SAE Aerospace
An SAE International Group

# AEROSPACE STANDARD

AS6003

Issued Proposed Draft 2010-12-21

# TTP Communication Protocol

## RATIONALE

TTP is used in a variety of aerospace applications (for example, Boeing 787 power generation systems and environmental controls, cabin pressure systems for Airbus A380, Aermacchi M-346 FADEC…) and continues to attract significant cross-industry attention for commercial and defense applications.

The SAE standardization of TTP:

- acts as an integration risk reduction mechanism
- ensures compatible physical implementations
- enables common test/maintenance equipment
- leverages industry investments
- ensures openness and enables multiple component and tool suppliers

and therefore reduces the overall cost and risk of applying this technology.

The SAE standardization based on TTP specification protects long-term system design investments, enables development of the COTS ecosystem, and minimizes sourcing risks for OEMs, integrators and system suppliers.

## INTRODUCTION

The Time-Triggered Protocol (TTP) is a real-time communication protocol for the interconnection of electronic modules of distributed fault-tolerant real-time systems. TTP is a core technology for fault-tolerant distributed embedded computing and enables design of deterministic embedded computing platforms for critical systems. As such, it contains communication protocol capability for data exchange among nodes, but also provides higher level services for design of reusable generic platforms with robust partitioning among functions. This document specifies the structure of the TTP protocol, services and mechanisms on an abstract level without implementation-specific details.

TTP's features support the design of systems with a high degree of dependability, safety, availability, reliability, maintainability, and reduced system complexity.

Some methods contained in the document are protected by granted and pending patents, and underlie SAE standard IP policy.

TABLE OF CONTENTS

# 1. SCOPE

This SAE Aerospace Standard (AS) establishes the specification for TTP communication protocol functionality as a core networking component for design of synchronous, time-triggered distributed real-time communication networks.

This document is referred to as the “base” specification, containing the generic specification of TTP communication protocol functionality, and frame formats, communication services and protocol state machines with transitions, TTP node operation, TTP network operation, distributed clock synchronization and error detection on the network layer.

The details described in this standard enable interoperable TTP controller implementations.

## 1.1 Purpose

The purpose of this document is to standardize mechanisms and operation of the TTP protocol for safety-critical and mission-critical applications. The information herein will be used to assist in the design, fabrication, system integration, and obsolescence management of TTP-based systems.

This document is controlled and maintained by the SAE AS-2 committee with technical support from TTA-Group members and TTP users.

## 1.2 Application

TTP is well-suited for the design of deterministic system architectures with hard real-time behavior in safety and mission-critical applications. The application of TTP as a fieldbus is viable for fault-tolerant distributed systems, deterministic networks, and distributed control system platforms (e.g., flight controls, by-wire steering, environmental controls, smart sensor/actuator networks, distributed power generation, or landing gear). TTP supports design of time-triggered architectures and reusable generic platforms compliant with RTCA DO-297. TTP can also be used for backplane communication and design of modular aerospace system controls.

## 1.3 Interpretation

The following interpretations shall be placed upon these words, unless stated otherwise, where they are used in this document.

*May*: An allowed action.

*Shall*: A mandatory requirement.

*Should*: A recommended action.

*Will*: A declaration of intent.

# 2. REFERENCES

The following publications form a part of this document to the extent specified herein. The latest issue of SAE publications shall apply. The applicable issue of other publications shall be the issue in effect on the date of the purchase order. In the event of conflict between the text of this document and references cited herein, the text of this document takes precedence. Nothing in this document, however, supersedes applicable laws and regulations unless a specific exemption has been obtained.

## 2.1 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.

940140 Fault Management in the Time Triggered Protocol (TTP)

## 2.2 Other Publications

[Kop87] H. Kopetz and W. Ochsenreiter. Clock Synchronization in Distributed Real-Time Systems. *IEEE Transactions on Computers*, 36(8):933–940, Aug. 1987.

[Pal97] R. Pallierer and T. M. Galla. Multiplexed SRUs in TTP – Concepts and Approaches. Research Report 19/97, Institut f¨ur Technische Informatik, Technische UniversitätWien, Vienna, Austria, October 1997. Confidential.

[Pfe99] Holger Pfeifer, Detlef Schwier, and Friedrich W. Von Henke. Formal Verification for Time-Triggered Clock Synchronization. In Charles B. Weinstock and John Rushby (eds.), Editors, *Dependable Computing for Critical Applications 7*, volume 12 of *Dependable Computing and Fault-Tolerant Systems*, pages 207–226. IEEE Computer Society, January 1999.

[Pfe00] Holger Pfeifer. Formal Verification of the TTP Group Membership Algorithm. In Tommaso Bolognesi and Diego Latella, Editors, Formal Methods for Distributed System Development Proceedings of FORTE XIII / PSTV XX 2000, pages 3-18, Pisa, Italy, October 2000. Kluwer Academic Publishers.

## 2.3 Structure of the Document

This specification document is self-contained and gives a bottom-up approach to the functionality of the TTP protocol.

It is divided into the following chapters:

- Chapter 3 provides an overview of the TTP network architecture.
- Chapter 4 defines the minimum characteristics and requirements for the host interface.
- Chapter 5 describes and defines the requirements on Data Link Layer level for the frame formats and the bus access scheme.
- Chapter 6 defines the minimum characteristics and requirements for the Physical Layer.
- Chapter 7 describes the timing and synchronization algorithm used by TTP.
- Chapter 8 describes and defines the mechanisms for the protocol service layer.
- Chapter 9 defines the operation of the TTP controller in the context of the TTP state machine model.

## 2.4 Typographic Conventions

The typographic conventions for this documentation are as follows:

| Element | Typographic format |
|---|---|
| Protocol variables / text emphasis | *lower case italic* |
| Protocol states | *Upper case italic* |
| Header within a section | **bold** |

3. OVERVIEW OF A TTP-BASED SYSTEM

The TTP controller is the core component of the time-triggered architecture, the conceptual time-triggered-based system design approach. This section describes general aspects of TTP-based network design, as well as system design aspects of an electronic module or *node*.

3.1 Structure of a TTP Network

A TTP-based communication network (TTP network) consists of a set of electronic modules that are typically connected by dual channels – *channel 0 and channel 1* – which comprise a *TTP-bus*. A TTP network – as shown in Figure 1 – is called a *cluster*. The basic building block of a cluster is the electronic module (node). Depending on the system architecture and depending on the system requirements, e.g. with respect to fault tolerance, communication between nodes can be established on a single channel (channel 0 or channel 1) or redundant channels (channel 0 and channel 1) basis.

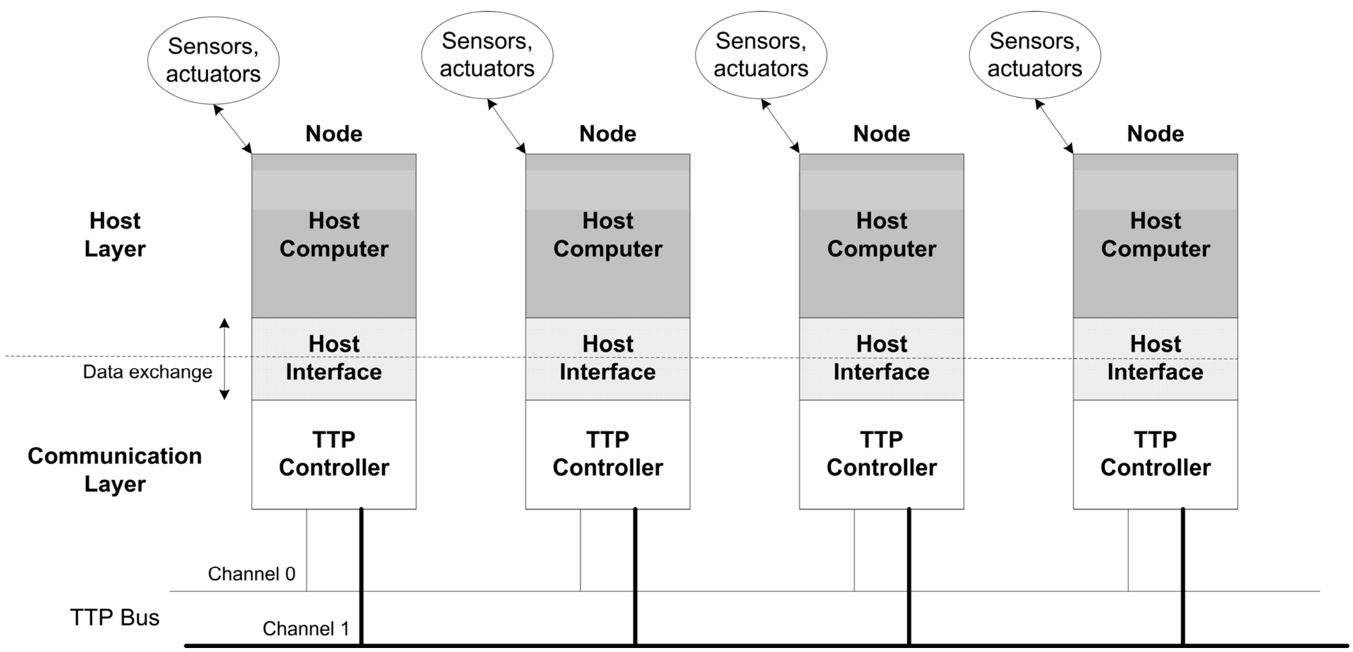


FIGURE 1 - TYPICAL TTP NETWORK

3.2 Structure of an Electronic Module (Node)

An electronic module has the general structure shown in Figure 2. A node comprises of a host (typically a CPU executing an operating system which manages the application software, with memory and the access to the I/O subsystem, or an FPGA) and a TTP controller.

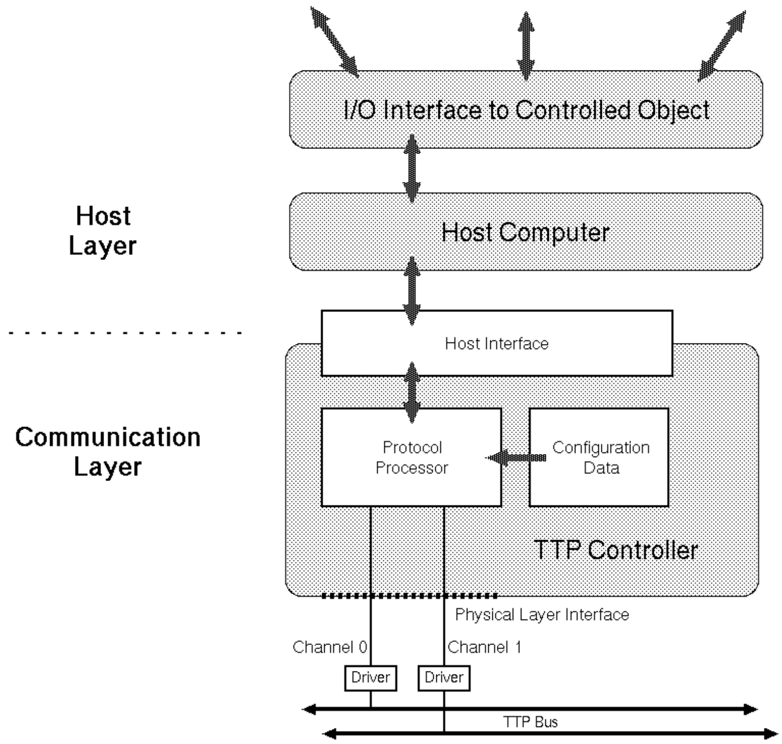


FIGURE 2 - STRUCTURE OF A TTP NODE

A TTP controller shall process the execution of the protocol services based on a statically defined set of communication requirements (shown as configuration data in Figure 2). Furthermore the Protocol Processor shall use a well-defined interface to the host and to the TTP bus to provide reliable operation of the services defined by the TTP (Protocol services).

From an abstract perspective, each node consists of two layers – the *communication layer*, which shall ensure reliable time-triggered communication, and the *host layer*, which provides/consumes application data to/from the communication layer. The host and the communication layer shall run as *autonomous subsystems* on a node, but depending on the chosen system design approach, both subsystems shall be able to run in a synchronous or asynchronous manner to each other.

3.2.1 Communication Layer

The time-triggered protocol defines all the processes in the communication layer. The central characteristics of the time-triggered protocol are the interfaces of the communication layer, which are described in the next section.

3.2.1.1 Central Characteristics of the Time-Triggered Protocol

3.2.1.1.1 Protocol Services

The protocol services are needed to reliably perform and to control the exchange of data between the nodes in a cluster. The protocol services can be divided into the following main categories:

- *Communication services* shall guarantee reliable data transmission, the *startup* of a cluster, the *reintegration* of nodes, the *acknowledgment*, fault-tolerant *clock synchronization* and handling of cluster *mode changes*.

- *Safety services* shall support error detection including *node membership*, the *clique detection* algorithm and *Host/Controller life-sign* service.

The protocol services are defined in Chapter 8.

#### 3.2.1.1.2 Autonomous Operation of the TTP Controller

The TTP controller shall function as an autonomous subsystem. That means that the operations that the TTP controller is able to perform at any given time are defined by the state machine model in chapter 9. But the TTP controller requires a running host for operation and some initialization data and control data as required to perform the individual TTP services (Chapter 8).

#### 3.2.1.1.3 Global Time Base

A TTP Controller shall provide a time base to trigger the protocol events (the execution of the TTP services, frame reception and frame transmission). This time base shall have a granularity in macrotick units (see section 7.1.1). The global time base shall ensure synchronous protocol execution and cluster operation between all nodes in the cluster within a configurable time interval – the so-called *precision*. Synchronous operation ensures that all communication layer-related operations on all nodes happen in a coordinated manner, to enable each node to have bus access without collision, and allow reliable data exchange with predictable latency and minimal jitter. Synchronized operation is achieved (between a minimum of two nodes) immediately after a successful startup – as mentioned in section 8.1.

#### 3.2.1.1.4 TDMA Bus Access

When synchronous cluster operation mode is accomplished, meaning that communication was initiated and synchronization could be achieved between at least two active nodes (nodes that are allowed to transmit data on the Bus), a TDMA-based strategy is used by all nodes in the cluster to access the bus. This means that every active node has a certain amount of reserved bandwidth – a *node slot* – to write data to the bus. Every TTP controller shall receive all data transmissions available on the bus.

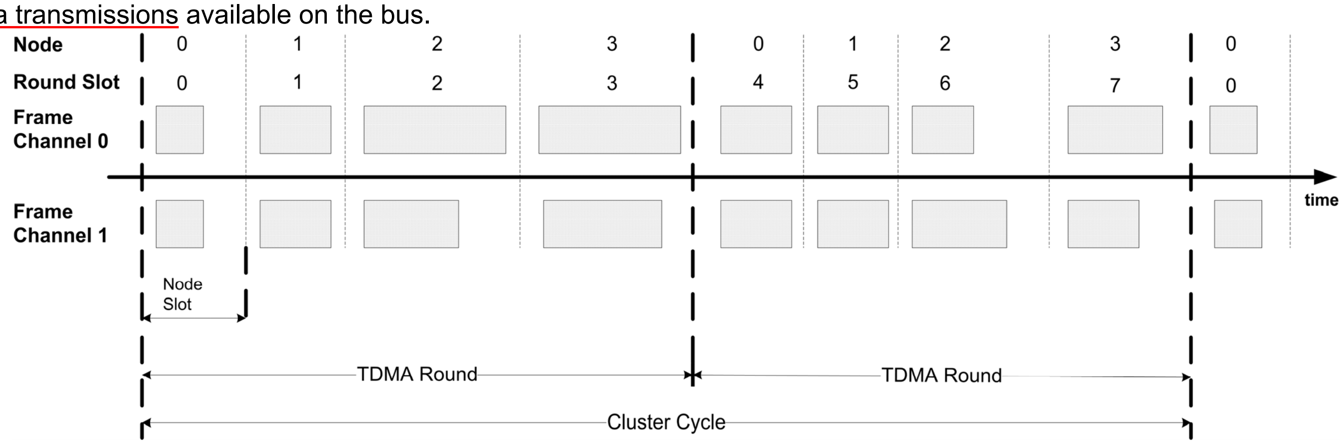


FIGURE 3 - MEDIA ACCESS SCHEME

Each active node shall periodically utilize the full transmission capacity of the bus in its dedicated node slot, by sending *TTP frames* on both channels. The frames on channel 0 and channel 1 may differ in their frame type (as defined in section 5.4), data length and data content.

The periodic sequence of node slots is called a *TDMA round*. The duration of each slot as well as the sending sequence of the different nodes in the cluster shall be equal and fixed in all TDMA rounds and for all cluster modes. The only possible variation between two TDMA rounds is the amount of application data transmitted in the slots (the frame length) and in case of multiplexed slots the assigned sending node in a particular round. The pattern of periodically recurring TDMA rounds is called a *cluster cycle*.

3.2.1.1.5 Configuration Data

A TTP controller shall perform according to application-specific communication and timing requirements which could be defined during a cluster design process. This set of information is referred to as configuration data and should be the output of the cluster design. The configuration data shall define what a TTP controller shall do at each point in time during a cluster cycle. Each node has its own unique slot position in the TDMA round; therefore each node shall be configured with an individual set of communication schedules, respective to its positions in the TDMA round. Some configuration data applies to all nodes in the cluster (e.g., bus speed, TDMA round duration, number of rounds). There is also node-specific configuration data (e.g., the time instant when a frame arrives / is to be sent). A TTP controller shall process the definitions in the configuration data based on the progress of a global time base. Depending on the TTP controller implementation, this set of configuration data may be realized, e.g., directly by the chip design, or made available as a data array in the memory. The set of configuration data is summarized in Table B3.

3.2.1.2 Overview of the Host Interface

The host interface is needed to enable data exchange between the host layer and the communication layer either in synchronous or asynchronous operation mode as described in section 3.2.2. The host interface is required to ensure interoperability – simultaneous random access – between the host and the TTP controller for frame, status and control data. The host interface specification in Chapter 4 specifies the minimum requirements

3.2.1.3 Overview of the Physical Layer Interface

The physical layer interface is needed to guarantee bus-compatibility – interoperability on the network level – between different TTP controllers. Bus-compatibility means that two different TTP controller types – fulfilling the same bus-compatibility specification – may exchange data and are fully compatible in all aspects of TTP communication given that they also share the same physical layer. This physical layer interface is characterized by the connection between a TTP controller and the line driver as described in Chapter 6, as well as on data link level with respect to frame format and bus access scheme as described in Chapter 5. SAE standardization projects AS6003/1, AS6003/2 and other future physical layer standards cover detailed description of physical layers for TTP communication protocol.

3.2.2 Application Layer - Time-Triggered Architecture

The host layer as depicted in Figure 2 is responsible for the execution of local or distributed application algorithms, and handles the interface to its environment (Sensors, Actuators) and to the TTP communication layer. The host layer may perform in a *synchronous* or *asynchronous* manner to the network-wide synchronized communication layer.

3.2.2.1 Application Layer Synchronous to Communication Layer

From the application perspective, if the host subsystem (the algorithms performed on the application layer) is aligned (synchronized) to the global time, data transmission between the host and the TTP controller always occurs at the right time and without data access conflicts, as defined at design time. This approach – *TTA approach* (time-triggered architecture approach) – helps to reduce system complexity, enhances resource use and improves host subsystem abstraction and independence from underlying networking and hardware details.

The TTA approach takes advantage of the availability of the global time to precisely and unambiguously specify the interfaces among the nodes, to simplify communication, to establish system state consistency, to perform continuous and prompt error detection, and to enable seamless system integration for distributed real-time applications.

3.2.2.2 Application Layer Asynchronous to Communication Layer

From the application perspective communication follows a deterministic schedule, but applications can receive data with some latency and delay which is influenced by application startup activity and its alignment with the communication schedule. The application operates in a “local” or event-driven fashion while the communication is strictly deterministic and not influenced by the progress of the application. This means that coordination of data transfers (e.g., prevention of host interface access conflicts) in time should be done by the application.

# 4. HOST INTERFACE

The host interface shall guarantee collision-free data exchange between the host and the TTP controller. The TTP controller shall make available received application data and continuously updated TTP controller and network related status data. The host shall make available updated message, control, and configuration data to the TTP controller.

Three different categories of data shall be exchanged between the host and the TTP controller:

a. **Message data** – The host interface shall provide means for the exchange of application data produced by the host (application data for transmission on the TTP bus) or received by the TTP controller (application data acquired from the TTP bus). Furthermore the host interface shall provide means to pass the evaluated frame status after each TTP frame reception or transmission (as described in section 5.4) to the host. The host should deal with application data, only if the status of the received frame is valid.

b. **Status data** – The TTP controller shall be able to inform the host about the status of the protocol execution by maintaining the status data continuously.

c. **Control data** – The host shall be able to provide setup information or control and maintenance data to the TTP controller. This is needed to enable protocol execution, valid frame transmission and to initiate operation mode changes.

Table 1 summarizes the different kind of data needed for the execution of the communication and protocol related operations.

TABLE 1 - HOST INTERFACE SPECIFICATION

| Host interface – data | Category | Reference |
|---|---|---|
| Acknowledgment error | Status | Section 8.3 |
| Clique error | Status | Section 8.4 |
| System blackout error | Status | Section 8.4 |
| Mode violation error | Status | Section 8.6 |
| Initialization error | Status | Section 9.2.3 |
| TTP controller life-sign | Status | Section 8.5 |
| Host life-sign | Control | Section 8.5 |
| Time Startup Field | Control | Section 9.2.5 |
| Protocol version | Status | Section 9.2.3 |
| Mode change request | Control | Section 8.6 |
| Protocol state | Status | Section 9 |
| TTP Controller On | Control / Status | Section 9.2.2 |
| Time Startup Field | Control | Section 9.2.5 |
| Valid C-state | Status | Section 9.2.5 |

## 5. DATA LINK LAYER

The data link layer deals with aspects of the TDMA-based bus access scheme, frame format and the calculation of the frame status. Sharing the same data link layer is mandatory for the compatibility of TTP controllers interacting (transmitting/receiving frames) within the same TTP cluster.

### 5.1 TDMA Scheme

The principle of the bus access strategy in synchronous operation mode – the TDMA scheme – is described in section 3.2.1. Thus, as long as each node uses only its own statically assigned node slot, collision-free access to the bus can be ensured. This can be guaranteed by the existence of a global time base and the static bus access schedule (given by the configuration data) that is known by all nodes. The *round slot* describes a logical slot in the cluster cycle – as illustrated in Figure 3.

A slot shall begin with *the pre-send phase* (PSP) and end at the beginning of the PSP of the next node slot. Figure 4 shows the partitions of a slot. The starting time of the *transmission phase* (TP), meaning the point in time when the frames are planned to be sent or received, is called action time (AT). The action time shall be perceived on all synchronized nodes of the cluster as the same point in time (within the precision interval as defined in section 7.1.3). The interval between two successive TPs is called *inter-frame gap* (IFG), consisting of the pre-send phase *post-receive phase* (PRP) and the *idle time*.

The PSP is required to protocol services for preparing the TTP controller to send or receive data (as described in section 9.2.6.1). During the PRP the TTP controller shall process the frame data and shall execute protocol services based on received data (as described in section 9.2.6.3). Since the slot length not only depends on the amount of data sent on the bus and the bus speed, but also on the definition of round length, the number of defined nodes (slots), usually there is a certain amount of time left which shall be called *idle phase*. During the idle phase the protocol operation shall be halted and shall wait for the start of the PSP of the next slot. The number of slots in a cluster cycle, the duration of the node slot, the action time relative to be begin of the cluster cycle and the duration of the TP, shall be defined in the configuration data.

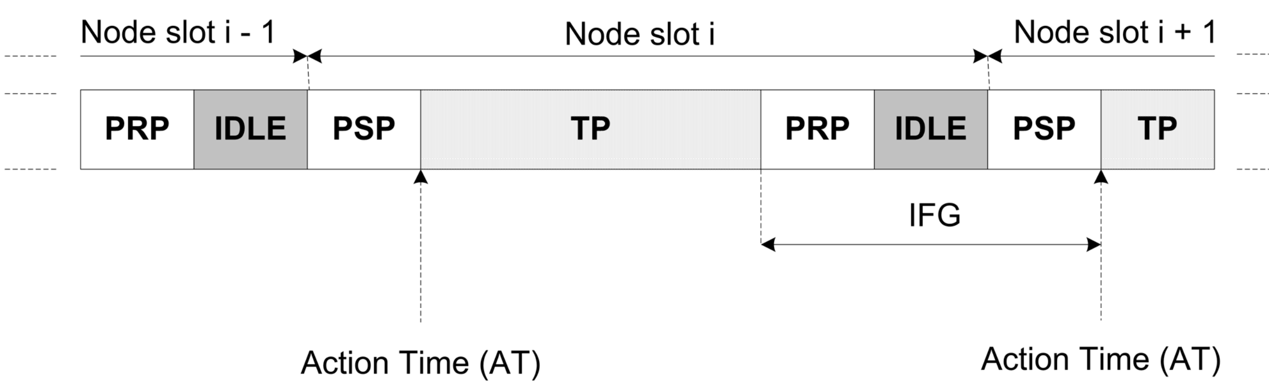


FIGURE 4 - SLOT TIMING

5.2 TTP Frame Layout

A TTP frame is transmitted in the transmission phase and is composed of protocol information and application data. Figure 5 illustrates the general frame layout.

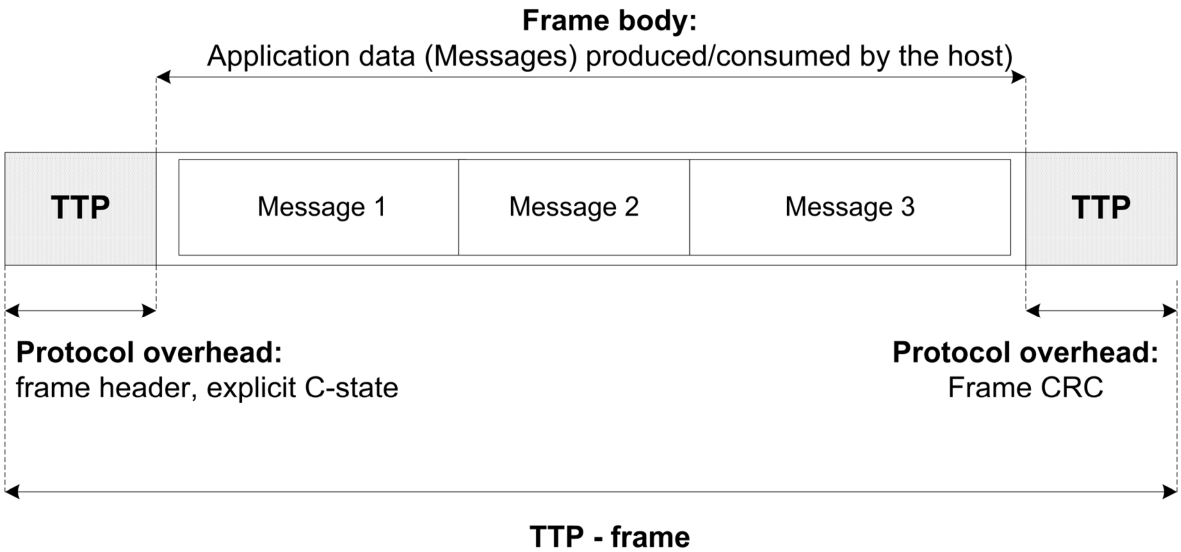


FIGURE 5 - TTP FRAME LAYOUT

**TTP Frame:** A TTP frame consists of a frame body (containing the application data) and the protocol overhead (necessary for protocol operation).

**Frame body:** The frame body is the main part of a TTP frame and carries application data produced or consumed by the host; it may be subdivided into parts that are interpreted by the host application called *messages*. Application data shall have no semantics for the TTP controller and therefore no impact on the protocol operation. The maximum size of the frame body is defined in Table B1. In case of a reception, the TTP controller stores the application data in the host interface, and in case of transmission it fetches it from the host interface.

**Protocol overhead:** The protocol overhead is necessary for the operation of the TTP protocol and shall consist of the frame header (as described in section 5.2.1), the CRC value (as described in section 5.2.3) and depending on the frame type of the explicit C-State (as described in section 5.2.2).

5.2.1 Frame Header

The header is a 4-bit field shall contain the frame type in the LSB and the mode change request field in the three MSBs.

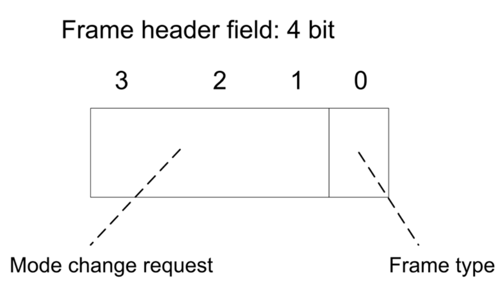


FIGURE 6 - FRAME HEADER

5.2.1.1 Frame Type

The frame type field shall contain information about how the C-state is incorporated in the frame. This information is needed for the interpretation of a received frame by a TTP controller. The following convention shall be provided for the frame type field:

- The frame type shall be set to one (bit set) to indicate the frame has an explicit C-state as described in section 5.3.

- The frame type shall be set to zero (bit reset) to indicate the frame has an implicit C-state as described in section 5.3.

5.2.1.2 Mode Change Request

The deferred mode change request field shall represent the value of a requested mode change as defined in section 8.6. The value range for this bit field is defined in Table B1.

5.2.2 C-State

**Global time field: 16 bit**

<table>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="16">Global Time in MT</td></tr>
</table>

**Cluster position field: 16 bit**

<table>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="2">DMC</td><td colspan="3">Mode</td><td colspan="11">Round slot</td></tr>
</table>

**Membership vector field: 4 x 16 bit**

<table>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="16">Membership flags 15-0</td></tr>
</table>

<table>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="16">Membership flags 31-16</td></tr>
</table>

<table>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="16">Membership flags 47-32</td></tr>
</table>

<table>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="16">Membership flags 63-48</td></tr>
</table>

FIGURE 7 - C-STATE LAYOUT

The controller state (C-state) is a collection of state variables describing the TTP controller's internal view of the cluster state with respect to the global time, cluster position and the membership status. The C-state shall have a length of 12 bytes as depicted in Figure 7. The local C-state shall be updated by each TTP controller in each round slot. The local view of the C-state shall be checked for agreement with the C-state received with each frame transmission. Therefore, the C-state represents a global view of the cluster and shall agree with the calculated C-states of the other nodes in a correctly synchronized TTP cluster. The C-state shall be exchanged by each frame transmission either explicitly or implicitly as described in section 5.2.3. A C-state disagreement between a node and the majority of the other nodes shall mark this node as faulty as described in section 8.2. Thus the exchange of the C-state between the nodes is one of the basic concepts of the protocol to support fault detection and system health monitoring. Furthermore, the C-state contains the minimum amount of data to enable integrating nodes to participate in an already ongoing cluster communication.

The local C-state shall be updated by the TTP controller in each slot in each TDMA round according to the following data definition:

- **Global time** field shall contain the macrotick value of the start time of the transmission phase (action time) in the current slot. The global time field shall be updated during the pre-send phase with respect to the current value of the global time and with respect to the current round slot position.

- **Cluster position** field shall contain information about the current round slot position, the current cluster mode and pending deferred mode change (DMC)

  - The deferred pending mode changes (DMC) field shall contain the value of requested successor cluster mode as defined in section 8.6.2.

    - The cluster mode field shall contain the value of the current active cluster mode the TTP controller is operating in. The TTP controller shall update the Cluster mode field as defined in section 8.6.3.

    - The round slot position field shall contain the current cluster position information. This field identifies the round slot-specific dataset in the configuration data. The round slot position shall be zero-based, meaning that round slot number 0 is the first round slot in the cluster cycle. The TTP controller shall update the round slot position field during the PSP.

  - **Membership vector** shall contain a consistent view of the operational status of all nodes in the cluster as described in section 8.2. A TTP controller shall update its membership vector in the post-receive phase based on its perceived status of the received frame. A sending node shall update its membership flag according to the acknowledgement algorithm (defined in section 8.2).

5.2.3 Frame CRC

The CRC check shall be done by a TTP controller to verify the correctness of any received frame. For each frame reception, a TTP controller shall compare the CRC value of the received frame with the computed CRC value over the received frame data (as described in detail in section 5.3). The CRC is used for error detection, not for error correction. CRC, C-state, and frame type are all related to each other as shown in Figure 8. The use of implicit C-States in frames improves data transmission performance, because no bandwidth is used for the C-State transmission. Explicit C-States in frames are required for the integration process and for cluster startup, because a TTP controller can only integrate on a received frame if it can extract the C-State directly. A TTP controller shall send one of these frames in its sending slot, which is defined in the configuration data. Depending on the explicit or implicit availability of the C-state in a transmitted frame, the receivers of the frame shall perform the CRC check differently as described in the following:

- **Explicit C-state:** The TTP controller shall compare its local C-state with the C-state of the received frame. This enables the TTP controller to check in case of an erroneous received frame, whether its cause is related to a transmission error or a C-state disagreement.

- **Implicit C-state:** In the case of an implicit C-state, the C-state is not part of the frame's data. The TTP controller shall use its local C-state together with received frame data for the CRC calculation. In case of an erroneous received frame, the TTP controller cannot distinguish between a transmission error and the occurrence of a C-state disagreement.

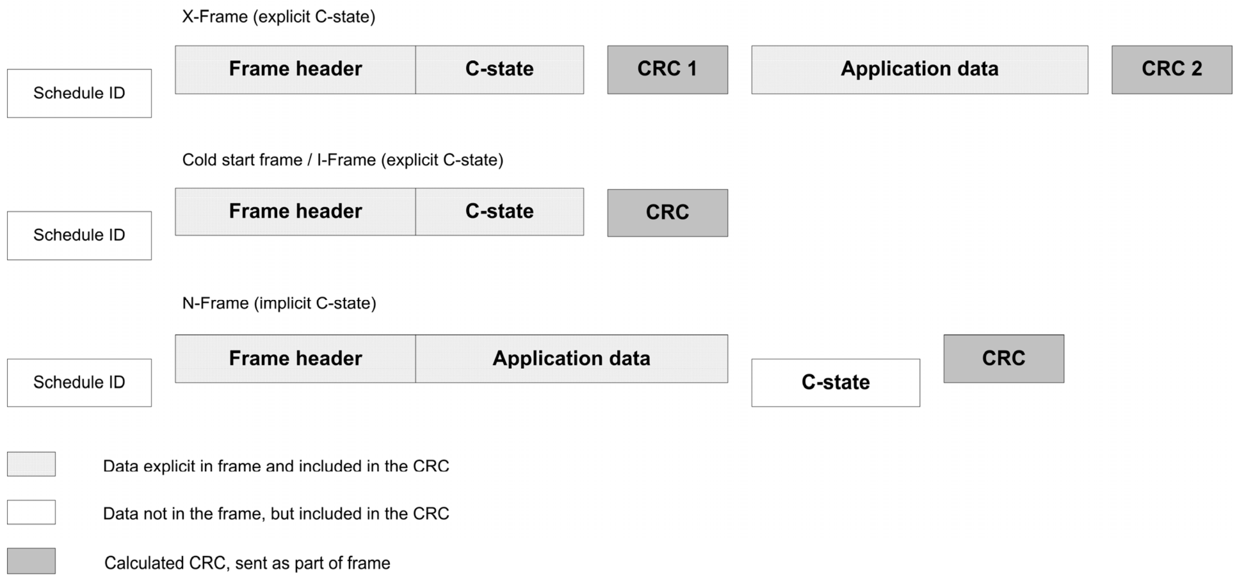


FIGURE 8 - IMPLICIT VERSUS EXPLICIT C-STATE

The CRC calculation on each node shall be initialized with the same CRC seed value, which shall be contained in the configuration data as defined in Table B1. This seed value shall relate to the specific set configuration data (cluster design) and shall be different for the CRC calculation for each channel. This will prevent the TTP controller from integrating on TTP communication based on a different cluster design which contains incompatible configuration data and crossed-over channels are detected immediately

The TTP controller shall calculate the CRC based on a generator polynomial defined in Table B1 that ensures that the defined Hamming distance is fulfilled for the maximum frame length. Table B1 specifies the parameters needed for CRC calculation which shall be fulfilled by any TTP controller.

## 5.3 Frame Types

TTP frames are coherent sequences of data, transmitted as broadcasts from a TTP controller on both channels within their slot.

TTP frames can be classified into:

- Frames with implicit C-State (N-frames)
- Frames with explicit C-State (I-frames, X-frames)
- Cold start frames

The frame type for all frames expected by a TTP controller to be sent or received shall be defined in configuration data, for each channel and for each slot.

### 5.3.1 N-Frame

The *N-frame* (*normal frame*) shall contain only application data. The C-state of the sender shall be implicitly included in the CRC value. Figure 9 specifies the frame layout.

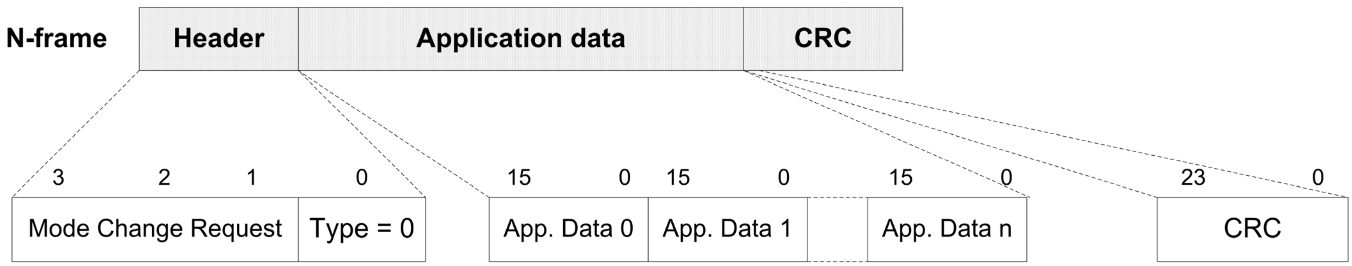


FIGURE 9 - N-FRAME

Because of the absence of an explicit C-state, nodes cannot integrate on an N-frame. N-frames have the least transmission overhead and therefore the best data efficiency.

An N-frame shall contain the following parts:

- Frame type identifier set to 0 (implicit C-state)
- Mode change request according to the mode change request field in the host interface (see section 8.6)
- Application data frame size: 1 to maximum data length (see Table B1), configurable per round slot
- CRC calculated over the frame header, the application data and the C-state

5.3.2 I-Frame

An *I-frame* (*initialization frame*) shall contain the explicit C-state of the sender, shall not contain application data, and is used for integration purposes only. Figure 10 specifies the frame layout of an I-frame.

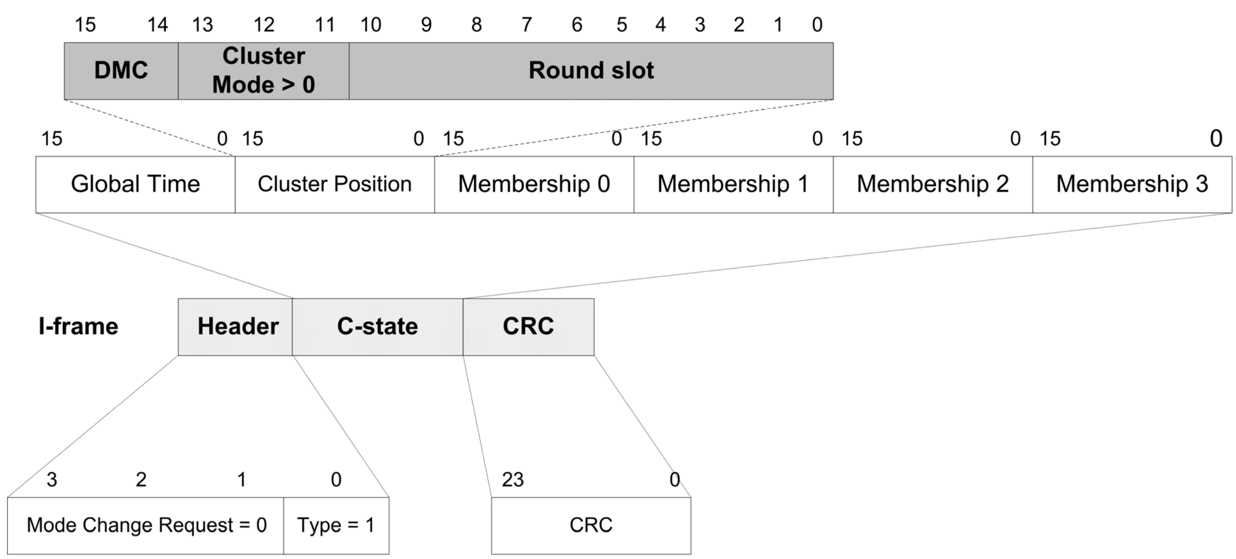


FIGURE 10 - I-FRAME

An I-frame shall contain the following parts:

- Frame type identifier set to 1 (explicit C-State)
- Mode change request according to the mode change request field in the host interface (see section 8.6)
- C-state

- CRC calculated over the frame header and the C-state.

The TTP controller shall transmit an I-frame in its sending slot whenever no application is scheduled for a round slot.

### 5.3.3 Cold Start Frame

The cold start frame is needed to initiate cluster communication. It has the format of an I-frame but with special value settings. Cold start frames shall be sent only by nodes that are allowed to initiate cluster communication. Figure 11 specifies the frame layout of a cold start frame.

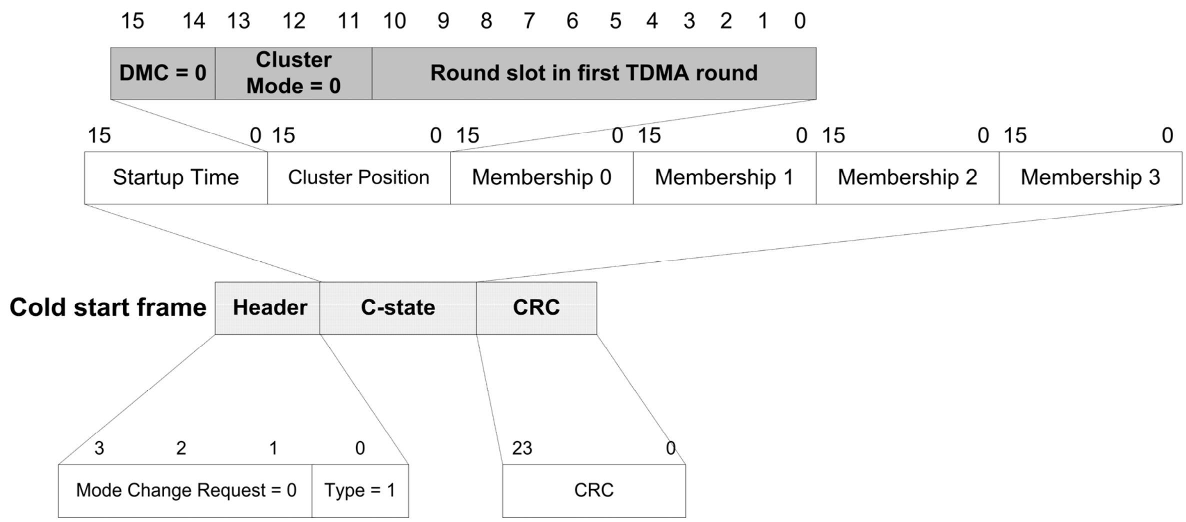


FIGURE 11 - COLD START FRAME

A cold start frame shall contain the following parts:

- Frame type identifier set to explicit C-State
- Mode change request set to ‘no request’ (see Table B1)
- The C-State’s Global Time field set to the contents of the *Time Startup* field of the host interface.
- The Round slot position field set to the round slot number of the nodes sending slot (the first TDMA round).
- The C-State’s Cluster Mode field set to the *cold start ID*. The cold start ID is a dedicated and unique cluster mode that is different from any cluster mode identifiers that may be used during schedule-synchronous operation as well as from startup mode.
- The C-State’s Deferred Pending Mode Change (DMC) field set to 0x0.
- Membership flags except that of the cold start node shall be set to 0 in the C-State’s membership vector field.
- CRC calculated over the frame header and the C-state.

### 5.3.4 X-Frame

An *X-frame* (*extended frame*) shall contain both the explicit C-state of the sender and application data.

Figure 12 specifies the frame layout of an X-frame.

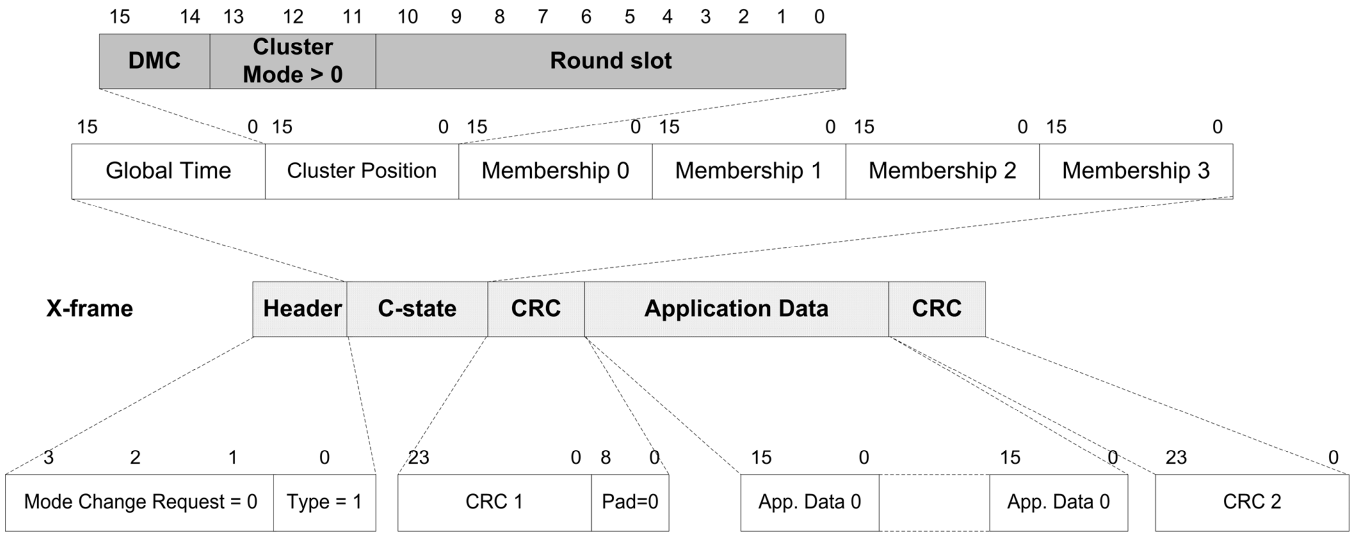


FIGURE 12 - X-FRAME

An X-frame shall contain the following parts:

- Frame type identifier set to 1 (explicit C-State)
- Mode change request according to the mode change request field in the host interface (see section 8.6)
- C-State
- CRC 1 calculated over the frame header and the C-state
- 8 padding bits for 16 word alignment (with value 0x00, which does not change the value of the CRC 2)
- Application data – frame size: 1 to maximum data length (see Table B1) - configurable per round slot
- CRC 2 calculated over the frame header and the C-state, CRC 1. 8 padding bits and the application data.

During the integration process of a node, X-frame shall be used as an I-frame, meaning that the pad byte, the application data and the CRC 2 are ignored in this case.

## 5.4 Frame Status

After reception, a TTP controller shall determine the frame reception status of each received frame. This is an evaluation of the frame contents in order to detect transmission faults or C-state disagreement. The TTP controller shall use the frame status for the acknowledgment algorithm and the clique detection which is described in Chapter 8. Sections 5.4.1 – 5.4.7 provide detailed descriptions of the frame status, and Figure 13 shows the algorithm for the frame status calculation. Section 5.4.8 provides the description for calculation of the slot status.

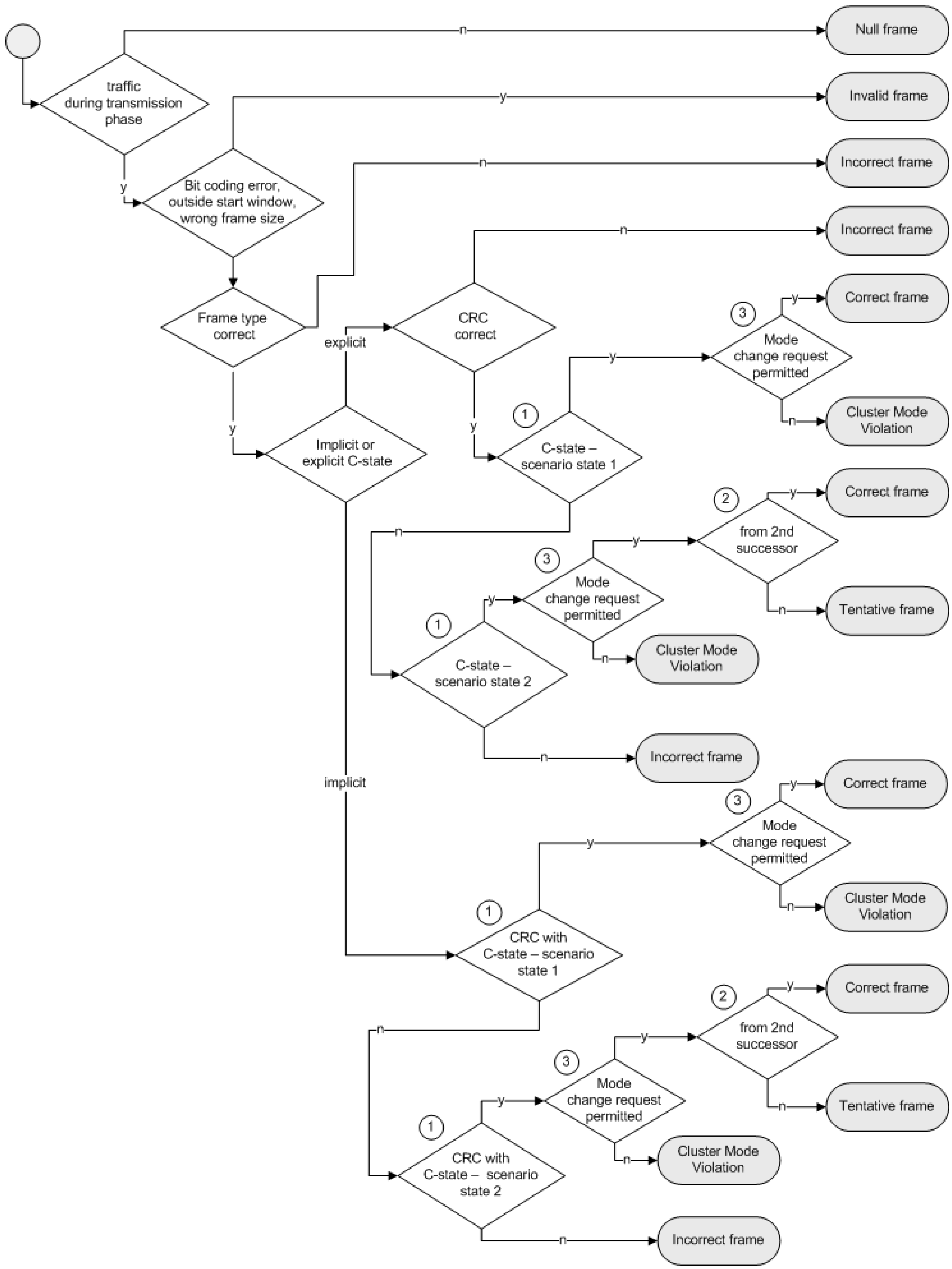


FIGURE 13 - FRAME STATUS CALCULATION

1 Described in section 8.3.2

2 Described in section 8.3.3

3 Described in section 8.6

### 5.4.1 Null-Frame Reception Status

If no activity (not even noise) is observed on the channel of the transmission medium during the transmission phase (from the opening of the receive window until the end of the transmission phase), the expected frame shall be considered to be a null-frame. A missing or fail-silent sender will not send any frames – this will be recognized as null-frames.

### 5.4.2 Valid Frame Reception Status

A received frame shall be considered syntactically valid if the following conditions hold:

- The frame starts during the receive window.
- No code rule violations are observed during the reception of the frame. This implies that a frame consisting of more or less bits than expected by the receiver is detected as invalid.
- No other transmission was active within the receive window before the start of the frame.

### 5.4.3 Invalid Frame Reception Status

If bus activity is detected by the receiver during the time interval between the opening of the receive window and the end of the transmission phase, but no valid frame is received, the frame shall be called invalid.

### 5.4.4 Incorrect Frame Reception Status

An incorrect frame shall be defined as a valid frame with an incorrect CRC (for both C-state scenarios failed as described in section 8.2) or in case of a C-state disagreement.

### 5.4.5 Tentative Frame Reception Status

A tentative frame shall be a valid frame with an agreed C-state according to scenario 2 of the acknowledgement algorithm described in section 8.3.3.

### 5.4.6 Correct Frame Reception Status

A correct frame shall be defined as a valid frame which passed the CRC check and all additional semantic checks at the receiver. Also, frames with an agreed C-state according to scenario 2 received from the second acknowledgement successor (as described in section 8.3) shall be defined as correct frames. The explicit/implicit C-state fully agrees between sender and receiver.

Furthermore, a correct CRC indicates that sender and receiver were connected correctly, since otherwise the different CRC init values (see section 9.3) would result in CRC errors.

### 5.4.7 Cluster Mode Violation

A frame marked with "Cluster mode violation" passes all checks (either the correct or the tentative membership is successful), but holds a mode change request that is not allowed by the MCP in this slot (section 8.6).

### 5.4.8 Slot Status

Because a TTP controller uses two transmission channels, the TTP controller shall combine the states of the received frames in the slot to build the so-called slot status. This is a combination of the frame states with the ‘better’ result taken in the order:

a. Correct

b. Tentative

c. Cluster mode violation

d. Incorrect

e. Null-frame

f. Invalid

This means if the TTP controller receives a correct frame on channel 0 and an invalid one on channel 1, the slot status is 'correct'. The slot status is used for the acknowledgment and the clique detection algorithm described in Chapter 8.

## 6. PHYSICAL LAYER – INTERFACE

Each TTP controller in a cluster shall be connected to the shared broadcast medium – the TTP bus. The data flow from the TTP controller to the TTP bus is depicted schematically in Figure 14 as a three-layer model. Protocol operation shall depend neither on a particular line-encoding scheme nor on certain transmission properties. The TTP controller shall be able to receive and transmit data according to the requirements in Chapter 5. The line coding and physical layer unit shall convert the TTP frame to the bit pattern representing the Bus frame according to the chosen line-encoding scheme and vice versa.

The bus frame encompasses the TTP frame as well as additional bit-patterns to detect the start and end of the TTP frame. The requirements for these additional bit patterns depend on the chosen en*coding* and the *physical layer* properties. A schematic depiction of a Bus frame is given in Figure 15. The requirements for the coding layer and the physical layer are specified in the related standardization projects (e.g. AS6003/1 and AS6003/2).

For compatibility reasons a TTP controller shall provide the following interface requirements to the underlying layers:

- The TTP controller shall simultaneously handle frames on both channels according to definitions of the properties for the communication layer in section 3.2.1. Therefore the following items apply on a channel basis. Since the independent treatment of channels is defined on TTP protocol level, this property shall also be guaranteed on the electrical level by the coding and the physical layer. From a system perspective, each TTP controller connected on a certain TTP bus channel shall be provided with the same line-encoding scheme and the same physical layer. Hence the type of line encoding or the physical layer may differ for both TTP channels.

- The TTP controller shall expect data from and provide data to underlying layers according to the TTP frame format specified in section 5.3.

- The TTP controller shall expect status information with respect to the occurrence of encoding errors for received and transmitted frame data according to section 5.4.

- The TTP controller shall be informed immediately when start of the received TTP frame (SOF) is detected. This is needed for the performance of the synchronization as defined by the requirements in section 7.2.1.

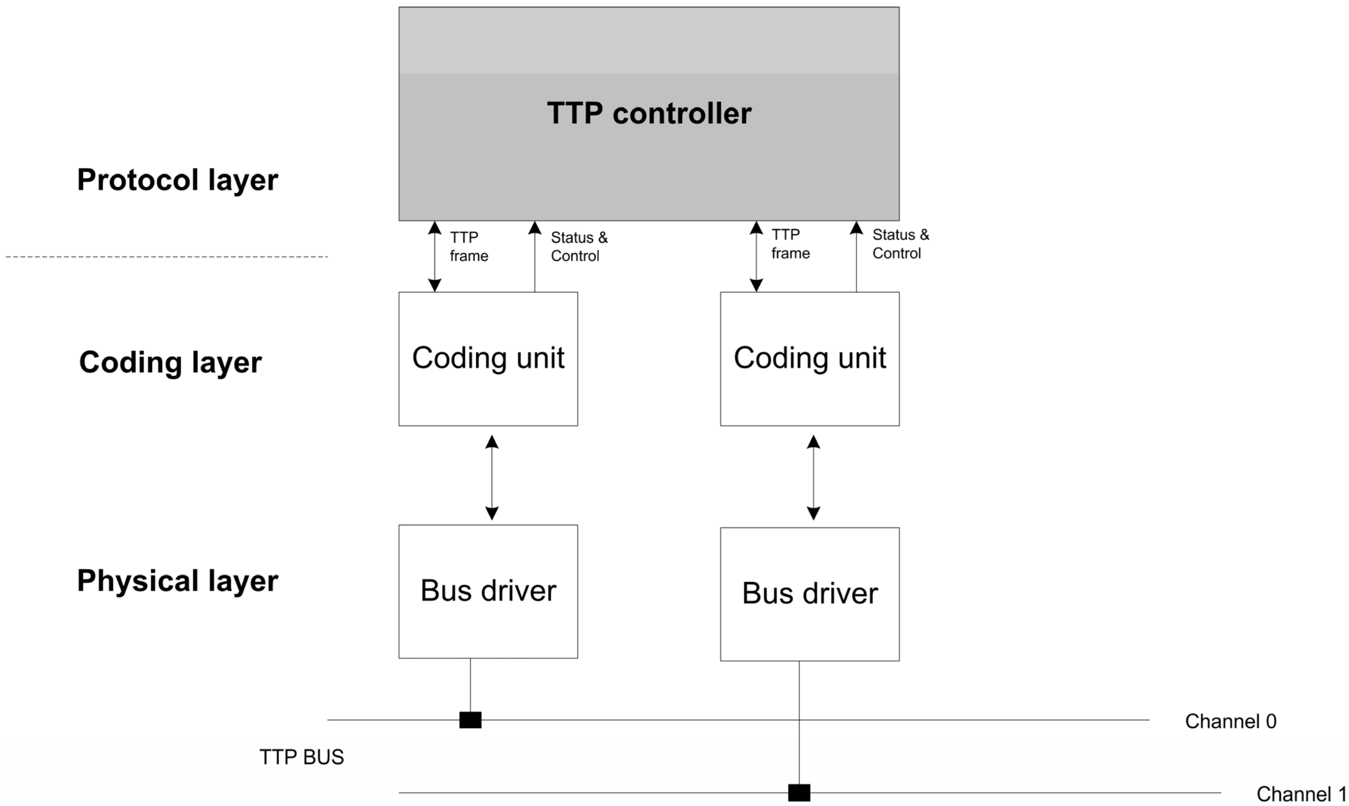


FIGURE 14 - TTP CONTROLLER - PHYSICAL LAYER INTERFACE

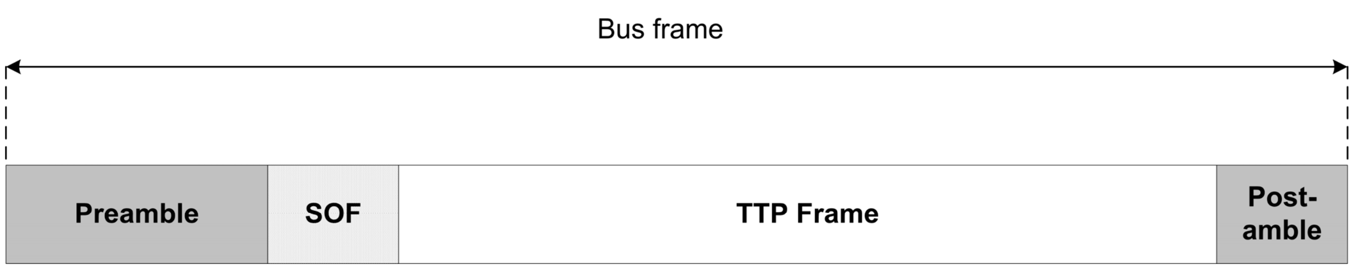


FIGURE 15 - ENCODED FRAME

## 7. FAULT-TOLERANT DISTRIBUTED CLOCK IN TTP

Synchronized and deterministic operation requires the availability of a global time base. Each TTP controller shall adjust its local clock according to this time base on the one hand, as well as being able to contribute to the development of the global time on the other hand. The fault-tolerant distributed clock algorithm describes the parameters and principles of the global time in a TTP network.

### 7.1 Timing Parameters

*Microtick*, *macrotick*, and *precision* are basic terms which are introduced in the following sections.

### 7.1.1 Microtick

A *microtick* shall be a periodic signal that defines the smallest timing measurement unit for a TTP controller. The duration of a microtick $\Delta_{mt}$ shall be related to the physical TTP controller clock, e.g., oscillator. The duration between two consecutive microticks is denoted as granularity *g* of the clock as depicted in Figure 16. $\Delta_{mt}$ shall be used for definition of a *macrotick* length.

### 7.1.2 Macrotick

A *macrotick* shall be a periodic signal that delimits a granule of the global time. This time base is used by the TTP controller to trigger the execution of protocol events and when a frame shall be transmitted or received by a TTP controller. Each macrotick is made up of a number of microticks and shall have a duration $\Delta_{MT}$. Although different nodes in the cluster may have clocks with different granularity *g,* all TTP controllers participating in a cluster shall operate based on the same value for the macrotick $\Delta_{MT}$, exemplified in Figure 16 by Node A and Node B. The macrotick/microtick ratio shall be configured in the configuration data, and may be different for different TTP nodes.

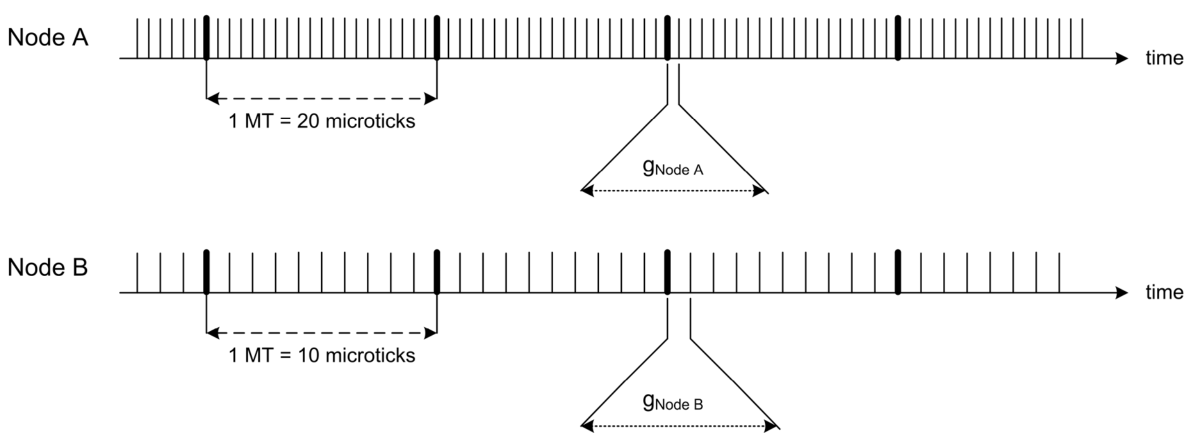


FIGURE 16 - MACROTICK – MICROTICK RELATIONSHIP

### 7.1.3 Precision

The global time base shall ensure synchronous protocol execution and cluster operation among all nodes in the cluster within a configurable time interval – the so-called *precision*. The precision interval $\Pi$ is the time interval in which the same instant of macrotick shall occur on each node as depicted in Figure 17. $\Pi$ shall be specified in the configuration data. Such a time interval is needed due to the fact that the clocks of the individual nodes will drift (e.g., due to environmental conditions). The TTP controller shall periodically check if the execution time of the local macrotick happens within this precision interval relative to the global time. How this is done is described in section 7.2. The actual clock skew of the macrotick clocks in the cluster may be much smaller than the precision interval, but it shall never be larger.

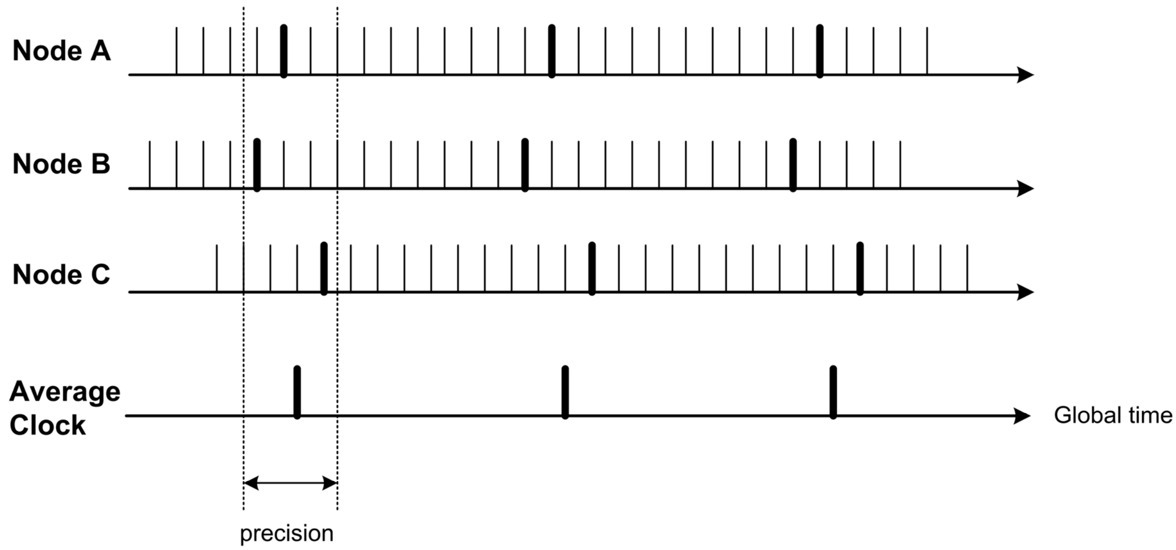


FIGURE 17 - PRECISION

### 7.2 Synchronization

During synchronized network operation, TTP communication protocol shall synchronize local clocks of all TTP nodes to generate a global time base within a certain precision. This shall be conducted in the following three phases:

a. Every TTP controller shall measure the actual arrival time of any received TTP frame and shall calculate its deviation from the expected arrival time, as described in section 7.2.1

b. Every TTP controller shall calculate, based on the recorded deviations, a correction term for its own clock using the distributed clock synchronization algorithm, as described in section 7.2.2.

c. Every TTP controller shall apply the correction term to its local clock to bring the clock into better agreement with the selected clock references found in other TTP controllers, as described in section 7.2.3.

#### 7.2.1 Action Time and Exchange of Timing Information in a TTP Network

The *action time* (AT) is denoted as the point in time when a TTP-frame shall be sent or received by the nodes in a slot. The difference between the planned and actual values for the AT, builds the basis for achieving synchronization among the nodes. Figure 18 depicts the relationship between the different timing parameters describing the time dependencies between the sender and the receiver of a TTP-frame. As can be seen, the sender's clock in this example ticks a little bit faster than that of the receiver. For this reason, the reception of the frame is perceived by the receiver to happen a little bit earlier compared to the computed time instant $t_{AT'r}$. Under ideal conditions, $t_{receptionr}$ happens exactly at the same time instant $t_{AT'r}$. Because conditions are never ideal, $\Delta_{dif}$ is used as a measure of the deviation between two different node clocks.

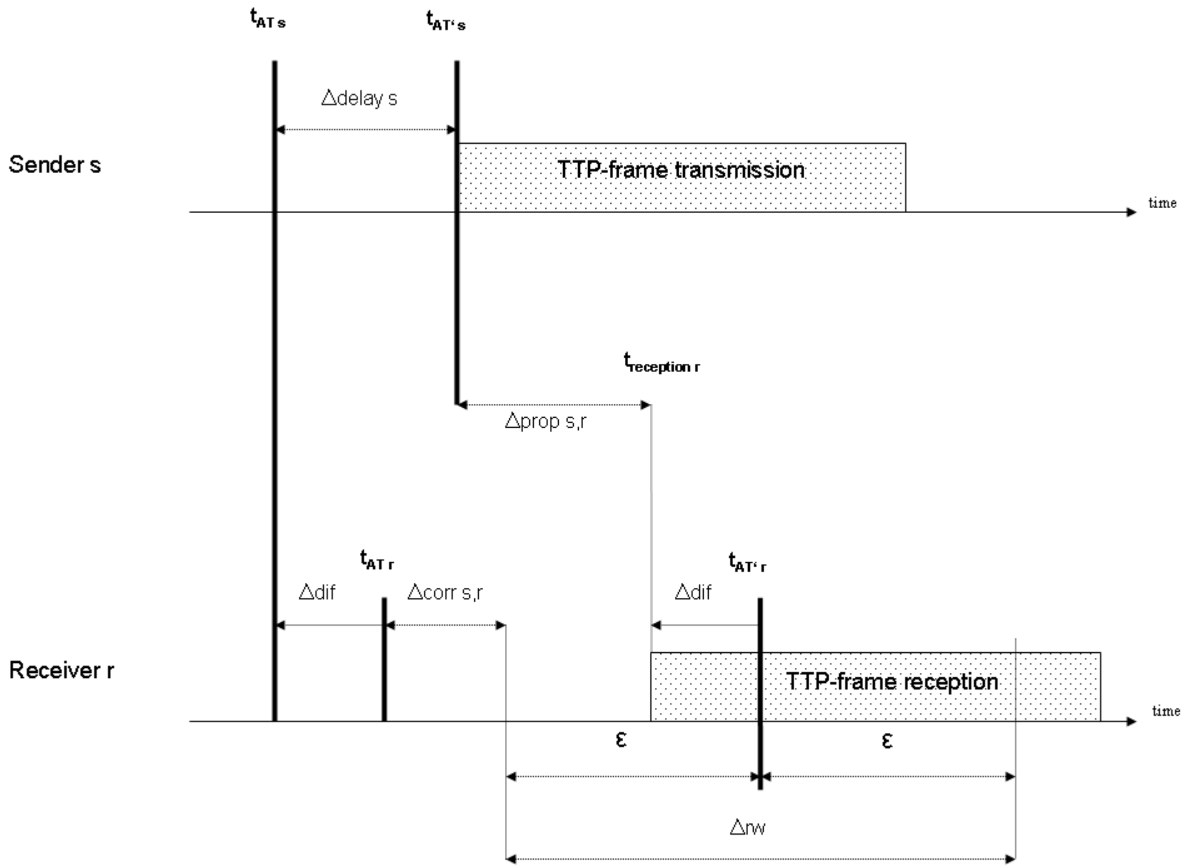


FIGURE 18 - TIME DIFFERENCE CAPTURING

The action time $t_{AT}$ is the point in time at which the TTP controller starts TTP-frame transmission (in its own sending slot) or expects TTP-frame transmission from another TTP controller. The value for any $t_{AT}$ shall be stored in the configuration data. The action time shall be raised synchronously on all TTP controllers within the precision. The described parameters in this section are considered for one channel only – in a system design both TTP channels shall be regarded because of their different propagation delays $\Delta_{prop\,s,r}$.

The receive window $\Delta_{rw}$, shown in Figure 18, of any receiver defines the size of the symmetric interval of valid reception relative to the expected receive time $t_{AT'r}$ of a particular frame. The receive window should have a minimum duration of $2 \times \Pi$ as defined in (Eq. 1), since two nodes may drift within the precision interval in either direction. The size of the receive window shall be specified in the configuration data.

$$\Delta_{rw} = \left[t_{AT'r} - \varepsilon, t_{AT'r} + \varepsilon\right] \qquad \text{(EQ. 1)}$$

For a reception to be valid, a frame start shall be detected within $\Delta_{rw}$. Receptions not started in this window shall be treated as invalid (cf. section 5.4). Captured values based on invalid frames shall not be used for clock synchronization.

The sender shall start transmission at time $t_{AT's}$ (from the sender's local time) which is also called the "delayed action time". As defined in (Eq. 2), $t_{AT's}$ shall be based on the scheduled action time $t_{ATs}$ as perceived by the sender, plus a send delay $\Delta_{delays}$ that guarantees that no perfectly synchronized receiver receives the transmission before the action time as perceived by the receiver. $\Delta_{delays}$ considered at sender's side is needed to compensate the delay evoked by the receivers due to the delay of the start window, and shall be calculated by the TTP controller according to (Eq. 4).

$$t_{AT's} = t_{ATs} + \Delta_{delays} \qquad \text{(EQ. 2)}$$

The expected receive time $t_{AT'r}$ (delayed action time) of the receiver r shall be calculated using (Eq. 3), and depends on the expected frame's action time $t_{ATr} + \varepsilon$ (from the receiver's point of view) and the channel-dependent delay correction term $\Delta_{corrs,r}$ between the sender and any receiver.

$$t_{AT'r} = t_{ATr} + \varepsilon + \Delta_{corrs,r} \qquad \text{(EQ. 3)}$$

(Eq. 4) and (Eq. 5) define the relationship between timing parameters on the sender's and the receiver's side based on action time $t_{AT'r}$. This condition shall be fulfilled for any sender-receiver constellation in each round slot. This means that the values for $\Delta_{props,r}$ and $\Delta_{corrs,r}$ should be determined on a round slot basis (e.g., by measurement). $\Delta_{delays}$ shall be determined for each channel individually. These three values shall be specified in the configuration data.

$$\Delta_{props,r} + \Delta_{delays} = \Delta_{corrs,r} + \varepsilon \qquad \text{(EQ. 4)}$$

$$\Delta_{props,r} \geq 0 \qquad \text{(EQ. 5)}$$

A receiver shall measure the actual time of the received frame $t_{receptionr}$ in microtick granularity. The difference $\Delta_{dif}$ between the scheduled and the measured time for any received frame shall be calculated according to (EQ. 6). $\Delta_{dif}$ represents the difference between the clocks of the sender and the receiver in microticks, and it shall be used as input value for clock synchronization.

$$\Delta_{dif} = t_{receptionr} - t_{AT'r} \qquad \text{(EQ. 6)}$$

If two correct frames from the same node are received on both channels (normal case), the average of the two time difference ($\Delta_{dif}$) values shall be calculated and used as measurement value. If only one correct frame from a node is received, then this value shall be used.

This measurement shall be done for received frames, which were sent by nodes of the master clock set. Nodes with master clocks (master nodes) are nodes in the cluster, whose clock is considered to be accurate enough to be used for clock correction. Frames sent by master nodes shall be defined in the configuration data.

### 7.2.2 Calculation of the Correction Term

In each slot, each TTP controller shall measure the time difference $\Delta_{dif}$ between expected and actual time of frame arrival in microticks, to calculate the correction term for its own global clock.

The resynchronization of global clock actions shall be executed at globally known points in time, simultaneously on all nodes at every *resynchronization interval*. The point in time (the round slot) when the clock correction is carried out, shall be defined in the configuration data. Clock correction shall be performed at least once in a TDMA round. The minimum resynchronization interval, where clock synchronization shall be finished, is restricted by the fact that a minimum of 4 slots with master nodes must be present in the cluster for fault-tolerant clock synchronization.

The Fault-Tolerant Average (FTA) algorithm [Kop87] is used for calculating the clock state correction term (*CSCT*) The CSCT shall be calculated based on the last four measurements (all older measurements shall be ignored), giving the CSCT in units of local microticks by executing the following two steps as depicted in Figure 19:

a. The smallest and the largest measurements of the four measurements shall be discarded.

b. The average of the two remaining measurements is the CSCT for the local clock synchronization.

Error detection – the detection whether the node's local clock is synchronized to the global time – shall be done by checking whether CSCT exceeds the boundaries of the precision interval. In that case, the TTP controller shall report the occurrence of a synchronization error to the host and shall stop operation.

If no correct frames were received or the respective sender is not a master clock node, the values taken for the clock state correction shall remain unchanged. At startup, the TTP controller shall initialize the values taken for clock synchronization with zero in order to prevent a drift in any direction due to uninitialized time difference values.

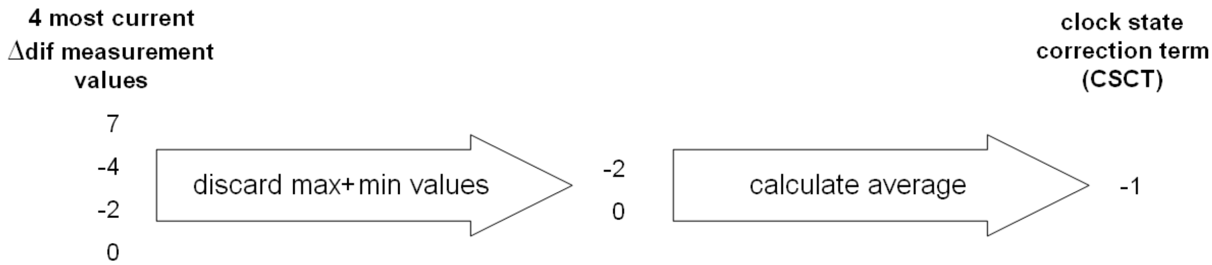


FIGURE 19 - CALCULATION OF CLOCK STATE CORRECTION TERM

### 7.2.3 Correcting the Local Clock

The clock synchronization in a TTP system shall maintain clock drifts on each node within the precision interval $\Pi$. After each repeated calculation of the correction term every resynchronization interval, the TTP controller shall adjust its local macrotick towards the 'average clock' by changing (correcting) the duration of the macrotick by the value of CSCT. Macrotick correction strategy in TTP controllers shall allow:

- a gradual shortening/lengthening of one microtick per macrotick – until the complete CSCT is exhausted (with or without free-running macroticks in-between corrected macroticks, which can be defined in the configuration data)

- an all-at-once reduction/expansion of macroticks, meaning that the CSCT is consumed in one macrotick.

After the CSCT is exhausted, the clock shall run free until the next synchronization instant marked in the configuration data as depicted in Figure 20. Synchronization shall be finished within the resynchronization interval.

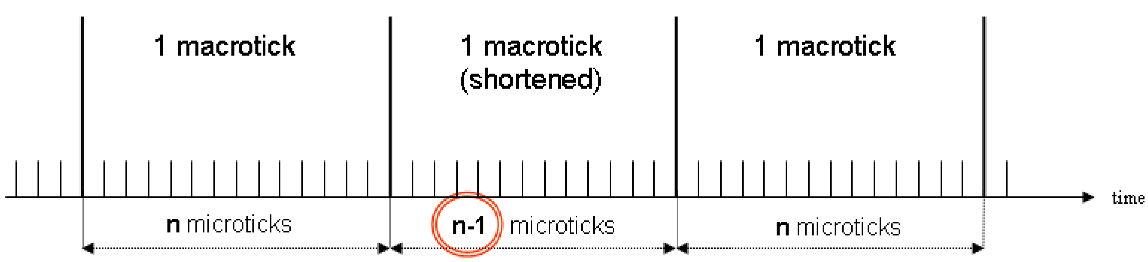


FIGURE 20 - MACROTICK CORRECTION

The clock synchronization algorithm of TTP is formally verified in [Pfe99].

Table B1 specifies the value ranges for TTP controller hardware parameters which shall be fulfilled by any TTP controller. Table B2 shows typical values of, and dependencies between, TTP controller hardware parameters.

## 8. PROTOCOL SERVICES

The TTP protocol defines two different groups of services – *communication* & *safety services* – to guarantee reliable fault tolerant data exchange among the nodes. The provision of these services is important for the operation of safety-critical real-time systems. This section describes in detail the principles and the operation for each TTP service.

**Communication Services** guarantee fail-operational cluster communication service – meaning that single node faults shall be detected and tolerated, and cluster communication shall be continued without loss of quality of service[1]. The communication services encompass:

- *Cluster startup / integration* service described in section 8.1 defines the process for changing a node or the whole cluster from unsynchronized to synchronized operation mode.

- *Acknowledgment* service described in section 8.3 defines the process for the evaluation of membership information for checking the correctness of a sender's frame transmission.

- *Fault-tolerant clock synchronization* service described in Chapter 7 defines the principles and the process for achieving cluster synchronization among all participating nodes.

- *Cluster mode change* service described in section 8.6 defines the principle of cluster modes and the switching between these at run time.

**Safety Services** guarantee a fail-silent behavior of any faulty node to prevent the failure distribution and impact of faults in the cluster. The communication services encompass:

- *Node membership* service described in section 8.2 defines the process which ensures that all nodes are informed about the operational state of all other nodes.

- *Clique detection* service described in section 8.4 defines the process for checking if a node agrees with the majority of the other nodes about the current cluster status prior to sending.

- *Host/controller life-sign* service described in section 8.5 defines the principle and the process for the exchange of life-sign information update between host and TTP controller.

### 8.1 Startup

The process by which a TTP controller changes from an unsynchronized mode to synchronized mode is called *startup*. Depending on whether cluster communication is ongoing at the time the TTP controller is starting up, a TTP controller shall either integrate (see section 8.1.1) or it shall initiate a *cluster startup* which is called cold-start (see section 8.1.2).

#### 8.1.1 Integration

Integration is a process a node performs in order to achieve synchronization in a running cluster communication. Therefore a node shall adopt the C-state from received TTP-frames in order to initialize its own local C-state in order to achieve synchronization.

The decision when the integration process is finished and therefore when a node can acquire a slot, i.e., when it is allowed to start sending in its sending slot, is depicted in Figure 21 and shall depend on the following conditions:

- The node's sending slot is marked in the configuration data.

[1] Assumption: System is designed in accordance with the single fault hypothesis

- The host life-sign has been updated correctly (as described in section 8.5).

- For integration on cold-start frames, an exception is the so-called *free-shot*: a node shall also be allowed to transmit frames with invalid host life-sign, if it is its first sending slot after the C-state becomes valid (after integration) and the current cluster mode is the startup mode.

- For integration on non-cold-start frames: if the integration counter condition is fulfilled, this means that integration is finished when a minimum number of correct frames have been received. This value shall be defined in the configuration data.

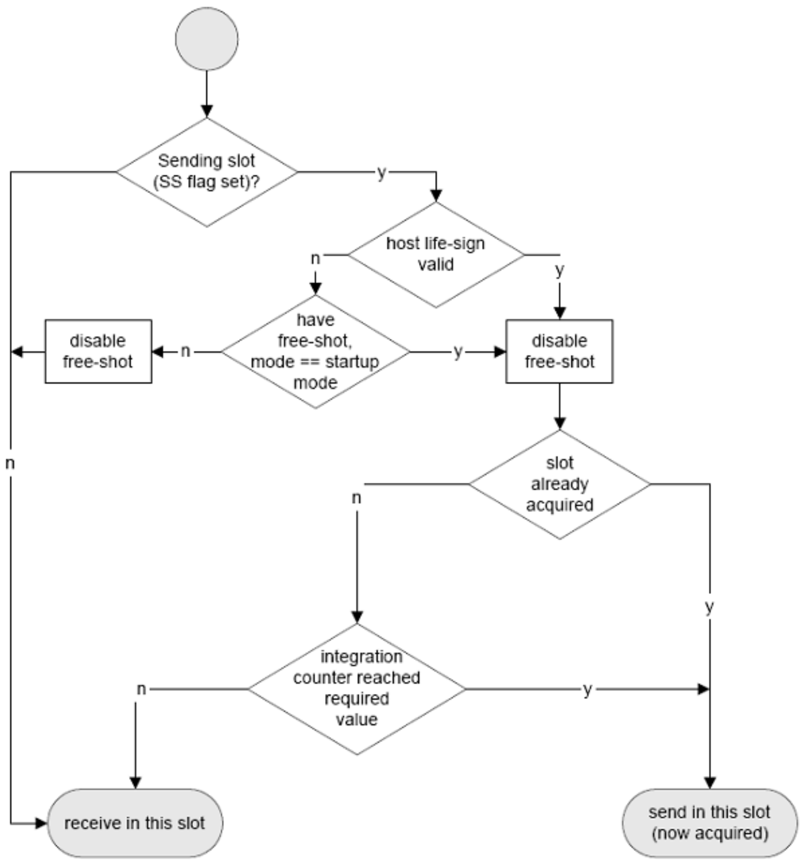


FIGURE 21 - SLOT ACQUISITION OF A NODE

### 8.1.2 Cold-start

A node shall *cold start* when it perceives itself as the first participant in the cluster communication, e.g., after system power on. This means, if the TTP controller didn't receive a TTP-frame with explicit C-state during the period of *listen timeout* (see section 8.1.2.1) it shall initiate cluster communication by sending cold start frames given that the following conditions hold:

- the host has updated its life-sign (as described in section 8.5),

- the TTP controller is allowed to initiate a cold start according to the configuration data and

- the maximum number of allowed cold starts, which is defined in the configuration data, has not been reached.

Virtual member nodes (as described in section 8.2) and nodes configured to transmit frames with implicit C-state (as described in section 5.3.1), shall not be allowed to initiate cluster startup or to integrate on cold start frames. Whether a node is allowed to integrate on cold start frames shall be defined in the configuration data.

*Timeouts* and *Big Bang* are mechanisms which shall systematically be used to overcome bus collisions during the cluster startup phase where asynchronous bus access occurs. A cluster startup phase is finished when at least two nodes in the cluster have established synchronous TDMA-based communication.

8.1.2.1 Timeouts

During cluster startup, nodes access the bus in an asynchronous manner. A cluster-wide phased timeout strategy shall be applied to prevent repetitive collisions.

8.1.2.1.1 Startup Timeout

This unique timeout principle shall guarantee that TTP controllers, each sending a cold start frame at (about) the same time, will not collide when they send a cold start frame the next time. Therefore the value of the *startup timeout* $\Delta_{startup\_timeout_i}$ of a node *i* shall be unique for each node that has the permission to perform a cold start, which is defined in the configuration data. The value for $\Delta_{startup\_timeout_i}$ shall be chosen, such that the cold start frame of any node can be transmitted without collision within a period of a TMDA round. A typical calculation for $\Delta_{startup\_timeout_i}$ is shown in (Eq. 8) . Please note that node index *i* starts with the value zero.

$$\Delta_{startup\_timeout_i} = \sum_{j=0}^{i} \Delta_{slot\_duration_j} \qquad \text{(EQ. 8)}$$

8.1.2.1.2 Listen Timeout

After start-up a TTP controller shall check whether TTP communication is ongoing on the bus. To do this, the TTP controller shall check for the duration of the *listen timeout* $\Delta_{listen\_timeout}$ whether a TTP frame with explicit C-state was received for integration. A typical calculation for $\Delta_{listen\_timeout}$ is shown in (Eq.9).

$$\Delta_{listen\_timeout} = 2 \cdot \Delta_{round} + \Delta_{startup\_timeout_i} \qquad \text{(EQ.9)}$$

This choice for $\Delta_{listen\_timeout}$ ensures that the longest duration between two cold starts of a node (startup timeout + duration of first TDMA round in first cluster mode) is shorter than the shortest $\Delta_{listen\_timeout}$ . This value shall be defined in the configuration data.

8.1.2.1.3 Cold Start Timeout

The cold start timeout $\Delta_{coldstart\_timeout}$ as defined in (Eq. 10), specifies the time between two successive cold start frames sent by a cold starting node. A cold starting TTP controller sends repetitive cold start frames either due to the Big Bang mechanism (as described in section 8.1.2.2) or due to colliding TTP frames as depicted in Figure 22.

$$\Delta_{coldstart\_timeout} = \Delta_{round} + \Delta_{startup\_timeout_i} \qquad \text{(EQ. 10)}$$

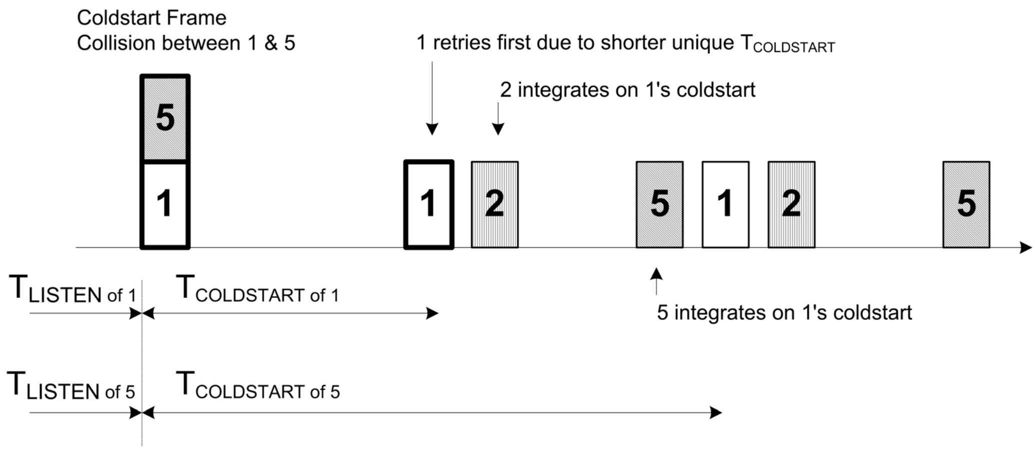


FIGURE 22 - COLD START SCENARIO

8.1.2.2 Big Bang

A mechanism called the Big Bang shall ensure that in case of a startup collision between two cold starting nodes no node will integrate on any of the collided frames. If a collision is detected by all nodes consistently, the Big Bang mechanism would not be needed, but in case of long propagation delays $\Delta_{prop} > \Delta_{coldstart\ frame\ duration}$ subsets of nodes may integrate on different cold starters. This leads to the building of startup cliques.

To prevent such startup cliques, a TTP controller shall reject the first received correct cold start frame and shall restart the $\Delta_{listen\_timeout}$ . As a consequence, the cold starting nodes will not detect traffic during the period of one TDMA round and shall start their startup timeouts after they recognized a communication blackout error (as defined in section 8.4.2). Due to the unique definition of the startup timeout values as defined in section 8.1.2.1, no further inconsistent cold start frame collision will appear.

Figure 23 shows a cluster scenario and Figure 24 a startup scenario with nodes A and E as cold starters. Without the Big Bang, only node C detects the collision. Nodes B and D will integrate on the frame of the node in their vicinity (Node A and Node D accordingly).

Using the Big Bang as depicted in Figure 24, all nodes except C integrate on A, because node A has a shorter startup timeout than E. Node C rejects the second cold start frame as its Big Bang, but will integrate on a frame of one of the synchronized nodes, e.g., node B sending after A.

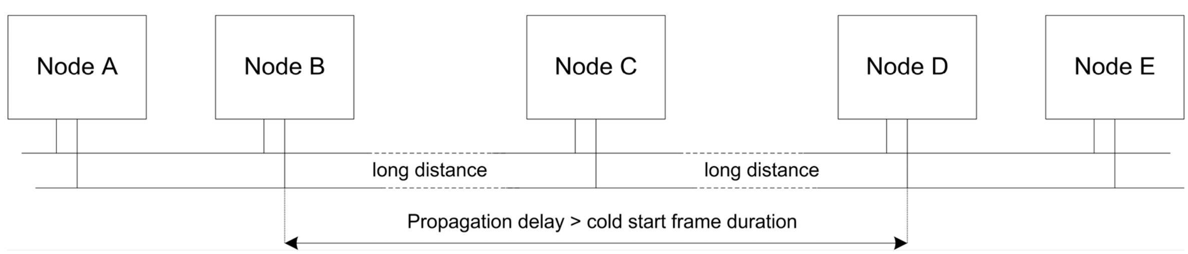


FIGURE 23 - CLUSTER SCENARIO

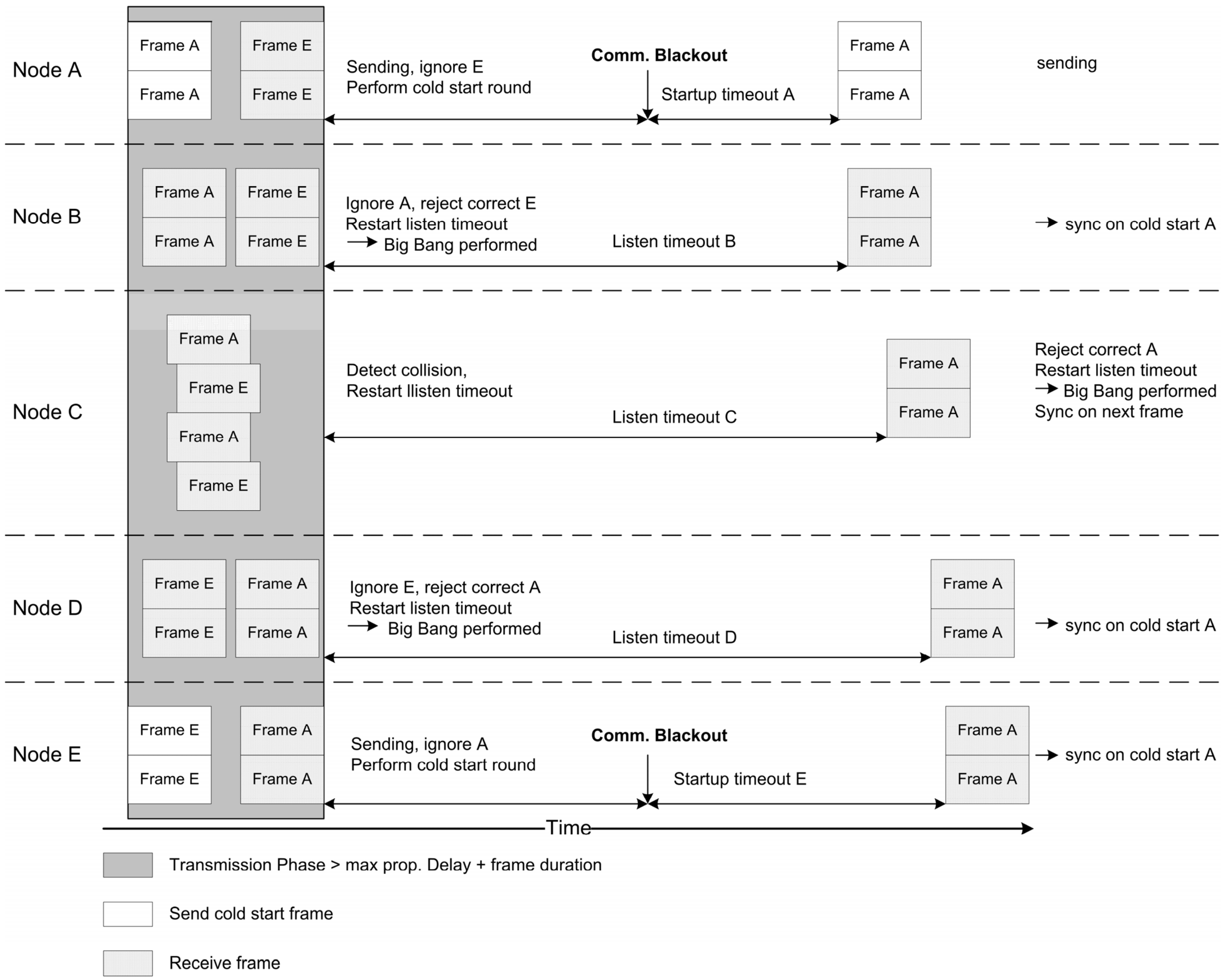


FIGURE 24 - STARTUP COLLISION – USING BIG BANG

## 8.2 Membership

The node membership service (denoted as membership service), formally verified in [Pfe00], shall ensure that all actively sending nodes communicate and adjust their local view about the operational state of each node within a latency of one TDMA round (except multiplexed nodes as described below). Therefore a TTP controller shall maintain the membership vector on frame level according to the format definition in section 5.2.2. The TTP controller shall maintain slot status statistics for each frame reception and transmission by continuously tracking the results of agreed or failed slot status based on the following criteria:

- The TTP controller shall vote a ”correct” slot status as agreed slot
- The TTP controller shall vote an ”incorrect” or ”invalid” slot status as failed
- The TTP controller shall not consider ”null frame” slot status for statistics

### 8.2.1 Membership Vector

A *member node* is a node which is allowed to transmit data in its dedicated slot. The membership status of each member node is recorded in the membership vector. The membership vector shall be a vector of flags, whereby one membership flag is assigned to each member node. The position of the flag for each node shall be defined in the configuration data. That means that silent nodes (meaning nodes which are not assigned to a slot) shall not be represented in the membership vector.

A TTP controller shall vote on the operational status of the other nodes based on its perception of the slot status (see section 5.4.8). The TTP controller shall reset the membership flag in its local C-state when the slot status evaluates to a value other than “correct” – meaning the sender in the current round slot shall be considered to be non-operational. Depending on the voting result, the TTP controller shall update the slot status statistics. This evaluation shall take place in the PRP. This point in time is also called the *membership point*. The decision about a node’s membership status remains unchanged until the next membership point.

A transmitting node makes a final decision about its own membership status at the *membership recognition point*. The actual point in time of the occurrence of the membership recognition point depends on the node’s slot position and the result of the *Acknowledgement algorithm* described in section 8.3.

### 8.2.2 Multiplexed Slots

It shall be possible that several nodes can share a single node slot to improve bandwidth utilization. The set of nodes that share a node slot shall be called a *virtual member node* [Pal97]. Each of these nodes shall be statically assigned to particular TDMA rounds. There is thus no conflict about the point in time when a node will send a frame.

Each node of a set of virtual member nodes shall have its own membership flag like a real member node. From the membership point of view, a virtual member node is a real member node. The membership flag only reflects the activity status of this specific node. Due to the reduced send period, however, this flag is updated with a lower frequency than the membership flags of real member nodes.

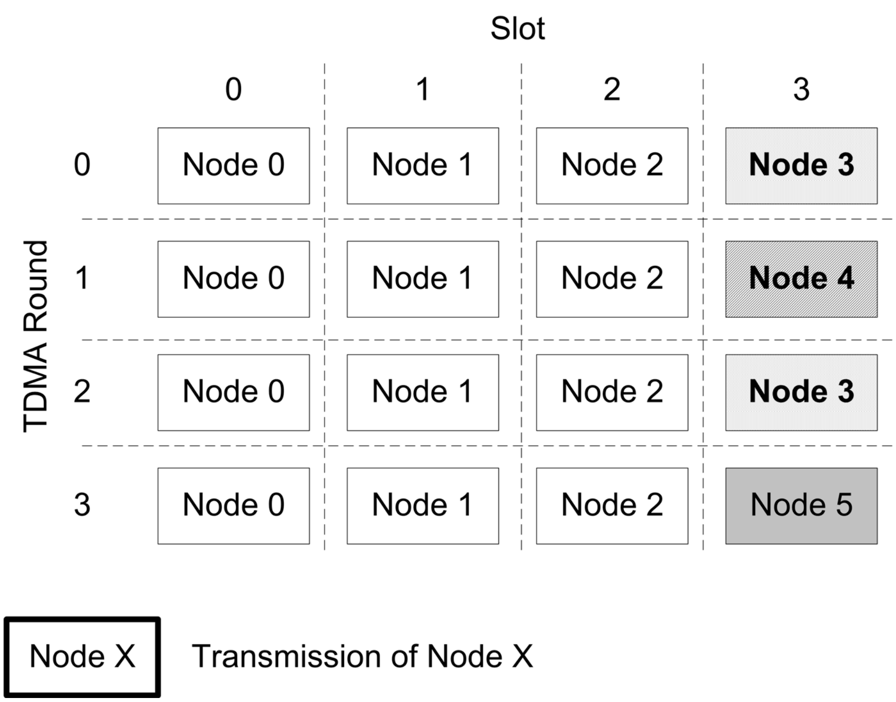


FIGURE 25 - MULTIPLEXED SLOT ASSIGNMENT

Figure 25 shows an example of a cluster cycle consisting of four TDMA rounds. The last slot is shared by the virtual member nodes 3, 4 and 5, with node 3 sending in the TDMA round 0 and 2, node 4 sending in round 1 and node 5 in round 3. Node 3 has half the transmission frequency of a real member node having a sending slot in each TDMA round. Node 4 and node 5 have a quarter of the transmission frequency of a real member node because they both send only once during a cluster cycle while, e.g., node 0 sends four times.

In different cluster modes, the assignment of virtual member nodes to TDMA rounds may change.

## 8.3 Acknowledgment

The acknowledgement service shall handle the information-gathering process regarding the correctness of a sender's frame transmission. The TTP acknowledgment service shall be performed implicitly based on the membership service described in section 8.2. The sender shall extract acknowledgment from a maximum of two valid received successor frames by checking whether the C-state membership vector of the received frames is in accordance with the scenarios defined by the acknowledgement algorithm. Depending on the given situation, the TTP controller shall update the slot statistics (as mentioned in section 8.2) as well as statistics with respect to the successive occurrence of acknowledgement errors.

### 8.3.1 Acknowledgement Algorithm

The successor relationship is dynamic, depending on the current sender position in the TDMA round and the slot status of the received frames. The first or the second successor node of the sender shall determine whether the sender has succeeded in sending at least one correct frame – as depicted in the example shown in Figure 26. Depending on the successor's reported membership vector, the status with respect to acknowledgement shall be evaluated using an acknowledgement algorithm that has two states as described below.

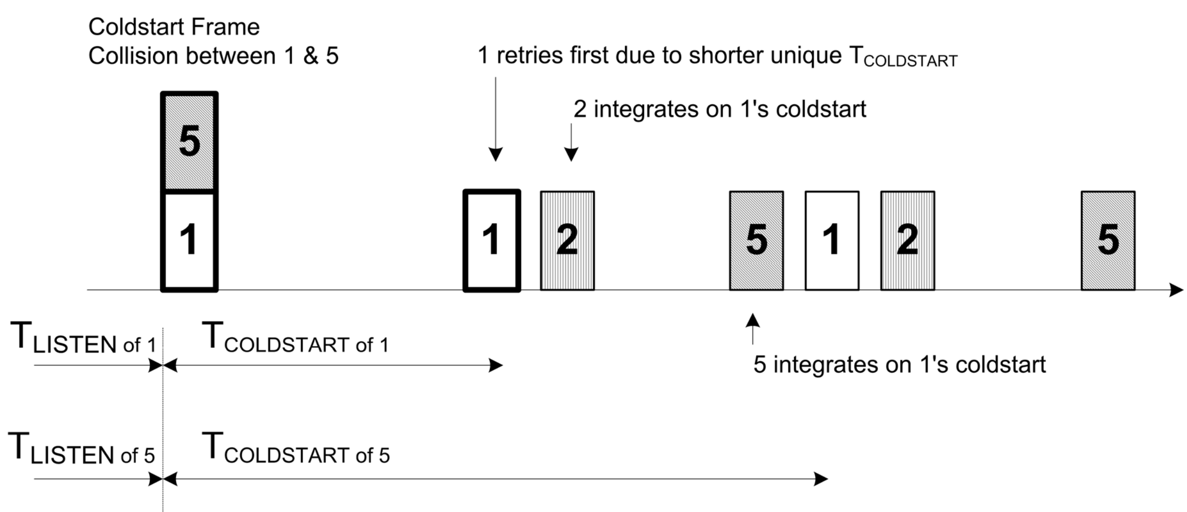


FIGURE 26 - MEMBERSHIP DECISIONS

### 8.3.2 Acknowledgement Algorithm – State 1

Node *A* sends a frame in its sending slot according to Figure 26, when *A* perceives itself as fully operational in its PSP, and shall therefore set its membership flag (denoted as MEMB(*A*)) in the membership vector.

If A's first successor, node *B*, has received at least one correct frame from node *A*, *B* shall set MEMB(*A*) in its local C-state for *B*'s imminent frame transmission.

*A* shall only consider transmissions from *B* for acknowledgment if it has received at least one valid frame from *B*. If this condition is not fulfilled, *A* shall reset MEMB(*B*) in its local membership vector, and shall use the next sender *C* as first successor.

In this acknowledgement algorithm state, *A* shall perform two CRC checks – **check 1a** and **check 1b** – based on *B*'s frames and based on two different membership vector scenarios as depicted in Figure 27. If one of the two CRCs matches, *A* knows *B*'s opinion about *A*.

**check 1a**: *A* shall set MEMB(*A*) and shall set MEMB(*B*) in its local C-state. Then it shall perform the CRC check over the received frame from *B* and its local C-state.

**check 1b**: *A* shall reset MEMB(*A*) and shall set MEMB(*B*) in its local C-state. Then it shall perform the CRC check over the received frame from *B* and its local C-state.

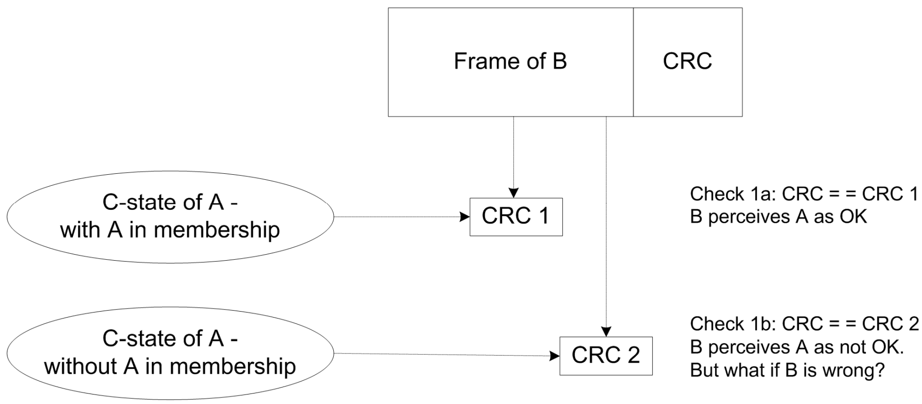


FIGURE 27 - ACKNOWLEDGEMENT ALGORITHM – $1^{ST}$ SUCCESSOR SCENARIO

If **check 1a** passes, *A* assumes that the transmission was correct and remains in the membership. Node *A* shall update its slot status statistics with respect to correct frame reception and shall reset the acknowledgement failure statistics. In this case, the decision about *A*'s membership in this TDMA round is final.

If **check 1b** passes, then at least one failure has occurred – meaning that either *A*'s transmission was faulty or corrupted, or that *B* made some error. Hence, the opinion of *A*'s second successor shall be used to retrieve acknowledgement. The update of the membership bits and the statistics shall be continued as described by the Acknowledgement algorithm state 2 in section 8.3.3.

If both checks (**check 1a** and **check 1b**) fail, it shall be assumed that a transient disturbance has corrupted *B*'s frames or *B* is not operational at all. If transmission activity was observed on both channels in that case, *B* is considered to have transmitted and failed, and *A* shall remove *B* from membership, shall update the slot statistics with respect to erroneous perceived slot status, and shall continuously look for a first successor.

If an invalid transmission activity was observed on only one channel, and silence on the other channel, transient noise is assumed. This kind of observation shall not update the frame statistics. *B* loses its membership and *A* shall continuously look for a first successor.

For example, **check 1b** will also fail if A transmits a mode change request in its frames, since B did not update its C-state with the mode change information (DMC field or mode number). The C-states of A and B will therefore differ by more than just the membership flag of A.

8.3.3 Acknowledgement Algorithm – State 2

It is possible for **check 1a** to fail and for **check 1b** to pass. The data transmitted by *B* shall be perceived as 'suspicious' by *A* at this time, as it is unknown whether *A* or *B* is correct. *A* therefore shall mark the frame data from *B* as 'tentative' in the frame status field (see section 5.4) and shall not change the membership flag of *B*. *A*'s final decision is made after receiving the next frame from *A*'s second successor, which is *B*'s first successor, node *C*.

Based on the frames from *C*, the original sender *A* shall perform two CRC checks over the valid frames received from its second successor as depicted in Figure 28.

**check 2a**: *A* shall set MEMB(*A*) and shall reset MEMB(*B*) in its local C-state. Then it shall perform the CRC check over the received frame from *C* and its local C-state.

**check 2b**: *A* shall reset MEMB(*A*) and shall set MEMB(*B*) in its local C-state. Then it shall perform the CRC check over the received frame from *C* and its local C-state.

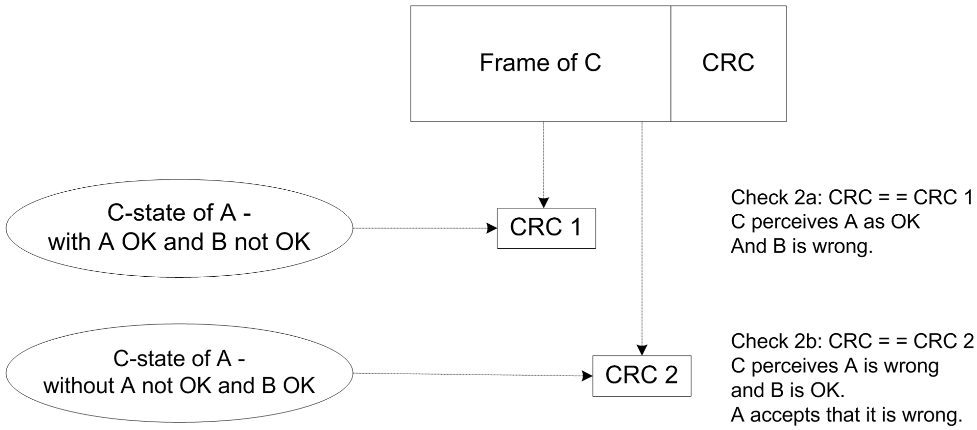


FIGURE 28 - ACKNOWLEDGEMENT ALGORITHM – 2$^{ND}$ SUCCESSOR SCENARIO

If the CRC of **check 2a** is OK, *A* assumes that its original transmission was correct and successor *B* was in error.

If the CRC of **check 2b** is OK, *A* assumes that the original transmission was erroneous and the successor *B* was correct.

Depending on whether **check 2a** or **check 2b** succeeds, *A* shall set the membership bits in its local C-state according to the applicable scenario. Furthermore, *A* shall update the slot status statistics and shall update the acknowledgment error statistics in case that **check 2b** applies or shall reset it in case of **check 2a**. If the number of successive acknowledgement errors reaches the maximum acknowledgment failure count value, which shall be defined in the configuration data, the TTP controller shall report an acknowledgment error in the host interface and shall stop operation. The decision is final for this TDMA round.

If neither **check 2a** nor **check 2b** succeeded, or A received a null-frame in C's slot, then MEMB(C) shall be reset, the failed slot statistics shall be updated due to *C*'s incorrect frame transmission and the next node shall become the second successor.

8.3.4 Sequence of Checks

If **check 1a** succeeds for the frame on at least one channel, the result of **check 1b** shall not be regarded. In the same manner, if **check 2a** succeeds for the frame on at least one channel, the result of **check 2b** shall not be regarded by the TTP controller. The point in time when the decision about the sender's membership is final, is called the *membership recognition point*.

Table 2 summarizes the different scenarios in context of the membership vector and the slot status statistics.

TABLE 2 - ACKNOWLEDGEMENT ALGORITHM – SUMMARY

| CRC of Check | | | | Action | | | | | | Comment | Decision after |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **1a** | **1b** | **2a** | **2b** | **MEM (*A*)** | **MEM (*B*)** | **MEM (*C*)** | **Agreed Slots** | **Failed Slots** | | | |
| T | | | T | T | T | | +1 | 0 | | *A* acknowledged | 1st successor |
| F | F | | | T | F | | 0 | +1 | do check 1 with next successor | Frames corrupted | |
| F | T | T | | T | F | T | +1 | +1 | | 1st successor failed | 2nd successor |
| F | T | F | T | F | T | T | +1 | +1 | slot acquirement lost | *A* failed | 2nd successor |
| F | T | F | F | T | F | F | 0 | +1 | do check 2 with next 2nd successor | Frames corrupted | |

## 8.4 Clique Detection

A *clique* is defined as a number of nodes that agree on the same C-state continuously communicated by every frame transmission. But due to error conditions or inconsistencies, a group of nodes may perceive other nodes in the cluster as incorrectly transmitting and erroneous. Hence, potentially different cliques could coexist within the same cluster. The TTP controller shall overcome the forming of cliques within one TDMA round through interpreting the slot status statistics and by acting in accordance with the clique detection algorithm defined in sections 8.4.1 and 8.4.2. This analysis shall be done in the PSP of a dedicated slot, which is defined in the configuration data.

### 8.4.1 Clique Error

Each TTP controller shall evaluate once in a TDMA round whether it is in the majority clique by checking if it has perceived more agreed slots than failed slots. If this condition is not fulfilled, meaning that the node detects that it is not consistent with the majority of the nodes, the TTP controller shall report a clique error to the host and shall stop operation.

### 8.4.2 Communication System Blackout

The TTP controller shall check if it has received at least one frame during the last TDMA round. If not, the check shall indicate that no correct transmission activity was observed in the last TDMA round, except possibly for the node's own. The TTP controller shall report a communication blackout error to the host and shall stop operation.

## 8.5 Host/Controller Life-sign

The host on the one hand and the TTP controller on the other hand shall provide periodic updates of life-sign information via the host interface to notify each other of their activity. Based on the value of the TTP controller life-sign, the host shall calculate the value for the host life-sign.

A TTP controller shall verify the correctness of the host life-sign in the PSP of its sending node slot. Therefore the update of the host life-sign shall be expected by the TTP controller to happen between the transmission phases of the node's sending slot and the beginning of the PSP preceding the node's next sending slot. Typically this life-sign update happens once in a TDMA round. For multiplexed nodes the life-sign happens less often according to their sending period as described in section 8.2.2.

If the host life-sign does not match the expected value, the TTP controller shall stop frame transmission until the host resumes correct life-sign updates. This means that a TTP controller shall remain synchronized and shall receive frames but shall not be allowed to send until the host life-sign is updated correctly. An updated host life-sign shall also be a pre-condition for a TTP controller to start up a cluster by sending a cold start frame.

After every check of the host life-sign by the TTP controller, the TTP controller:

- shall provide an update of the TTP controller life-sign field to the host to notify it of its activity and
- shall reset the host life-sign field in the host interface after processing.

## 8.6 Cluster Modes

Many real-time systems exhibit mutually exclusive phases of operation and control. For example, an aircraft can be on the ground in maintenance mode, or in flight mode. Each of these mutually exclusive phases can be called an operation mode. With respect to protocol execution, each operation mode may be assigned to a cluster mode. The cluster modes may differ with respect to frame constitution and the associated set of configuration data (transmission schedule parameters). A TTP controller shall support different cluster modes and shall manage the change from one cluster mode to another (cluster mode change). Cluster mode changes shall be requested by the host (this is a so-called *cluster mode change request*). A host's cluster mode change request is distributed by the TTP controller in its sending slot (in the frame header's mode change request field as described in section 5.3) to the other nodes of the cluster. A TTP controller receiving a cluster mode change request shall check whether the mode change request is permitted according to the mode change permission in the configuration data. If it is allowed, a TTP controller shall take over this request in its local C-state (DMC field). A TTP controller shall not perform cluster mode changes immediately, but at the start of a new cluster cycle. The changes are therefore called *Deferred Mode Change*s. A pending cluster mode is communicated by all nodes in DMC field until it becomes permanent at the end of the current cluster cycle. If another valid cluster mode is requested before the first one is executed, the new request shall overwrite the old one. The following rules shall apply to all defined cluster modes:

- All cluster modes shall be based on the same node slot sequence.
- All synchronized nodes in a TTP cluster shall operate at the same time in the same mode. (This condition is checked by the C-state agreement as described in section 5.2.2)

The process of a cluster mode change can be divided into the following three distinct phases.

### 8.6.1 Distribution Phase

The TTP controller shall check in the PSP of its sending slot whether the host has requested a mode change (via the host interface). The TTP controller shall read and clear (set to "no request") the request afterwards in the host interface. The permission/acceptance of a specific mode change shall be defined in the configuration data specifically for each round slot and node individually. The TTP controller shall check whether the host's request was valid. If the request was valid, the TTP controller shall set the mode change request field in the frame header for the frame to be sent, to the value of the requested deferred successor mode. If no mode change was requested the mode change request field shall be set to the value of "clear pending request". Otherwise the TTP controller shall report a mode violation error in the host interface and stop frame transmission.

### 8.6.2 Acceptance Phase

The TTP controller shall check in the PRP, whether the sender of a frame has requested a mode change. Then the TTP controller shall check if the mode change request transmitted in the frame header is allowed, according to the definitions in configuration data. If it is, the TTP controller shall set the value of the local C-state's DMC field to the value of the requested mode change (see section 5.2.2). This value shall be distributed by all active TTP controllers in the DMC field until the mode change "execution phase" has been reached (see below). The reception of frames with an invalid or not permitted mode change request shall be reported as mode change violation error in context of the frame status to the host.

A TTP controller shall evaluate the cluster mode change request based on frames correctly received on channel 0. Otherwise the TTP controller shall handle the mode change request based on frames received on channel 1.

#### 8.6.3 Execution Phase

In the first slot of every cluster cycle a TTP controller shall set the cluster mode field of the C-state to the mode requested by the C-state’s DMC field and shall set the DMC field to “no request” afterwards. The TTP controller shall then continue operation based on the new cluster mode-specific transmission schedule parameters in the configuration data.

## 9. PROTOCOL STATES

The protocol states described in this section define the operation of the Time-Triggered Protocol by means of a state machine. Section 9.1 provides an introduction to the protocol states and the corresponding transition criteria, while section 9.2 explains in detail the operation sequence of how the TTP services described in Chapter 8 are executed in the context of the protocol states.

### 9.1 Protocol States and State Transitions

This state machine defines how the TTP services relate to each other in the time domain. This section provides an overview of the protocol states and the state transitions.

#### 9.1.1 TTP Controller – Protocol State Overview

In the course of this section the protocol states are going to be introduced in a brief form.

Table 3 completes the overview with the most important properties of each protocol state.

**Freeze state:** This is the initial state of the TTP controller at power-on. The protocol shall transit to this state in case of any protocol error (e.g., synchronization error). The execution of the protocol is halted until the TTP controller is turned on by the host.

**Init state:** In this state, the TTP controller shall initialize all status fields and prepare for protocol operation.

**Listen state:** In the listen state, the TTP controller shall attempt to integrate on a received frame with synchronization information (I-frame, X-frame or cold start frame).

**Cold start state:** In this state, the TTP controller shall facilitate the integration of other TTP controllers by periodically sending cold start frames until it receives a response from another TTP controller.

**Active state:** In this state, the TTP controller is synchronized to the TTP network and it transmits and receives frames as defined by the configuration data.

**Passive state:** In this state, the TTP controller is synchronized to the TTP network but it shall not transmit any data. It shall receive frames as defined by the configuration data.

TABLE 3 - PROTOCOL STATE PROPERTIES

| Protocol State | Impact of the host activity in this protocol state | TTP controller On (CO) Flag | Sending frames | Synchronized with TTP network |
|---|---|---|---|---|
| **Freeze** | None | Off | No | No |
| **Init** | None | On | No | No |
| **Listen** | None | On | No | No |
| **Cold start** | Host is active (life-sign) | On | Yes, cold start frames only if configured | Yes, after the TTP controller receives one correct frame (the synchronization is initiated in this state) |
| **Active** | Host is active (life-sign) | On | X-, N- or I-frames (depending on configuration) | Yes |
| **Passive** | Either the host is inactive, OR the host is active, but the TTP controller is configured to play the passive node | On | No | Yes |

9.1.2 TTP Controller – Protocol State Transitions

The process of the protocol execution can be described by a state machine. Figure 29 depicts the possible states and state transitions. The states drawn with gray background denote states where the TTP controller is synchronized with the rest of the cluster. States drawn with a dotted background denote states where the TTP control sends frames on both channels. The transitions between these states are caused by the events listed in Table 4.

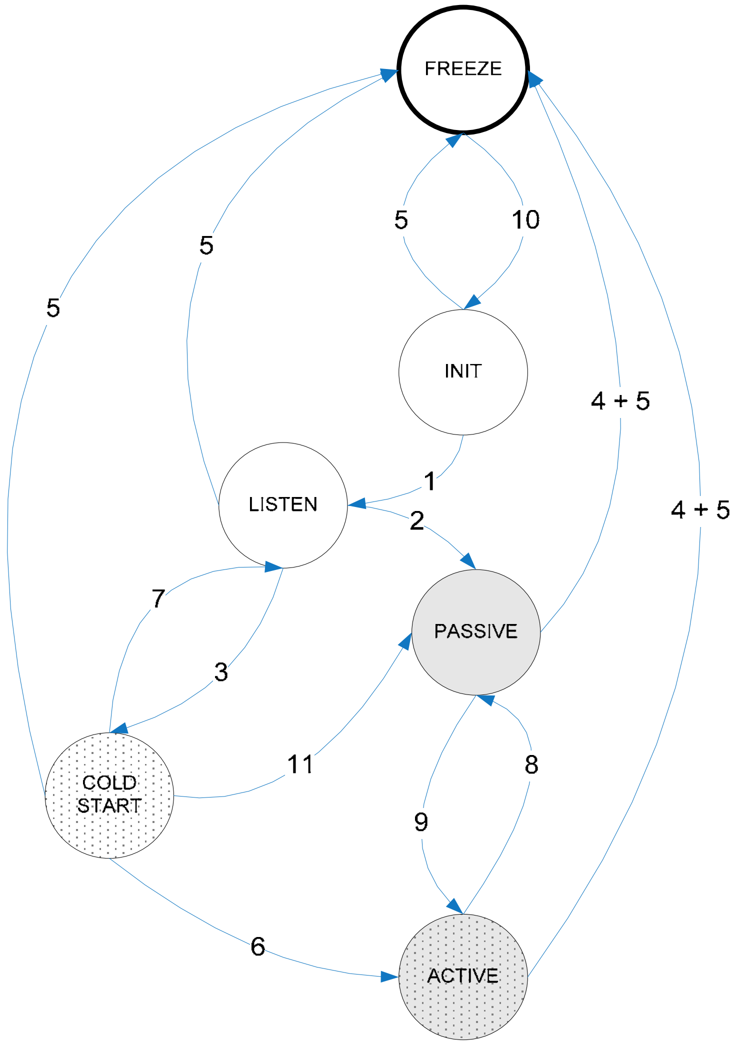


FIGURE 29 - PROTOCOL STATE TRANSITIONS OF THE TTP CONTROLLER

TABLE 4 - PROTOCOL STATE TRANSITIONS OF THE TTP CONTROLLER

| Transition No. | From State | To State | Conditions |
|---|---|---|---|
| 1 | **Init** | **Listen** | Initialization complete |
| 2 | **Listen** | **Passive** | Correct frame containing explicit C-state (I-frame or X-frame) received |
| 3 | **Listen** | **Cold start** | Listen timeout expired, cold start is allowed by configuration, and host life-sign is updated |
| 4 | **Active, Passive** | **Freeze** | Clique error, communication blackout, synchronization error, acknowledgment error, configuration data CRC check failed |
| 5 | **Listen, Cold start, Active, Passive, Init** | **Freeze** | TTP controller turned off, initialization error |
| 6 | **Cold start** | **Active** | TTP controller in majority clique (after clique detection), at least 1 correct frame received (min. 2 TTP controllers alive), host life-sign updated |
| 7 | **Cold start** | **Listen** | TTP controller in minority clique, host life-sign not updated, max. cold start entries exceeded or traffic detected during startup timeout |
| 8 | **Active** | **Passive** | Host life-sign not updated (host not alive), mode violation error, acknowledgment failed |
| 9 | **Passive** | **Active** | Node slot acquired, host alive update OK |
| 10 | **Freeze** | **Init** | Controller On field set, status field reset and TTP controller ID updated |
| 11 | **Cold start** | **Passive** | TTP controller in majority clique but host life-sign not updated or mode violation error (see transition 6) |

## 9.2 Operation in TTP Protocol States

This section provides a detailed description of the operations within the TTP states. This description provides the context of interaction between the TTP services and the protocol state transitions between those.

### 9.2.1 Protocol Variables

Protocol variables which shall represent the status of the protocol services as defined in Chapters 7 and 8 during operation are introduced in this section.

*Cold start counter* → shall count the number of performed cold start attempts of a TTP controller.

*Integration counter* → shall count the correct slots after integration on a non-cold start frame. This is used to define the number of correct slots before the node is allowed to acquire the slot and become synchronized.

*Big Bang flag* → shall indicate a recognized big bang frame – after the initialization, the first received correct cold start frame is ignored, and the listen timeout is restarted (see section 8.1.2.2).

*Free-shot flag* → shall indicate a TTP controller's free-shot (as described in section 8.1.1).

*Agreed slots counter* → shall count the correct slots received during a TDMA round. The TTP controller shall count its own sending slot as it is also assumed to be correct.

*Failed slots counter* → shall count the failed slots received during a TDMA round. Failed slot means that the receiver either disagrees with the sender on C-state or the reception was invalid.

*Acknowledgment failure counter* → shall count the successive losses of the membership due to negative acknowledgment described in section 8.2. The counter is reset after the TTP controller is positively acknowledged.

*ClockSyncFIFO* → shall maintain a FIFO queue containing four entries for storing time difference measure values for synchronization reasons as described in section 7.2. In this context, FIFO means that the last inserted value will push the oldest value out.

NOTE: Handling and presentation of the protocol status information stored in those variables to the host is optional and depends on the TTP controller implementation.

NOTE:
*) is used to indicate "field available in context of the host interface"

**) is used to indicate "field available in context of the configuration data"

### 9.2.2 Freeze State

Upon entering the *Freeze* state, the TTP controller shall suspend protocol execution, meaning that it is switched off. Only the host shall be allowed to switch the TTP controller back on. The TTP controller shall be able to be switched off either by the host or autonomously by the protocol processor (due to the defined state changes summarized in Table 4). When the host switches the TTP controller off, the TTP controller shall transit from any protocol state into the *Freeze* state. Switching the TTP controller on or off shall be handled via the *TTP Controller On*[*)] field.

It is assumed that all necessary data (e.g., configuration data) to operate the TTP controller is available to the TTP controller before it is switched on. After power-up or reset, the TTP controller shall reside in the *Freeze* state, until it is switched on by the host. When the TTP controller is in *Freeze* state, the host shall be able to read the contents of the host interface in order to determine the cause of a previous error reported by the TTP controller (e.g., before it changed to *Freeze* state). The host shall be able to write to the host interface when it is switched off. This property is used for turning on the TTP controller and for setting up the configuration data. If the host turns the TTP controller on while the TTP controller is in *Freeze* state, the TTP controller shall reset the status area[*)]. The TTP controller may update the status area with TTP controller-specific identification information, e.g., controller version, and protocol version. After completion of the initialization and authentication steps, the TTP controller shall transit into the *Init* state.

TTP controller may provide capability to set up a transition into optional states, before it is switched on. The TTP controller may provide optional asynchronous services (e.g., download, system self-test, etc.) instead of executing the TTP protocol. Those additional states shall not interfere with synchronized TTP network operation.

### 9.2.3 Init State

The TTP controller shall check the availability and shall ensure the integrity of the used set of configuration data (e.g., by performing a CRC check). An error detected in the set of configuration data shall be considered an initialization error*). Furthermore, the occurrence of this error shall cause the TTP controller to transit to *Freeze* state. The TTP controller shall ensure that it is configured with a cluster-specific set of configuration data as listed in Table B3. The TTP controller shall initialize the *cold start counter* with 0. Upon successful completion of all initialization tasks, the TTP controller shall report to the host*) that the initialization is complete, and shall transit into the *Listen* state.

### 9.2.4 Listen State

In the *Listen* state a TTP controller shall verify whether communication is taking place on the bus. If the TTP controller perceives communication, it shall integrate – by adopting the C-state – on received frames.

Therefore the TTP controller shall wait for the duration of $\Delta_{listen}$ to receive a correct frame with explicit C-state (i.e., X-Frame or I-frame) or a cold-start frame – a so-called “suitable frame” to integrate on. If two such frames are received (one on each channel), the TTP controller shall check if both frames carry the same C-state. If the C-state differs in the two frames, the TTP controller shall reject both and re-enter the *Listen* state.

When a suitable frame has been received, the TTP controller shall operate differently based on whether the frame received is a cold start frame or a non-cold start frame with explicit C-state for achieving integration.

The following operations shall be performed when the TTP controller has received not a cold-start frame, but a frame with explicit C-state:

- The TTP controller shall set its local C-state’s*) *Membership vector* and the value of the *DMC* fields to the corresponding values of the received frame.

Independent of the frame type, the TTP controller shall continue with the following operations:

- If the cluster-positioning data – represented by the round slot position and cluster mode field – does not refer to a specified round slot-specific set of configuration data, the frame shall be ignored. Integration shall be attempted based on the next suitable frame.
- The TTP controller shall set its local C-state’s*) *Global time*, *Deferred Pending Mode Change* (DMC) and *Round slot position* fields to the corresponding values of the received frame. In case of a cold start frame, the TTP controller shall verify whether the value of DMC is set to “no request”.
- The TTP controller shall ensure that it is initialized with a round slot specific-set of configuration data (which is defined in Table B3), according to the retrieved cluster positioning data. The TTP controller shall ensure the integrity of the used set of configuration data (e.g., by performing a CRC check). An error detected in the set of configuration data shall be considered an initialization error*) and shall force the TTP controller to transit to *Freeze* state.
- The C-state time transmitted in a suitable frame refers to the action time (as defined in section 7.2). However, the initialization of the receiver’s clock can only be done after reception of the complete frame at $\Delta_{GT\underline{r}}$ according to (EQ. 11) which corresponds to the corrected value for the global time base*).

$$\Delta_{GT\underline{r}} = t_{C-state_s} + \Delta_{corr\,s,r} + \varepsilon + \Delta_t \qquad \text{(EQ. 11)}$$

$t_{C-state_s}$ : Value of the global time field of the C-state.

$\Delta_{corr\,s,r}$ ; $\varepsilon$ : see section 7.2.

$\Delta_t$: The interval from the detection of the frame start to the point in time the synchronization is performed.

- The TTP controller shall wait until the end of the transmission phase**) (end of slot) is reached.

- If there is a mode change request carried in the header of the received frame, the TTP controller shall check if the mode change is permitted**). If the request is not permitted, the TTP controller shall ignore the frame and shall continue waiting for a suitable frame.

- The TTP controller shall check if it has received a suitable frame on the other channel. Then the TTP controller shall check if both frames carry the same C-state. If this check fails, the TTP controller shall ignore both frames, and shall try to integrate on the next received suitable frame.

- The TTP controller shall reset the content of *clockSyncFIFO* to “no deviation”.

The TTP controller shall perform the following actions when it is allowed**) to integrate on cold start frames:

- The TTP controller shall set the membership flag in the membership vector assigned to the sender of the cold start frame and reset the other bits in the C-state*). The position of the membership flag shall be derived from the configuration data and the C-state's *round slot number* in the received frame.

- The *integration counter* shall be set to the threshold value**), meaning that no more correct receptions are needed for acquiring a slot.

- If it is the first received cold start frame, the TTP controller shall ignore it, shall mark that event in its *Big Bang flag*, and shall re-enter the *Listen* state (as described in section 8.1.2.2)

- The TTP controller shall report the availability*) of the complete C-state to the host.

The TTP controller shall perform the following actions when it has synchronized on a non-cold start frame:

- The *integration counter* shall be set to one, meaning that more correct frame receptions**) are (when the TTP controller is in *Passive* state) needed before acquiring a slot.

The TTP controller shall perform the following actions when it has synchronized on a cold start or non-cold start frame:

- The *agreed slots counter* shall be set to two and the *failed slots counter* shall be set to zero (this setting enables an integrating node to send immediately after it has adopted the C-state. See the definitions for clique error and communication blackout in section 8.4).

- The TTP controller shall invalidate (reset) the host life-sign (see section 8.5).

- The TTP controller shall set the frame status of the frame used to achieve synchronization to frame status “correct frame” (cf. section 5.4).

- The TTP controller shall transit into the *Passive* state and shall start schedule-synchronous operation at the respective entry point of the PRP (cf. section 9.2.6 ).

If $\Delta_{listen}$ is elapsed and no suitable frame has been received, a TTP controller shall re-enter the *Listen* state or it shall transit to *Cold start* state when the following criteria are fulfilled:

- The TTP controller is permitted**) to initiate a cold start

- The current value of the *cold start counter* is below the maximum number**) for allowed cold start attempts.

- The host life-sign is maintained accordingly (cf. section 8.5).

9.2.5 Cold Start State

In *Cold start* state a TTP controller shall initiate synchronization after cluster reset or power-up.

The operation sequence is depicted in Figure 30.

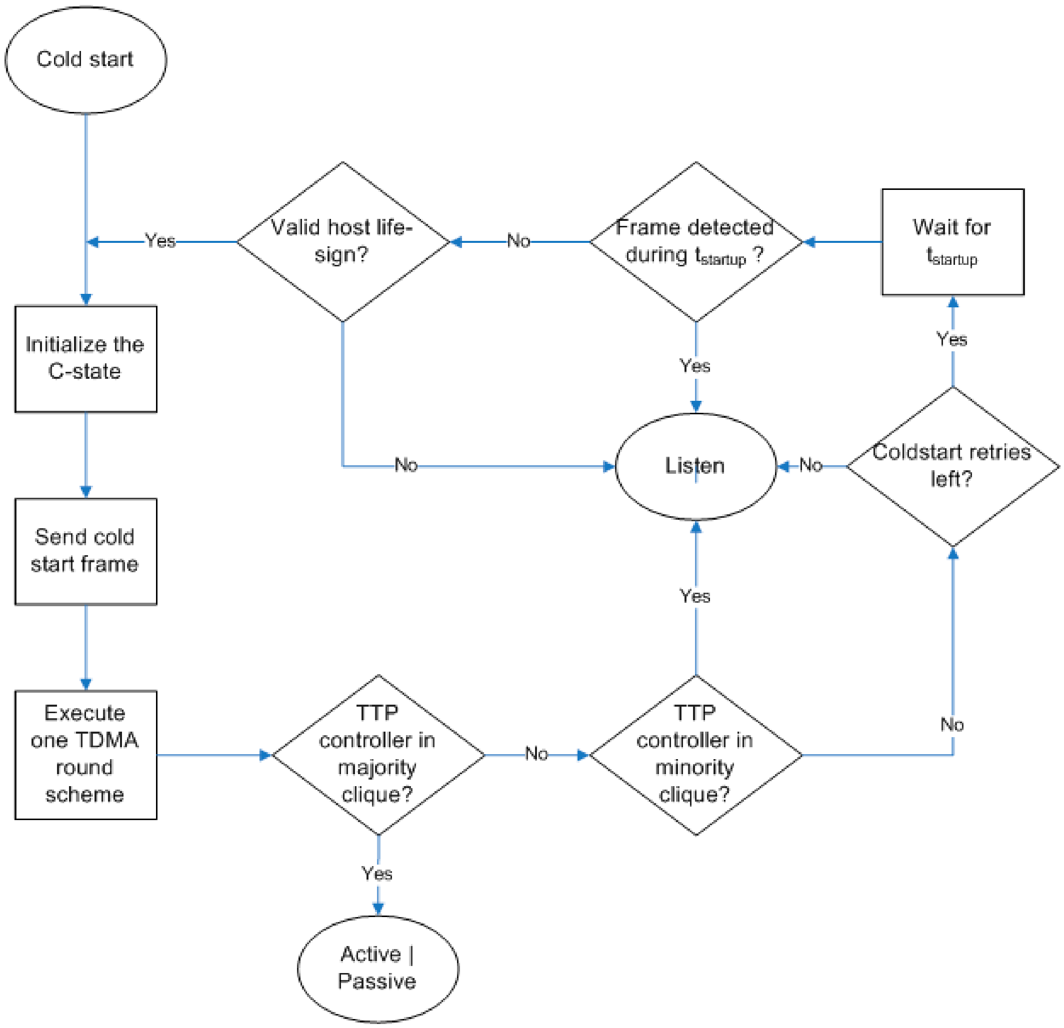


FIGURE 30 - COLD START STATE

The TTP controller shall initialize its local C-state*) in the following way:

- set the global time base in the host interface and the C-state's Global Time to the initial macrotick value which shall be specified by Time Startup Field*).
- Set the C-state's Round Slot Position field to its sending slot position in the first TDMA round.
- Set the C-state's Deferred Pending Mode Change (DMC) field to "no request".
- Set the C-state's Cluster Mode field to "startup mode" described in section 8.6.
- Clear all membership flags and set the node's own flag within the C-state's membership vector field.

Furthermore, the TTP controller shall ensure the integrity of the used set of configuration data (e.g., by performing a CRC check). An error detected in the set of configuration data shall be considered an initialization error*) and shall be reported to the Host and shall force the TTP controller to transit to *Freeze* state.

Since the TTP controller did not detect suitable TTP communication in *Listen* state, the TTP controller shall concurrently send a cold start frame with above defined C-state on every channel.

The TTP controller shall:

- Increment a *cold start counter* whenever it sends a cold start frame.
- Set the protocol variables *failed slots counter* to zero
- disable the free-shot by clearing the *free-shot flag*
- set the *integration counter* to the threshold**) value (meaning that no more correct receptions are needed for slot acquisition)
- set the *clockSyncFIFO* to "no-deviation".

The TTP controller shall start schedule-synchronous operation at the respective entry point of the PRP (see section 9.2.6.3) and shall now execute the TDMA scheme accordingly. Reaching its own sending slot after one TDMA is elapsed, the TTP controller shall perform the clique detection algorithm and check for communication blackout (as described in section 8.4). If the TTP controller is in majority clique, it shall report valid*) C-state information to the host and shall immediately transit to the A*ctive* state or *Passive* state, depending on the status of the host life-sign and the correct request of cluster modes. If a TTP controller is in the minority clique, it shall transit to the *Listen* state. In case of communication blackout, the TTP controller shall be allowed to re-enter the *Cold start* state when it is still permitted to initiate communication**) if the host life-sign has not been updated since the last transmission of a cold start frame. Otherwise the TTP controller shall transit into the *Listen* state. The TTP controller shall re-enter the *Cold start* state when timeout period of $\Delta_{startup\_timeout}$ + remaining duration of the current round slot has elapsed.

### 9.2.6 Active and Passive State

Once the communication system is operating synchronously, the TTP controller shall either be in *Active* or *Passive* state.

In *Passive* state the TTP controller shall receive all frames from other nodes, shall continue to update the host interface of the TTP controller and shall wait until it can acquire its sending slot. While in the *Passive* state a TTP controller shall not send any frames; the membership flag of a passive TTP controller shall be reset.

In *Active* state, the TTP controller shall transmit and receive frames according to the TDMA round-specific definitions stored in the configuration data.

The sequence of synchronous operations is subdivided according to the different timing sections within a node slot as depicted in Figure 4 and 5, the so-called pre-send phase (PSP), transmission period (TP), the post-receive phase (PRP) and the idle phase (Idle).

#### 9.2.6.1 Pre-send Phase

Figure 31 depicts the operation sequence performed in the pre-send phase in each slot. This is done by all nodes in synchronous operation mode, when the TTP controller is in *Active* or *Passive* state. The filled block in Figure 31 is only executed during protocol execution in *Passive* state.

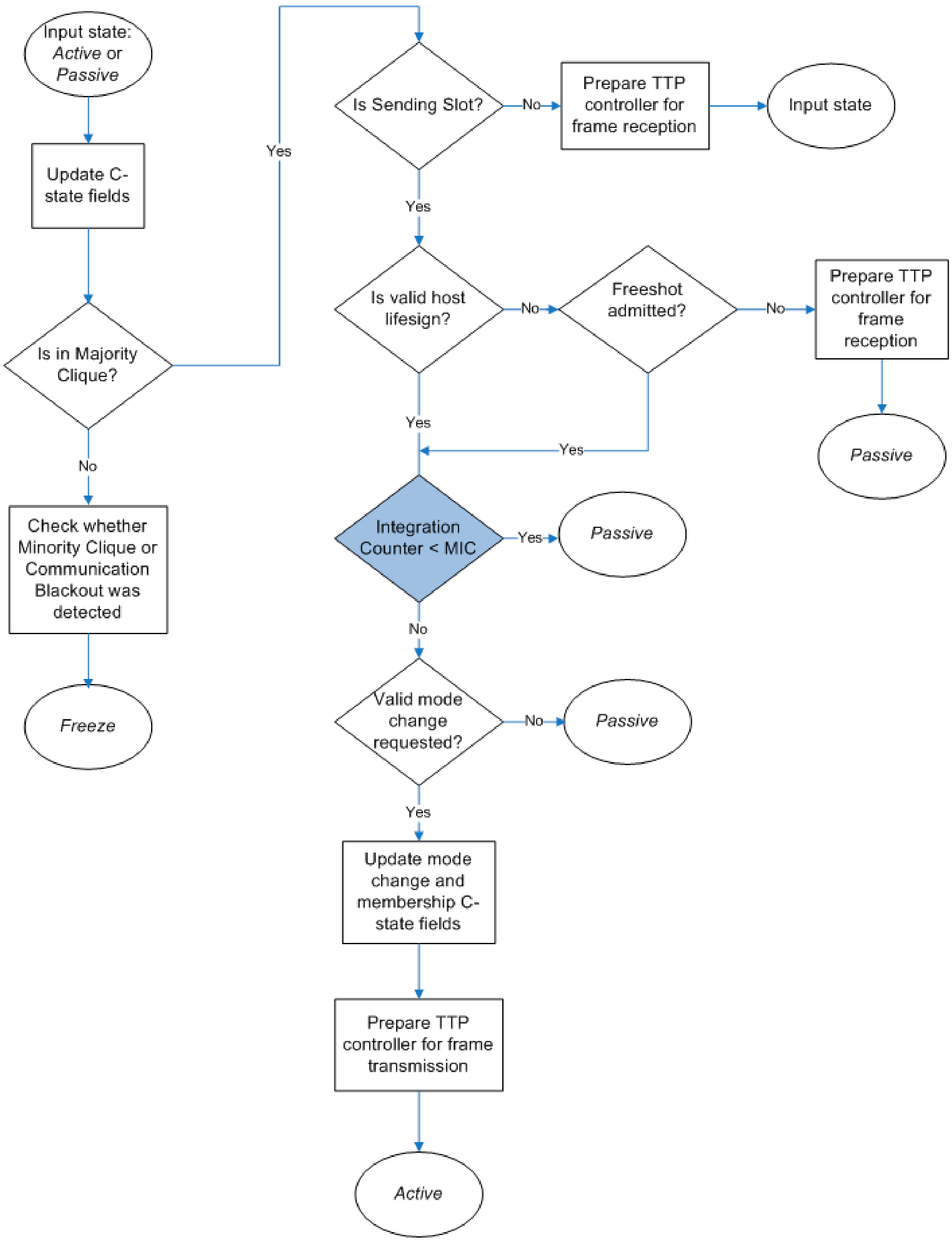


FIGURE 31 - PRE-SEND PHASE

The TTP controller shall update its local C-state*) by incrementing the *round slot position* field. If the value of the round slot position has exceeded the size of the cluster cycle, the round slot position shall be set to the start of the cluster cycle again. If the round slot position points to the first round slot in the cluster cycle, the TTP controller shall check the DMC*) field in the C-state if mode change is pending. If it is, the TTP controller shall set the *cluster mode**) field in the C-state to the requested mode, shall disable the free-shot, and shall set the DMC field to “no request”.

Based on the round slot position and the cluster mode, a TTP controller shall ensure that it is initialized with a round slot-specific set of configuration data (which is defined in Table B3) according to its current slot position. The TTP controller shall ensure the integrity of the used set of configuration data (e.g., by performing a CRC check). An error detected in the set of configuration data shall be treated as an initialization error*) and shall be reported to the host and shall force the TTP controller to transit to *Freeze* state.

The TTP controller shall calculate the value for the upcoming start of the transmission phase (Action time) and shall update the *global time**) in its local C-state.

The TTP controller shall perform clique detection if this is defined by the configuration data for the current round slot. As a result of the clique detection algorithm as described in section 8.4, the TTP controller can perceive itself in the following situations and shall perform actions accordingly:

- Majority clique: The TTP controller shall remain in current TTP controller state (*Active, Passive* state) and shall set the *agreed slot counter* and *failed slot counter* to zero.

- Minority clique or communication blackout: When the TTP controller is in *Active* state or *Passive* state, the TTP controller shall report a clique error*) or a communication blackout*) to the host interface and shall transit to *Freeze* state.

The TTP controller shall check according to the current round slot configuration data, if it shall act as a sender. If the TTP controller is a sender, it shall perform the following actions:

- the TTP controller shall perform the life-sign algorithm as described in section 8.5,

- the TTP controller shall check if the free-shot is enabled as described in section 8.1.1 .

If none of the checks results is true, the TTP controller shall:

- transit to (or stay in) *Passive* state and

- prepare for reception at the next transmission phase

When the TTP controller is in *Passive* state, it shall check if the current value of the *integration counter* (MIC) is below the threshold**). Otherwise it shall transit to or stay in *Active* state.

The TTP controller shall check if the host has requested a cluster mode change and, if applicable, the TTP controller shall check if this mode change is permitted**). If the requested mode change is not allowed, the TTP controller shall report a mode violation error*) and shall transit to *Passive* state.

Otherwise, if the requested mode change is valid (or no mode change was requested), the TTP controller shall:

- transit to or stay in *Active* state,

- set the mode change request field in the frame header, if one is requested,

- set its own membership flag*) in the C-state's membership vector

- prepare to start transmission at the start of the next transmission phase

If the slot is not considered to be a sending slot in the current round slot, the TTP controller shall prepare to start reception at the start of next transmission phase.

9.2.6.2 Transmission Phase

The TTP controller shall start to receive or to transmit a TTP frame at the delayed action time**). In order to do this, the TTP controller shall calculate the actual value for the action time (cf. section 7.2). Furthermore, the TTP controller shall cease to receive or transmit at the end of the transmission phase**). The action time and the duration for the TP shall be defined in the configuration data.

9.2.6.3 Post-receive Phase

Figure 32 depicts the operation sequence performed in the PRP in each slot. This is done by all nodes in synchronous operation mode, when the TTP controller is in *Active* or *Passive* state.

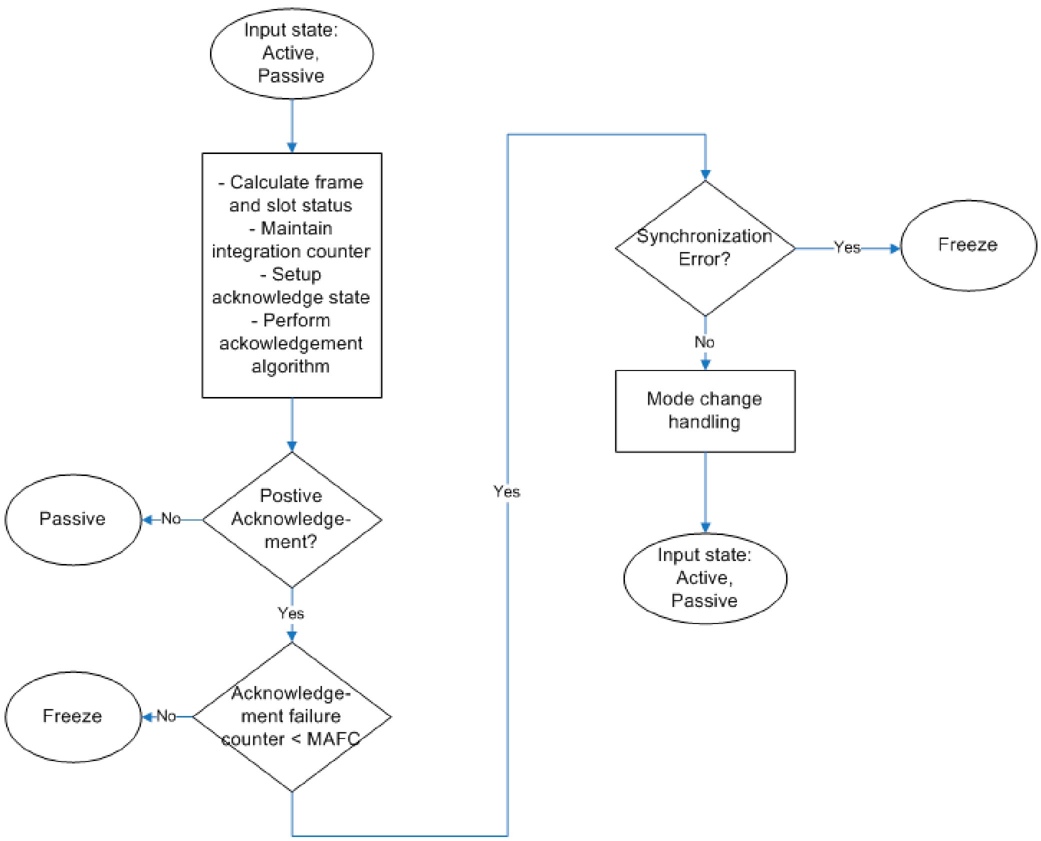


FIGURE 32 - POST-RECEIVE PHASE

The TTP controller shall calculate the frame status for received and sent frames for both channels. This shall be done for received frames according to the criteria described in section 5.4 and for transmitted frames. The TTP controller shall set the frame status to “correct frame” if the transmission was completed successfully within the transmission phase. "Successfully" means without detection by the coding layer or the physical layer.

The slot status shall be calculated from the frame status according to section 5.4.8. In case of a received frame, the TTP controller shall update the *agreed slot counter* and the *failed slot counter* as described in section 5.4.8. In the case of a sent frame, the *agreed slots counter* shall be set to 1. The *integration counter* shall be incremented by one in every slot whose status evaluates to “correct” until the counter reaches the threshold value defined in the configuration data. If the *integration counter* reaches this threshold, the TTP controller shall report this to the host.

Based on the slot status, the TTP controller shall retrieve membership and acknowledgement status respectively.

In case of a reception slot, the acknowledge algorithm shall update the membership vector in its local C-state as described in section 8.2. A sending node shall perform the acknowledgement mechanism to detect whether its frame transmission was successfully received by the successor nodes. If the algorithm results in a positive acknowledgement from the first or second successor, the *acknowledgement failure counter* shall be set to zero. If the node loses membership due to a negative acknowledgement from the second successor, the *acknowledgement failure counter* shall be incremented by one, and the TTP controller shall transit to *Passive* state. When the *acknowledgement failure counter* reaches the threshold[**)], the TTP controller shall report this error in the host interface and shall transit to *Freeze* state.

The TTP controller shall perform the synchronization algorithm according to the requirements described in Chapter7. In the case of a synchronization error[*)], the TTP controller shall report this to the host and shall transit to *Freeze* state.

The TTP controller shall evaluate mode change requests as follows. The controller shall select a correct frame for mode change request evaluation with channel 0 having precedence over channel 1. Note that frame status “correct” implies that the respective frame carries a valid mode change request. However, requests for different follow-up modes may only be allowed at a particular time (see section 8.6). Consequently, the frames received on the two channels during the same slot may both be correct despite carrying conflicting mode change requests. In this case, the rule above guarantees that all receiver nodes will consistently decide on the mode change request to be accepted. Note further, that a correctly transmitted frame is treated just as a correct frame received. If an allowed mode change was requested in the received frame, the *DMC* field of the C-state shall be updated accordingly. For example, if a “clear pending request” is requested in the selected frame, the *DMC* field in the local C-state shall be set to “no request”.

#### 9.2.6.4 Idle Phase

The TTP controller shall execute the idle phase by waiting for the start of the next pre-send phase.

APPENDIX A – GLOSSARY ACRONYMS

| | |
|---|---|
| **C-state** | Controller state |
| **CO** | Controller on flag |
| **CRC** | Cyclic redundancy check |
| **DMC** | Deferred pending mode change |
| **ID** | Identifier |
| **IFG** | Interframe gap |
| **MCP** | Mode change permissions |
| **MIC** | Minimum integration count |
| **PRP** | Post-receive phase |
| **PSP** | Pre-send phase |
| **SAE** | Society of Automotive Engineers |
| **TDMA** | Time division multiple access |
| **TP** | Transmission phase |
| **TTA** | Time-triggered architecture |
| **TTP** | Time-triggered protocol |

# APPENDIX B - COMPATIBILITY REQUIREMENTS FOR EXISTING TTP CONTROLLER IMPLEMENTATION

## B.1 PERFORMANCE & CONFIGURATION COMPATIBILITY REQUIREMENTS

Table B1 presents performance requirements that shall be fulfilled or configurable by any TTP controller implementation.

TABLE B1 – MINIMUM REQUIRED PARAMETER VALUE RANGE

| Description | Value range | Reference |
|---|---|---|
| Data size of TTP frame (maximum value) | 240 Byte | Chapter 5 |
| Duration range for a macrotick | $0.5\mu s \leq \Delta_{MT} \leq 25.6\mu s$ | Chapter 7 |
| Media propagation delay configuration (maximum value) | 25.6µs | Chapter 7 |
| Precision interval | $\Pi \leq 12.75\mu s$ | Chapter 7 |
| PRP / PSP | 494 microticks / 440 microticks | Chapter 5 |
| 24-Bit Frame CRC Generator Polynomial (which guarantees a Hamming distance of 4 for all data with a length of 2024 bits (253 bytes) | 0x8B4BC7 | Chapter 5 |
| DMC value range (Possible values of successor modes) | 0x0 → 'no request'<br>0x1 → First successor mode<br>1x0 → Second successor mode<br>1x1 → Third successor mode | Section 5.2.2 and section 8.6 |
| Mode change request value range | 0x0 →Field cleared, no mode change request.<br>0x1 →First successor mode requested.<br>0x2 →Second successor mode requested.<br>0x3 →Third successor mode requested.<br>0x4 →Clear Pending Mode Change (CPM) request.<br>0x5-0x7 Invalid, will cause a mode violation host interrupt. | Section 5.3 and section 8.6 |

## B.2 TYPICAL COMMUNICATION PARAMETER VALUE RANGES

Error! Reference source not found.B2 provides an overview of typical TTP communication parameters.

TABLE B2 – TYPICAL PARAMETER VALUE RANGES

| Description | Value range | Reference |
|---|---|---|
| Relationship between precision interval and macrotick | $\Pi < \Delta_{MT}$ | Chapter 7 |
| Maximum allowed value of the clock state correction term | $\lvert \max CSCT \rvert \leq \Pi / 2$ | Chapter 7 |
| Relationship between start window and precision interval | $\varepsilon = \Pi$ | Chapter 7 |
| Macrotick | 1 … 10µs | Chapter 7 |
| Macrotick / Microtick ratio *) | 200 | Chapter 7 |
| Minimum integration count 2 | 2 | Section 8.1.1 |

*) Assumption: Microtick = 25ns; Macrotick = 5µs

**) Assumption: Propagation delay = 100m; Precision = 5µs

TABLE B3 – CONFIGURATION PARAMETER – SUMMARY

| Configuration parameter | Description | Definition-Level | Reference |
|---|---|---|---|
| Send delay | Defines the time span $\Delta_{delays}$ between the scheduled and the actual value for the action time at the senders side | Channel | Section 7.2 |
| Action time | Defines the start of the TP in units of MT | Round-slot | Section 5.1 |
| Master node | Defines whether the time difference value $\Delta_{dif}$, measured in the current round slot, is used for the clock correction. | Round-slot | Section 7.2 |
| Perform clique avoidance | Defines whether the clique avoidance algorithm is performed. | Round slot | Section 8.4 |
| Resynchronization interval | Defines whether the clock synchronization algorithm is executed during the PRP of the current round slot. | Round-slot | Section 7.2 |
| Sending Slot | Denotes a node's own sending slot. | Round-slot | Section 8.1.1 |
| Slot duration | Defines the duration of the current round slot in units of macroticks. | Round-slot | Section 5.1 |

| | | | |
|---|---|---|---|
| Successor Modes | Defines whether a requested cluster mode change is allowed in the current round slot | Round-slot | Section 8.6 |
| Transmission Phase | Defines the duration of the TP in units of MT | Round-slot | Section 5.1 and Section 7.2 |
| TP Duration | Defines the duration of the TP in units of MT | Round-slot | Section 5.1 |
| Application data length | These fields represent the lengths of the application data contained in the frame. | Round-slot & Channel | Chapter 5 |
| Delay correction term | Defines the time span between the scheduled and the expected value of the action at the receiver's side. The value of the parameter depends on physical layer and topology specific properties, defined by $\Delta_{corr\,s,r}$ and $\Delta_{prop\,s,r}$ | Round slot & Channel | Section 7.2 |
| Frame Type | Defines whether a frame with explicit C-state is transmitted on the in the current slot on this channel. | Round-slot & Channel | Section 5.3 |
| Cold start allowed | Defines whether the TTP controller can enter the cold start state. | Cluster | Section 8.1 |
| Cold start Integration allowed | Defines whether a TTP controller integrates a cold start frame. | Cluster | Section 8.1.1 |
| Frame CRC - Initialization value | Used as seed values for the 24-bit bus CRCs. | Cluster | Section B.1 |
| Free-Running Macroticks | Specifies the number of macroticks remaining unchanged between any two MTs which are modified by the clock synchronization mechanism. | Cluster | Section 7.2 |
| Listen timeout | Defines a unique timeout to to check for communication during integration. | Cluster | Section 8.1 |
| Maximum acknowledgment failure count | Specifies the number of successive acknowledgment failures at which the TTP controller raises the acknowledgment failed protocol error. | Cluster | Section 8.3 |
| Maximum number of allowed cold starts | Specifies the maximum number of cold start attempts by the TTP controller. | Cluster | Section 8.1 |
| Membership (Own Flag) | Defines the flag position (zero-based) in the membership vector used by this node. | Cluster | Section 8.2 |
| Microtick / Macrotick ratio | Defines the cluster-wide macrotick duration in relation to the controller's local microtick clock. | Cluster | Section 7.1.2 |

| | | | |
|---|---|---|---|
| Minimum integration count | To ensure that controllers do not integrate on faulty frames a specific number of correct frames has to be received before a node can transmit actively. | Cluster | Section 8.1 |
| Precision | Influences the detection of synchronization errors of the TTP controller. | Cluster | Section 7.1.3 |
| Receive window | This parameter defines the size of the symmetric start window. | Cluster | Section 7.2 |
| Round Slots Number | Holds the number of round slots in this mode. | Cluster | Section 5.1 |
| CRC seed value for channel 0 | 24 Bit CRC Initialization value for channel 0 | Cluster | Chapter 5 |
| CRC seed value for channel 1 | 24 Bit CRC Initialization value for channel 1 | Cluster | Chapter 5 |
| Startup timeout | Defines a unique time out value to avoid multiple startup collisions, in units of MT | Cluster | Section 8.1 |