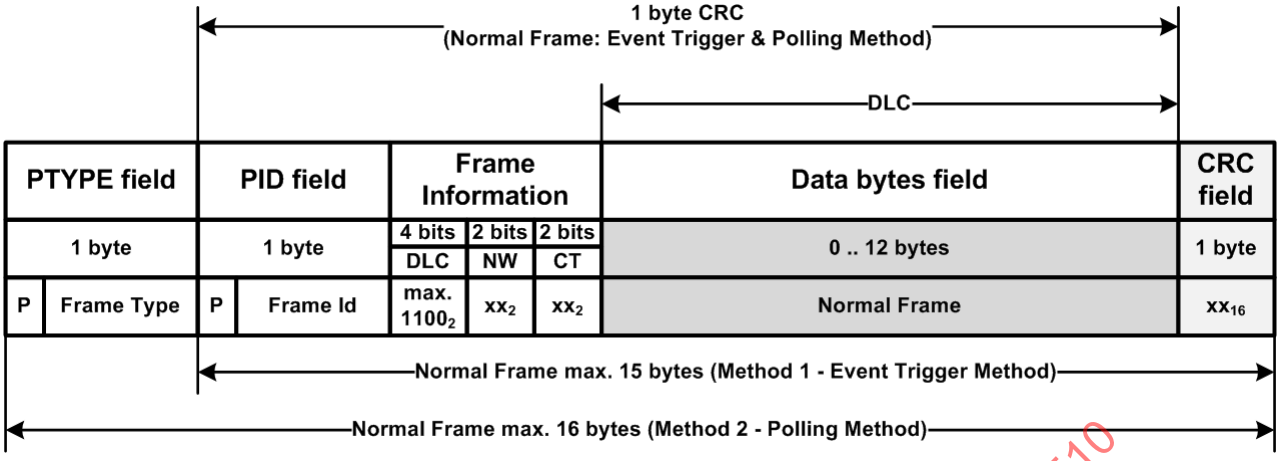


Figure 17 - CXPI normal frame structure

6.2.2.4 Sleep Frame

The sleep frame format supports both methods, Event Trigger and Polling and is only used by the master node to command each slave node to transition into sleep state. The sleep frame is of fixed length for both, Event Trigger and Polling method. The sleep frame is identified by a fixed PID field value and contains fixed data byte values.

The sleep frame structure is illustrated in Figure 18.

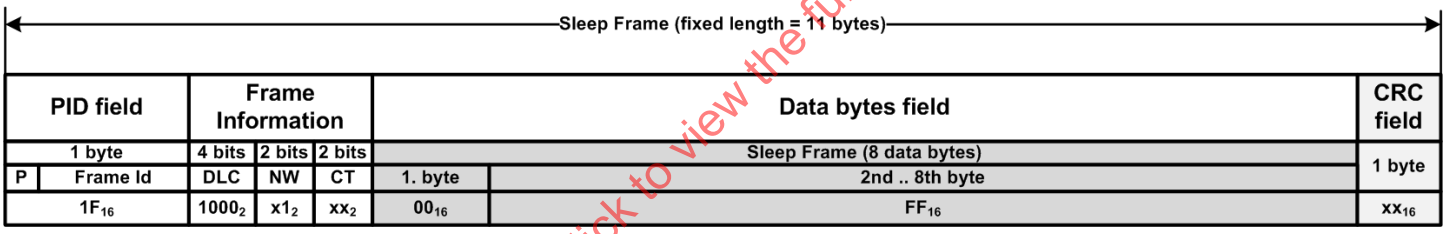


Figure 18 - CXPI sleep frame structure

6.2.2.5 Long Frame

The long frame format supports both methods, Event Trigger and Polling and is mainly used to communicate large amount of data either event based or scheduled by the master node.

The long frame structure is illustrated in Figure 19.

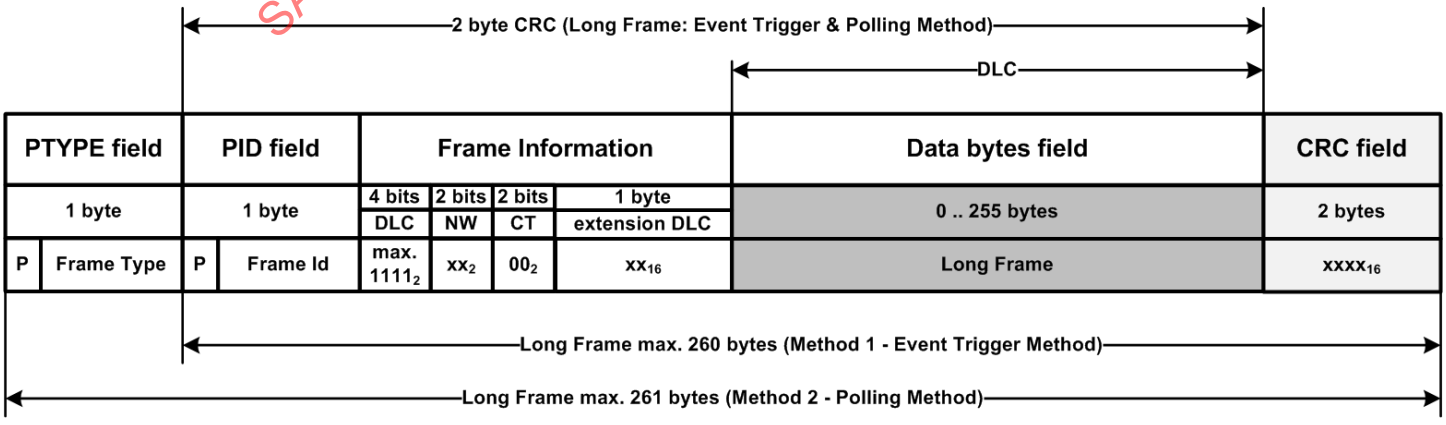


Figure 19 - CXPI long frame structure

6.2.3 Universal Asynchronous Receiver Transmitter (UART) Byte Format

A UART byte (8 bits) transmission starts with the start bit followed by the LSB (Bit 0) followed by bits 1 .. 7, where bit 7 is the MSB, which is followed by the stop bit.

Figure 20 shows the UART byte transmission.

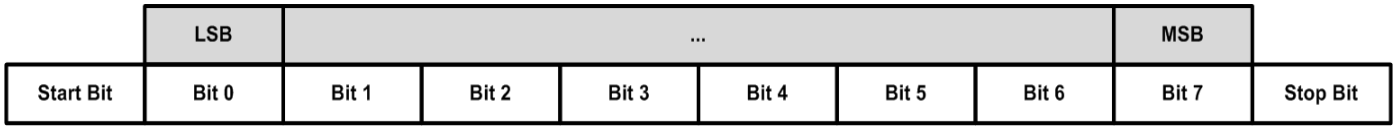


Figure 20 - UART byte transmission

6.2.4 Data Transmission Timing

6.2.4.1 Overview

The CXPI data link implements two (2) data transmission timing definitions:

- Inter-Frame-Space (IFS), and
- Inter-Byte-Space (IBS).

6.2.4.2 Inter-Frame-Space (IFS)

The purpose of the IFS between frames is to force the CXPI bus signal to a logic '1' state for a minimum amount of time which is longer than the transmission time of two (2) bytes with start and stop bits. After the IFS expired the CXPI bus signal stays in idle state (logic '1') until a CXPI node starts transmitting the start bit of the first byte of a frame.

Figure 21 shows the Inter-Frame-Space between frame and bus idle state.

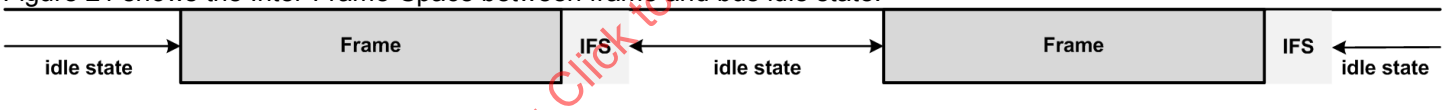


Figure 21 - Inter-Frame-Space between frame and bus idle state

The IFS timing definition is specified in the JASO D015-03 CXPI Protocol Specification.

6.2.4.3 Inter-Byte-Space (IBS)

The purpose of the IBS between bytes within a frame is to allow CXPI devices without a UART to transmit the bytes of a frame with a short idle period (less than the transmission time of a byte with start and stop bit) between bytes.

Figure 22 shows the Inter-Byte-Space between bytes within a frame.

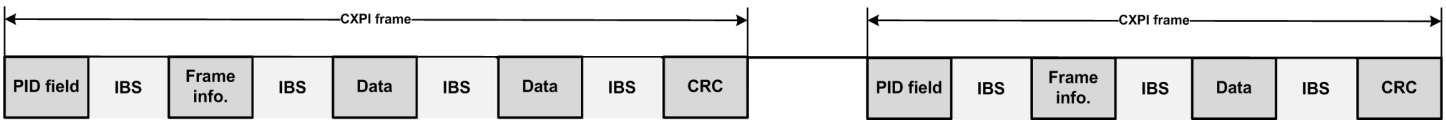


Figure 22 - Inter-Byte-Space between bytes within a frame

The IBS timing definition is specified in the JASO D015-03 CXPI Protocol Specification.

6.2.5 CXPI Bus Access Determination

6.2.5.1 Method 1 – Event Trigger Method Bus Access

If the Event Trigger Method is implemented in all CXPI nodes then the master and slave nodes determine bus access based on the expiration of an IFS followed by an idle state.

6.2.5.2 Method 2 – Polling Method Bus Access

If the Polling Method is implemented in all CXPI nodes then the master is the only node to access the bus. The master node waits until the expiration of the IFS followed by an idle state or alternatively the master node is allowed not to wait until the expiration of the IFS if the master node assures via the master node schedule that the response of the slave node does not overlap with the following periodical transmission of the master node.

6.2.6 Error Detection

Multiple types of error detection are supported by the CXPI Data Link Layer.

a. Bit error:

   This type of error is detected at the time of the transmission by comparing the received bit with the transmitted bit. If the logic state of both bits is not the same a bit error has occurred.

b. CRC error:

   The transmitting node stores the CRC calculation result in the CRC field of the frame to be transmitted and then performs the transmission.

   The receiving node performs a CRC calculation on the received frame and compares this value with the value contained in the CRC field of the frame. If the values are different a CRC error is indicated and the frame is disregarded.

c. Parity error:

   Method 1 – Event Trigger Method: The transmitting node sets the Parity bit of the PID field to either '1' or '0' so that the sum of all bits of the PID field equals '1' (odd parity). The receiving node calculates the bits 6 through 0 of the PID field and then adds the Parity bit value of the PID field. If the result is not equal to '1' (odd parity) a framing error occurred during the transmission of the frame.

   Method 2 – Polling Method: The transmitting node sets the Parity bit of the PTYPE and PID field to either '1' or '0' so that the sum of all bits of the PTYPE and PID field equals '1' (odd parity). The receiving node calculates the bits 6 through 0 of the PTYPE field and PID field and then adds the Parity bit value of the PTYPE field and the PID field. If the result is not equal to '1' (odd parity) a framing error occurred during the transmission of the frame

d. Physical bus error

   Support for Wakeup/Sleep: Each node on the bus detects physical bus errors if a frame is not received within a specific time if in standby or normal mode of operation.

   No support for Wakeup/Sleep: The difference to "Support for Wakeup/Sleep" is the error handling. The device which does not support the Wakeup/Sleep function only notifies the physical bus error to the application when the device detects the physical error.

e. Data length error

Normal frame: The receiving node reads the DLC (Data Length Code) in the Frame Information field and compares this value with the data byte field contained in the received frame. If the value contained in the data byte field is different from the value of the DLC a data length error occurred.

Long frame: The receiving node reads the extension DLC (Data Length Code) in the Frame Information field and compares this value with the data byte field contained in the received frame. If the value contained in the data byte field is different from the value of the extension DLC a data length error occurred.

f. Counter error (optional)

The receiving node reads the CT (Counter) in the Frame Information field of the frame (initial value of first frame = '00_2', value of CT of 2nd frame = '01_2', value of CT of 3rd frame = '10_2', value of CT of 4th frame = '11_2', value of CT of 5th frame = '00_2', etc. The receiving node detects a CT error in case a non-continuous CT value has been read from the frame information field. The transmitting node increments the CT value by '1' every time it sends a frame.

g. Overrun error

Frame in the UART receive buffer is overwritten by another frame received before the previous frame has been read.

h. Framing error

Each byte transmission has a stop bit at the end. The stop bit of a byte received must always have a logic '1' state. A logic '0' state is qualified as framing error.

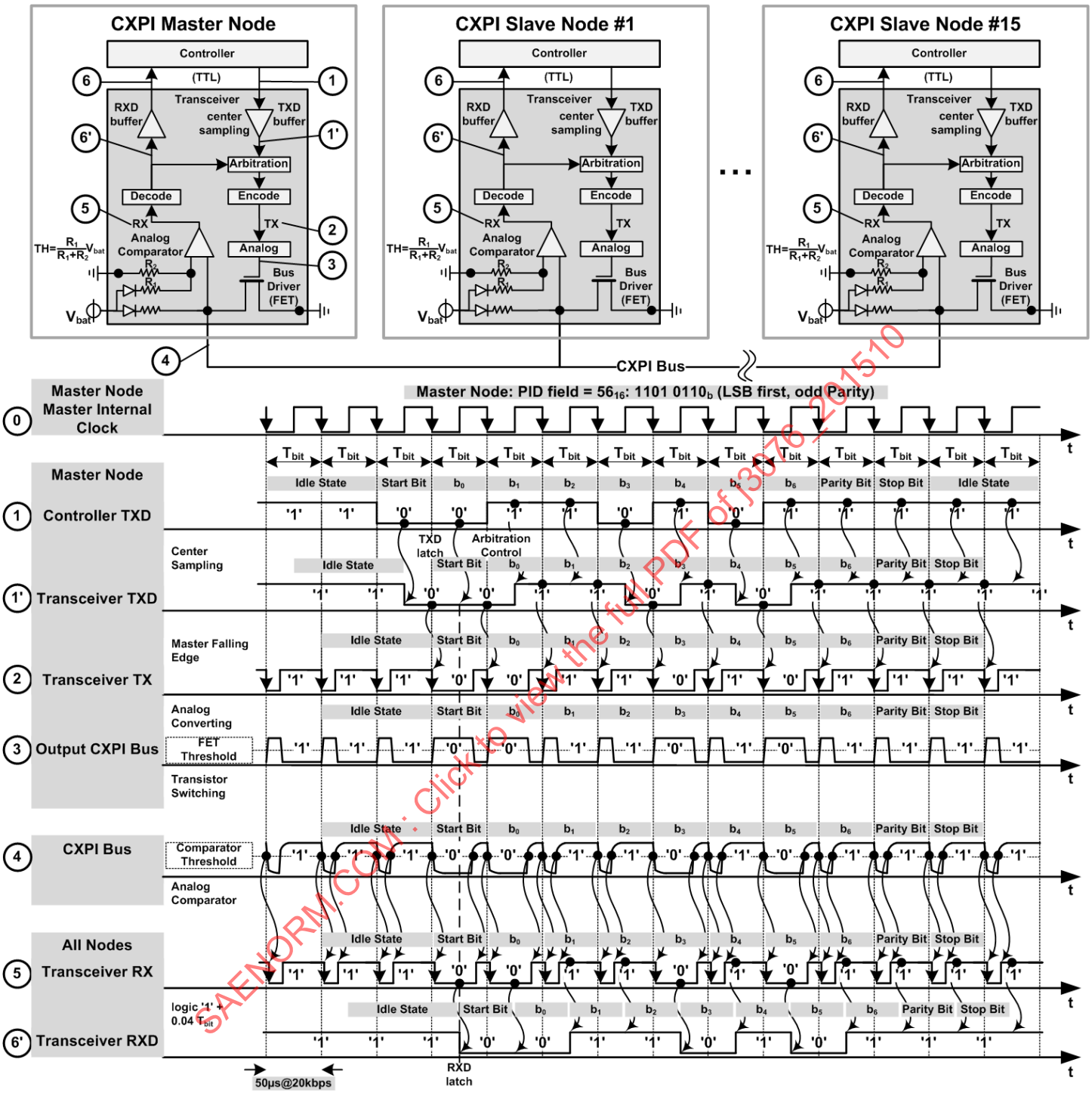
6.2.7 Error Handling

The error handling function prevents the CXPI bus from sending false data back to back when there is failure in a node. Each node has an error counter. If a transmitting node detects an error when sending a frame, the value of the error counter is added. If the error counter exceeds the limit as specified in JASO D015-03, the transmitting node prohibits any transmission until it has met the recovery condition.

6.2.8 CXPI Communication System with CSMA/CR

The CXPI Transceiver supports the Carrier Sense Multiple Access (CSMA) with Collision Resolution (CR) Bit Arbitration functionality. The node transmitting a logic '0' bit state wins arbitration against nodes transmitting logic '1' bit state at the same time. The node which wins arbitration continues the transmission and the nodes loosing arbitration stop sending/disconnect from the bus to not further disturb the CXPI bus signal transmission.

Figure 23 shows a timing diagram of the CXPI communication system.



No	
0	CXPI Master Node Clock
1	CXPI Master Node Controller TXD line which transmits bytes from the UART in TTL level
1'	CXPI Master Node Transceiver TXD line which shows ½ x T_{bit} delay because of center sampling at (1)
2	Master Node Encoder logic which performs conversion of NRZ signal from TXD buffer to PWM signal
3	Master Node Transceiver analog circuit shapes the TTL into an analog signal
4	CXPI communication bus (analog signal waveform is interpreted by all connected nodes)
5	Actual CXPI communication bus analog RX signal waveform received by all nodes to be decoded
6	Controller RXD line which receives bytes from the RXD buffer in TTL level
6'	Controller RXD line which receives bytes from the Decoder logic

Figure 23 - Simplified CXPI communication system