TTTech

tta Group

TTP

Time-Triggered Protocol TTP/C High-Level Specification Document

Specification edition 1.0.0 of 4-July-2002
Document number D-032-S-10-028

This document was issued by TTTech to CSC, Defense Group.

TTP As predictable as Time

TTA-Group
Schoenbrunner Strasse 7
A–1040 Vienna
Austria

Tel: +43 1 585 34 34–0
Fax: +43 1 585 34 34–90
Http: www.ttagroup.org
Email: info@ttagroup.org

Contents

List of Tables

List of Figures

1 Preface

1.1 Objective of this Document

This document specifies the structure of the TTP/C protocol. It describes the services and mechanisms on an abstract level without implementation specific details.

The Time-Triggered Protocol meets the requirements for safety critical distributed real-time systems in several application domains, e.g., automotive, aerospace, and industrial control electronics; additionally, several mechanisms for specific applications and configurations are defined in the protocol specification, which are not required for interoperability of basic TTP/C configurations, but constitute options that are available for implementations targeting specific fields of use.

Some methods contained in the document are protected by granted and pending patents.

1.2 History

The TTP/C protocol is the result of intensive discussions among potential users of the TTP technology. Compared to the first published TTP protocol [Kop94b], the current TTP/C protocol incorporates changes in the bus access, the acknowledgment scheme and the redundancy management. The first specified [Kop97b] and implemented TTP/C protocol [Ang98] including all services described in this specification is dated back to the year 1997.

We especially would like to thank the companies involved in the formation of TTA-Group, including Audi/VW, Honeywell, Delphi Inc., PSA-Peugeot Citroën, Renault and Airbus for their valuable support.

2 Scope

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/C was originally intended to meet the requirements of SAE class C automotive applications.

The current protocol specification is targeted at distributed real-time systems with strong requirements for safety, availability, and composability in the fields of automotive and aerospace electronics as well as industrial control.

2.1 The Time-Triggered Architecture

A computer architecture establishes a blueprint and a framework for the design of a class of computing systems that share a common set of characteristics. The *Time-Triggered Architecture (TTA)* generates such a framework for the domain of distributed embedded real-time systems in high-dependability environments. It sets up the computing infrastructure for the implementation of applications and provides mechanisms and guidelines to partition a large application into nearly autonomous subsystems along small and well-defined interfaces in order to control the complexity of the evolving artifact. Architecture design is thus interface design. By defining an architectural style that is observed at all component interfaces, the architecture avoids property mismatches at the interfaces and eliminates the need for unproductive 'glue' code. A central characteristic of the Time-Triggered Architecture is the treatment of (physical) real time as a first-order quantity. The TTA decomposes a large embedded application into clusters and nodes and provides a fault-tolerant global time base of known precision at every node. The TTA takes advantage of the availability of this global time to precisely specify the interfaces among the nodes, to simplify communication, to establish state consistency, to perform prompt error detection, and to support the timeliness of real-time applications.

2.1.1 Structure of the TTA

The basic building block of the TTA is a *node* (see figure 2.1 on the following page). A node comprises a processor with memory, an input-output subsystem, a time-triggered communication controller, an operating system and the relevant application software in a self-contained unit (possibly on a single silicon die). Two replicated communication channels connect the nodes to build a *cluster*. The physical interconnection structure and the communication controllers of all nodes of the cluster form the communication subsystem. In the TTA, the communication subsystem is autonomous and executes an a priori specified periodic TDMA (time-division multiple access) schedule. The communication subsystem reads a data frame containing state information from the communication network interface (CNI) at the sending node at the a priori known fetch instant and delivers it to the CNIs of all receiving nodes of the cluster at the a priori known delivery instant, overwriting the previous version of this frame. The periodic fetch and delivery instants are contained in a scheduling table within the communication controller consistently known to all communication controllers in a cluster.

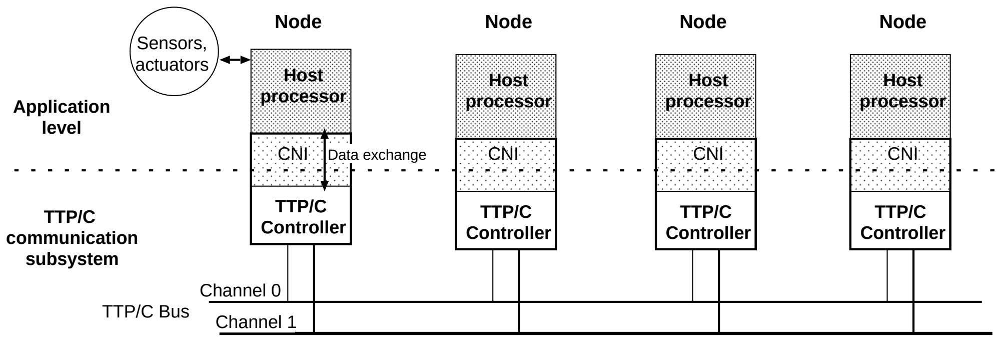

Figure 2.1: TTA Cluster

2.1.2 Design Principles

The following paragraphs shortly discuss the principles that have driven the design of the Time-Triggered Architecture.

Consistent Distributed Computing Platform The main purpose of the TTA is to provide a consistent distributed computing base to all correct nodes in order that reliable distributed applications can be built with manageable effort. If a node cannot be certain that every other correct node works on exactly the same state, then the design of distributed algorithms becomes very cumbersome [Pea82] because the intricate agreement problem has to be solved at the application level.

Unification of Interfaces – Temporal Firewalls A good architecture must be based on a small number of orthogonal concepts that are reused in many different situations thus reducing the effort required for understanding complex systems. TTP/C carries autonomously - driven by its time-triggered schedule - data frames from the sender's communication network interface (CNI) to the receiver's CNI. The sender can deposit the information into its local CNI memory according to the information push paradigm, while the receiver must pull the information out of its local CNI memory. Since no control signals cross the CNI in the TTA (the communication system derives control signals for the fetch and delivery instants from the progress of global time and its local schedule table), control-error propagation is eliminated by design. We call an interface that prevents control-error propagation by design a temporal firewall.

Composability In a distributed real-time system the nodes interact via the communication system to provide the emerging real-time services. These emerging services depend on the timely provision of the real-time information at the interfaces of the nodes. We define an architecture to be composable in the temporal domain, if it guarantees the following four principles [Kop97a]:

1. Independent development of nodes
2. Stability of prior services
3. Constructive integration of the nodes to generate the emerging services
4. Replica determinism

Scalability The TTA is intended for the design of complex distributed real-time applications. A complex system can only be constructed if the effort required to understand a particular system function is independent of the system size. Horizontal layering (abstraction) and vertical layering (partitioning) are the means to combat the complexity of large systems. In the TTA, the CNIs encapsulate a function and make only those properties of the environment visible to the function that are relevant for its operation.

Transparent Implementation of Fault Tolerance In the TTA the fault-tolerance mechanisms are implemented in a dedicated fault-tolerance layer, possibly with its own middleware processor, such that the fault-tolerant CNI (FT-COM CNI) is identical in structure and timing to the non-fault-tolerant CNI. A properly structured application software [Kop97a] can thus operate in a fault-tolerant system or a non-fault-tolerant system without any modifications. The fault-tolerance mechanisms remain transparent to the application in the TTA.

2.2 Consistency

Design and development of complex distributed systems can be eased considerably by supporting state consistency. In single node systems consistency can be taken for granted since data written to memory becomes available to all software subsystems at the same time and all subsystems read the same value, given the node is correct. In distributed systems it is no longer justified to assume this kind of consistency. There are two reasons for this: Firstly, message transmission delays that have an effect on the current state must be considered; it is not guaranteed that a message arrives at all receivers at the same point in time. Secondly, individual nodes may fail or messages may get lost.

In abstract system theory, the notion of state is introduced in order to separate the past from the future [Mes89]:

> ”The state enables the determination of a future output solely on the basis of the future input and the state the system is in. In other words, the state enables a ‘decoupling’ of the past from the present and future. The state embodies all past history of a system. Knowing the state ‘supplants’ knowledge of the past. Apparently, for this role to be meaningful, the notion of past and future must be relevant for the system considered.”

Taking this view it follows that the notions of state and time are inseparable. If an event that updates the state cannot be said to coincide with a well-defined tick of a global clock on a sparse time-base, then the notion of a system-wide state becomes diffuse. It is not known whether the state of the system at a given clock tick includes this event or not. The sparse time-base [Kop97a] of the time-triggered architecture makes it possible to define a system-wide notion of time, which is a prerequisite for an indisputable borderline between the past and the future, and thus the definition of a consistent system-wide state.

Application design and programming in a distributed system becomes extremely cumbersome, if not unmanageable at all for complex applications, if state consistency is not supported at the architecture level. The huge number of different possible scenarios of node failures, communication failures or differences in message arrival timing or sequence would make the application subsystem logic extremely large and complex. Instead, state consistency needs to be supported as a basic service of the computing platform satisfying the following properties. Let us assume a node sends a message to a set of receiver nodes. The system is now called consistent if (1) all correct nodes agree on the same value, (2) given the sender is correct all nodes agree on the sent value and (3) all correct subsystems deliver the received value at the same point in time. TTA took the route to implement consistency directly at the protocol level in hardware in order to relief the application CPU from executing consistency protocols and more important, to verify the correctness of these algorithm at the hardware level once for all applications. The mechanisms supporting consistency are described in the following.

2.3 Characteristics of TTP/C

The design of the TTP/C protocol has been guided by the following principles:

Time-Triggered Bus Access In TTP/C communication is organized according to a static TDMA schedule: each node has a sender slot and knows the points-in-time when to send and to receive. Due to this access strategy bus collisions are avoided, the transmission delays are known and the jitter minimal.

Temporal Firewall The communication network interface between the host computer and the controller rules out a control error propagation by design. Because the properties of this interface are fully specified in the temporal domain, TTP/C systems are *composable* from the point of view of their temporal characteristics.

Fault-Tolerant Global Time Base The TTP/C controllers process a fault-tolerant clock synchronization that establishes a sparse global time base without relying on a central time server. The time base with a precision in the microsecond range is provided to all host computers.

Consistent Membership Service The TTP/C controller informs its host computer about the state of every other computer in the cluster with a latency of less than two TDMA rounds[1]. The membership service employs a distributed agreement algo-

[1]For multiplexed nodes, the latency becomes larger.

rithm to determine, in case of a failure, whether the outgoing link of the sender or the incoming link of the receiver has failed.

Single Hardware Failure *Any* single hardware failure in a properly configured TTP/C system must be tolerated.

Malicious Software A *maliciously faulty* host computer (including its software) can produce erroneous data outputs, but can never interfere in any other way with the correct operation of the rest of a TTP/C system. Fail-silence behavior in the temporal domain is in the scope of the controller, while fail-silence in the value domain is in the scope of the host.

Clique Avoidance The clique avoidance is used to detect faults outside the fault hypothesis which cannot be tolerated at the protocol level.

Confidence Principle A TTP/C controller will always assume that it is operating properly unless protocol mechanisms explicitly state otherwise. In this way it is assured that upon a disagreement between nodes the node that finally wins has never made a wrong decision.

2.4 Structure of the Document

This specification document is self-contained and gives a bottom-up approach to the functionality of the TTP/C protocol. It is divided into the following chapters:

Glossary Description of TTP specific terms and abbreviations.

Architecture Overview of the TTP/C protocol layers and the underlying fault hypothesis.

Media Access Description of the TTP/C media access scheme.

Physical Layer Characteristics of the physical layer.

Data Link Layer Characteristics of the data link layer.

Protocol Service Layer Description of the protocol services.

MEDL Characteristics of the message descriptor list.

CNI Characteristics of the communication network interface.

Protocol States Description of the TTP/C protocol operation.

An overview of all related documents is given in chapter A on page 113.

3 Glossary of Terms and Acronyms

3.1 Glossary of Terms

Action Time The point in time at which the controller starts frame transmission (in its own sending slot) or expects frame transmission from another controller. The action time is raised synchronously on all controllers within the precision.

Agreed Slots Counter A counter that counts in each TDMA round the number of nodes that have sent at least one correct frame. It is reset in the own slot.

Application Identification The name of the personalized schedule description (MEDL) that is loaded into the controller, as defined by the cluster designer. It has no influence on protocol execution.

Await State If the host processor decides that the node needs a software update, the controller does not perform TTP/C communication and instead waits for download in the await state.

Blackout Temporary interference of the TTP/C system's operation by some powerful external disturbance, causing correlated failure of a set of nodes. Not the same as Communication Blackout.

Built-in Self Test Flags Flags indicating that the controller hardware has a transient or permanent fault.

Built-in Self Test Interrupt This interrupt is raised whenever the controller finishes a self test requested by the host, or when it detects an internal hardware fault. It subsequently transits into the freeze state.

Bus In TTP/C the bus consists of two replicated communication channels and interconnects all nodes of a TTP/C cluster. The term 'bus' here denotes all possible topologies — including bus, star, and ring architectures — unless explicitly noted otherwise.

Bus Guardian An independent unit that protects the bus from timing failures of the controller. Can be part of the controller silicon or an external unit.

Byzantine Failure In a multiple receiver scenario, the different receivers see differing, possibly incorrect, results.

Cluster Mode See 'mode' (section 3.1 on page 11).

Controller State (C-state) The internal state of the controller, consisting of the current time, current position in the MEDL, the current cluster mode, a pending deferred mode change (DMC) and the current membership.

C-state MEDL Field The C-state MEDL field contains the current cluster mode, the current position in the MEDL and a possible pending deferred mode change.

C-state Membership The C-state membership contains the current membership vector of the cluster.

C-state Time Field The C-state time field contains the synchronized global time in node slot granularity.

C-state Valid Interrupt This interrupt informs the host that the C-state field in the CNI holds valid data. This indicates that the controller is now synchronized with the cluster (or rather the active nodes in the cluster) and from now on performs communication as defined in the MEDL according to the current cluster mode.

Channel The physical communication channel. In TTP/C there are always two replicated communication channels in a cluster. The channels are called channel 0 (Ch0, alternatively named channel A, ChA) and channel 1 (Ch1, alternatively named channel B, ChB).

Class C Control applications that are safety critical (e.g., anti lock brakes), used by the SAE.

Clock State Correction Term The current value of the correction term computed by the clock synchronization algorithm, in units of microticks.

Cluster The set of nodes sharing a bus in a TTP/C system.

Cluster Cycle The sequence of different TDMA rounds. Each cluster mode may have cluster cycles of different length.

Cluster Time Field This field contains the synchronized global time with a granularity of one macrotick.

Cold Start When the cluster is powered up, the first transmission of a cold start frame in the cold start state is asynchronous, as no global timebase has yet been established. Cold start is controlled by several well-defined timeouts in order to prevent multiple collisions between cold-starting nodes and to guarantee a worst-case startup time of a cluster.

Cold Start Counter Holds the number of cold starts already performed by this node; this number is limited by the Maximum Cold Start Entry.

Communication Blackout Protocol error raised by the controller if no traffic was observed on any channel during one TDMA round.

Communication Network Interface (CNI) The interface between a TTP/C controller and the host computer.

Controller Await Flag (CA) This flag is set by the host to force the controller into the await state.

Controller Life-sign The host can use the contents of this field to check whether the controller is still alive. The controller must update it periodically.

Controller On Flag (CO) This field is used by the host to switch the controller on or off. Additionally it informs the host whether the controller is operational or whether it has turned itself off.

Controller State (C-state) see C-state.

Controller Version Number The version number of the controller implementation.

Correct Frame A valid frame which passed the CRC check and all additional semantic checks at the receiver.

CRC Cyclic redundancy check.

Current Logical Name The name that determines the role of a node in a cluster. The logical name can be changed by reconfiguration (if a controller supports this feature)

Deferred Mode Change A mode change that is deferred until the beginning of the next cluster cycle.

Deferred Mode Change Field (DMC) A field in the C-state that stores pending deferred mode changes.

Delay Correction Term A time interval contained in the MEDL that denotes the expected delay of signal propagation between two nodes.

Download TTP/C download is a dedicated communication protocol apart from normal TTP/C communication; it serves the purpose of updating MEDLs and/or application data in a node from a central maintenance node (the download master node). Download is a safety critical functionality, as it can change MEDLs in the controllers. The download protocol is a point-to-point protocol that does not utilize the TDMA strategy and cannot be executed simultaneously with normal TTP/C communication.

Electronic Module An electronic module is an electronic control unit in a vehicle. (An electronic module that is connected to a TTP/C bus and has the structure shown in figure 4.3 on page 20 is called a 'node'.

External Rate Correction Field A field in the CNI that contains the external rate correction term used for external clock synchronization, e.g. to a GPS receiver. The contents of this field are updated by the host application; the controller adds the contents of this field to the current clock state correction term. Many applications do not need the service of external clock correction, and will not use this field.

Fail-Silent A node is called fail silent, if it either

- operates correctly by sending correct (both in value and time domain) frames or
- sends frames which all receivers can reliably detect as incorrect (e.g. by means of a checksum)
- sends no frames at all.

Thus a fault in a fail-silent node is detectable by all receivers without additional requirements, like TMR voting.

Failed Slots Counter A counter that counts in each TDMA round the number of nodes sending at least one failed frame but no correct frame. This counter is reset in the node's own slot, after the clique avoidance has been performed.

Failed Frame An invalid or incorrect frame.

Frame A frame is one complete transmission of information on a communication channel. A frame is delimited by two interframe gaps.

Frame Status Result of the syntax evaluation of a received frame and the C-state agreement check.

Free-Running Macroticks Number of unchanged macroticks between two modified macroticks for the purpose of clock adjustment.

FT-COM Layer Fault-Tolerance communication layer used for redundancy management of messages by the host application.

H-State The h-state encompasses all information that is required to start an ‘empty’ node (or task) at a given point in time.

Host The computer within a node that executes the application software.

Host Error An error in the host.

Host Error Flags Flags that denote the type of the host error detected by the controller (e.g., the host has violated the CNI access timing). The controller also generates a host error interrupt and transits into the passive state.

Host Error Interrupt An interrupt generated by the controller whenever it judges the host to be in error.

Host Life-sign A field in the CNI that must be periodically updated by the host (at least once between two transmissions of its controller) to show host activity.

Idle Phase The MEDL may contain slot durations that specify an inter-frame gap longer than required by the protocol (the sum of post-receive phase and pre-send phase). In this case, the controller will idle between the transmission phase and the post-receive phase. This decreases data throughput, but can be desirable in order to set up application related round durations.

Implicit Acknowledgment Acknowledgment of the receipt of a frame by the successor of the sender is implicit by the node membership.

Incorrect Frame A syntactical valid frame (coding and size correct) for which all CRC checks have failed at the receiver.

Inter-frame Gap (IFG) The time interval required by the controller to execute the protocol tasks following a transmission phase and preceding a frame transmission. See also post-receive phase, idle phase, pre-send phase.

Interrupt Line A signal from the controller to the host computer that is used to inform the host computer about interrupt conditions within the controller and about host errors detected by the controller, and to provide a global synchronized time signal. The TTP/C controller offers several interrupt conditions to be selectively enabled or disabled by the host.

Interrupt Status Field This field contains information on the condition(s) that triggered an interrupt from the controller to the host computer.

Invalid Frame A frame that is syntactical invalid, i.e., coding rules (e.g., coding or expected length) are violated.

Macrotick A periodic signal that delimits a granule of the global time.

Maximum Cold Start Entry A value contained in the MEDL that denotes the maximum number of allowed cold starts of a controller.

Maximum Membership Failure Count (MMFC) A value contained in the MEDL that denotes the maximum number of successive membership failures before a node has to terminate its operation. The MMFC is the upper limit for the membership failure counter.

Measured Time Difference The time difference between the expected arrival time and the actual arrival time of a frame on a channel.

MEDL See Message Descriptor List.

Member node A member node is a real member node (a non-multiplexed node) or a virtual member node (a set of multiplexed nodes). Each member node is assigned to one unique node slot of a TDMA round.

Membership The information about which nodes in the cluster are currently operational, based on correct transmission activity.

Membership Failure The event when an operational node fails, as judged by a majority of nodes in a cluster.

Membership Failure Counter (MFC) A counter used to count the number of successive membership failures. A controller will assume its own failure if the MFC reaches the value of the maximum membership failure count.

Membership Point Point in time when the membership is established. It is in the PRP after a node was supposed to send a frame.

Membership Recognition Point The membership recognition point is the point in time when an observing node makes a final decision about the membership of a sending node. In a failure scenario, the membership recognition points at different observing nodes may not be the same.

Membership Vector A vector that has a unique flag assigned to each member node. If this flag is set, the member node was operational at its last membership recognition point, otherwise it was not operational.

Message Data whose semantics is only known by the host is called message. Messages are transmitted in the application data part of a frame. The frames sent on the two channels in one slot by one sender may or may not contain identical messages. In fault tolerant systems, the channels are typically used for fully redundant transmission.

Message Descriptor List (MEDL) In an abstract form, the complete communication design of the TTP/C cluster. In a personalized form, the data structure in the controller that contains the control data for the controller. The contents of the MEDL determine when a particular frame has to be sent or received or what mode change is permitted at any given point in time.

Microtick A periodic signal that is generated by the oscillator of the controller. Each macrotick is made up of a number of microticks.

Minimum Integration Count Number of correct slots that a node must receive before it is allowed to send (if integrated on a non-cold start frame).

Mode The term ‘mode’ (also known as ‘cluster mode’) denotes application-defined functional behavior of the distributed system; for the communication system, a specific cluster mode is associated with a specific set of data that needs to be transmitted between the nodes, and different modes (if a system design has them) may require quite different sets of data, for example for ‘application mode’, ‘diagnosis and maintenance mode’, and ‘emergency mode’.

The TTP/C protocol requires one dedicated cluster mode – the startup mode – to establish synchronized communication among the nodes. Each MEDL contains at least two modes, the startup mode and one application mode; a large number of cluster modes can be defined in a single cluster design.

The current cluster mode and pending mode changes are part of the C-state and therefore subjected to the continuous state agreement performed by all controllers.

Mode Change A transition from one cluster mode to another cluster mode. A TTP/C mode change always affects the cluster as a whole, even if some nodes may not have to change their internal mode of operation (from the view of the host application).

Mode Change Permissions (MCP) A field in the MEDL round slot entries that contains the information about the permitted mode change requests in each slot.

Mode Change Request Field The control field in the CNI that informs the controller about the mode change request from the host.

Multiplexed Node A node that shares a node slot with one or more nodes. A multiplexed node role is statically assigned to a specified node slot in a specified TDMA round for reintegration. With multiplexing, the number of physical nodes in a cluster can exceed the number of TDMA slots in a round.

Node An electronic module that is connected to a TTP/C network and that has the structure of figure 4.3 on page 20.

Null-Frame No activity is observed on any of the channels during a node slot.

Post-Receive Phase (PRP) The time interval that the controller requires to process the frames received in the previous transmission phase, and to perform the appropriate updates in the CNI. The phase after the own transmission of a controller is also called PRP. The duration of the PRP is an implementation specific parameter of a controller design. The PRP is part of the IFG.

Pre-Send Phase (PSP) The time interval that the controller requires to prepare to send in the next transmission phase. The time interval to prepare a reception is also called PSP. The duration of the PSP is an implementation specific parameter of a controller design. The PSP is part of the IFG.

Precision (Π) Maximum interval between any two corresponding ticks of the synchronized clocks of a global timebase. The precision is defined by the cluster design and continuously checked by the controllers; if a controller detects that its clock cannot achieve this level of synchronization anymore, it raises a synchronization protocol error and terminates operation.

Protocol Error An error detected by the checking mechanisms of the protocol that makes further execution of the protocol impossible. The occurrence of a protocol error forces the controller into a halt – it enters the freeze state.

Protocol Error Interrupt Interrupt raised whenever a protocol error occurs.

Protocol State The TTP/C protocol controller is always in a defined state of operation. The protocol specification describes the state machine for the execution of the protocol. This field contains the number of the state the controller currently is in.

Reconfiguration Advanced protocol feature, which is not described in this specification. Allows the host to specify and change the node role during protocol operation.

Real Member Node An operational node that does not share a node slot with another node.

Resynchronization Interval The time interval after which the clocks of a cluster are resynchronized. This time interval is determined by the ClkSyn flags in the MEDL.

Role The function assumed by a node at a particular point in time.

Round Slot Slot oriented to the cluster cycle not to the TDMA round.

Shadow Node Shadow nodes are provided to avoid spare exhaustion after permanent failures.

Shared Slot A slot that is shared by more than one node in a cluster cycle; the nodes sharing the slot are multiplexed nodes, the group of nodes sharing the slot are called *virtual member node*.

Slot A slot is the smallest time interval of a TDMA schedule.

Node Membership The node membership service informs all nodes of a cluster about the operational state of each node within a latency of about one TDMA round (see membership). The node membership is shortly called ‘membership’.

Node Slot The total bus capacity is statically subdivided into several time windows called node slots, which are exclusively assigned to nodes for transmission. The relation between the length of the node slot and the length of the TDMA round represents the share of the total available bandwidth that is assigned to this node and can be utilized by it.

Startup Timeout The startup timeout is a node specific, unique timeout value (relevant only for nodes that have cold start permission).

Synchronization Flag (SYF) A flag in the MEDL that denotes that the node sending in this slot is part of the ensemble of nodes that provide the basis for the global clock synchronization.

TDMA Time Division Multiple Access. Media access scheme used by the time triggered protocol family – bus access is divided into non-intersecting time slots which are statically assigned to the communicating nodes in a cluster.

TDMA Round The sequence of node slots in a cluster.

Tentative Frame Frame that is correct after the second CRC check. This means the first successor during the acknowledgment is right and the controller has failed.

Timer Field A CNI control area field that can be used as a time interrupt source by the host computer. A.k.a. alarm timer. TTP/C controllers must support at least one such timer to allow efficient access to the global time for the host CPU, e.g., for embedded operating systems.

Time Gateway The time gateway is a special node that has access to an external reference time (e.g., a GPS receiver or another cluster). The time gateway provides the drift rate correction term for the cluster to allow for external rate correction.

Time Interrupt A programmable interrupt generated by the controller whenever the timer field in the CNI is equal to the current global time. See Timer Field.

Transmission Phase (TP) The period of time which is reserved for sending or receiving frames in a slot. The start of the TP is called Action Time.

Valid Frame A frame is valid if it has a syntactically correct frame format, i.e., it starts during the time window contained in the MEDL and does not violate any coding rules. A valid frame can still be incorrect if it was corrupted during transmission or if sender and receiver are in disagreement.

Virtual Member Node A virtual member node consists of a set of multiplexed nodes that share a single node slot.

3.2 Acronyms

AR	Allow external rate correction
BG	Bus guardian
BIST	Built-in self test
BR	BIST error
C-state	Controller state
CA	Controller await
CB	Communication blackout
CC	Concurrency control error
CE	Clique error
CF	Cold start allowed flag
Ch0	Channel 0 (same as channel A)
Ch1	Channel 1 (same as channel B)
ChA	Channel A (same as channel 0)
ChB	Channel B (same as channel 1)
CIA	Cold start integration allowed flag
ClkSyn	Clock synchronization slot flag
CNI	Communication network interface
CO	Controller on flag
CPM	Clear pending mode change
CR	Controller ready interrupt
CRC	Cyclic redundancy check
DMC	Deferred pending mode change
DPRAM	Dual-ported random access memory
FT-COM	Fault-tolerance communication
FT-COM CNI	Fault-tolerance communication network interface
HE	Host error interrupt
ID	Identifier
IFG	Interframe gap
MC	MEDL CRC error
MCP	Mode change permissions
ME	Membership error
MEDL	Message descriptor list
MIC	Minimum integration count
ML	Membership loss interrupt
MM	Multiplexed membership flag
MMFC	Maximum membership failure count
MOC	Mode change interrupt
MV	Mode violation error

NR	Frame not ready error
OS	Operating system
OSEK	Offene Systeme und deren Schnittstellen für die Elektronik im Kraftfahrzeug (Open Systems and the Corresponding Interfaces for Automotive Electronics)
PE	Protocol error interrupt
PRP	Post-receive phase
PSP	Pre-send phase
RA	Reintegration allowed flag
RS	Ready status flag
SAE	Society for automotive engineers
SE	Synchronization error
SO	Slot occupied error
SYF	Synchronization frame flag
TI	Timer interrupt
TDMA	Time division multiple access
TP	Transmission phase
TTA	Time-triggered architecture
TTP	Time-triggered protocol
UI	User defined interrupt

4 Description of the Architecture

4.1 General

The TTP/C protocol controls the exchange of data between different electronic modules connected to a TTP/C network. In a time-triggered architecture the communication system decides autonomously according to a static schedule when to exchange data. Every controller contains its own schedule control data, which is stored in a personalized *message descriptor list (MEDL)*, that specifies at which instant a data transmission must be performed or accepted by the controller (described in chapter 10 on page 79).

4.2 Structure of the TTP/C Network

A TTP/C network consists of a set of electronic modules called *nodes* that are connected by two replicated channels as shown in figure 4.1. These two channels are called *channel 0* (or *channel A*) and *channel 1* (or *channel B*). A TTP/C network and the associated electronic modules are called a *cluster*.

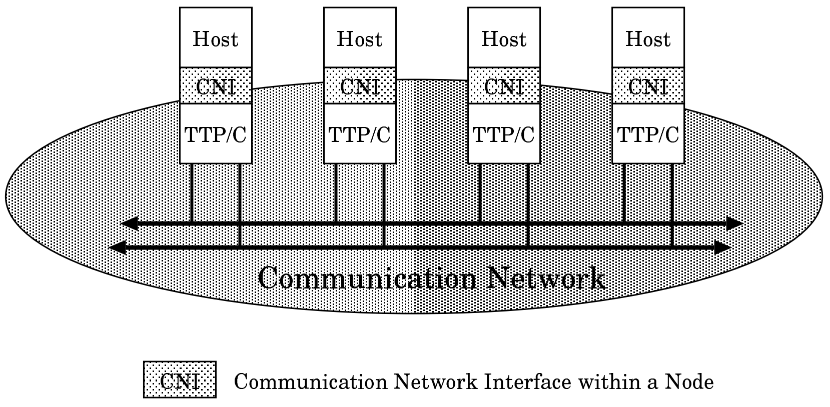

Figure 4.1: An Example of a TTP/C Network

4.2.1 Network Topologies

TTP/C works on different network topologies (figure 4.2 on the next page) with the *bus topology* and the *star topology* as the most common ones. The star uses central *star-coupler* devices, one for each channel, which also operate as a central *bus guardian* and provide a better fault-protection of the communication network than the bus topology (see section 9.2.4 on page 70).

4.2.2 Media Access

Access to the physical layer is controlled by a cyclic time-division multiple access (TDMA) scheme derived from a global notion of time. The sequence of node slots in which each electronic module performs (at most) one transmission on each of the redundant communication channels forms a TDMA round. After a TDMA round has been completed, the next TDMA round with the same temporal access pattern but possibly different data content is started. The number of TDMA rounds determines the length of the cluster cycle. After a cluster cycle is finished, the its transmission pattern starts again at the beginning of the next cluster cycle.

The media access is explained in details in chapter 5 on page 29.

4.3 Structure of an Electronic Module

An electronic module has the general structure shown in figure 4.3 on page 20.

It consists of a host computer, a controller and an input/output interface to the sensors and actuators in the vehicle. The host computer executes the application software.

The communication network interface (CNI, see chapter 11 on page 87) inside the node – between the host computer and the controller – consists of a memory area that allows simultaneous random access for the host CPU and the controller. There should also exist a signal (interrupt) line from the controller to the host.

The memory area is the data sharing interface between the host computer and the controller that holds the data that are exchanged between these two subsystems – its location depends on the controller implementation (e.g. DPRAM in the controller, or located in the host memory and accessed via DMA from the controller).

The signal line delivers two kinds of signals to the host computer:

- Ticks of the globally synchronized time – in order to reduce the load on the host computer, only significant ticks are signaled (programmable timer).
- Exceptional conditions like errors or other asynchronous events.

The controller comprises the protocol engine, the storage for the TTP/C control data (the TTP/C message descriptor list, MEDL), and an independent hardware unit, the bus guardian, to protect the bus from timing failures of the controller.

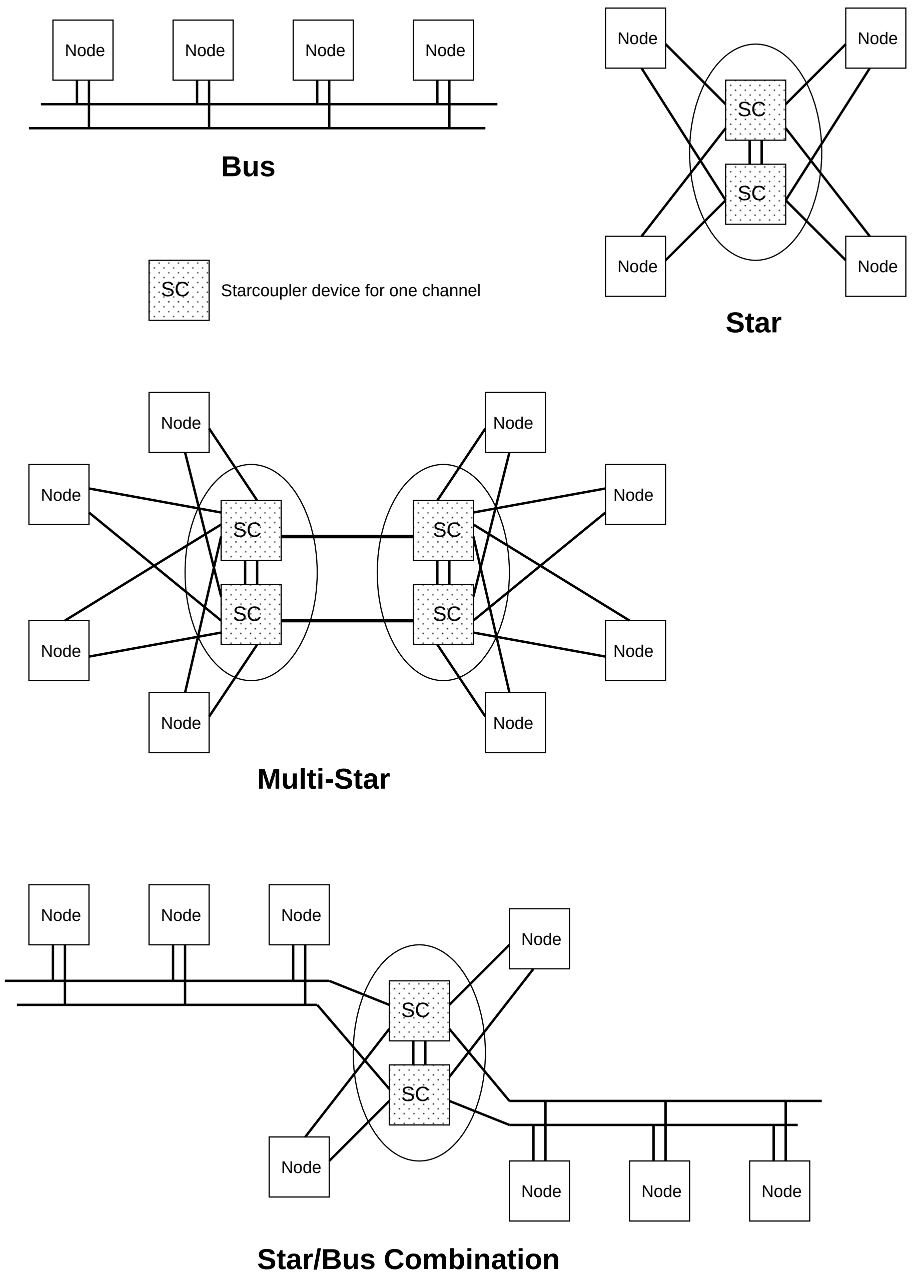

Figure 4.2: Examples of TTP/C Network Topologies

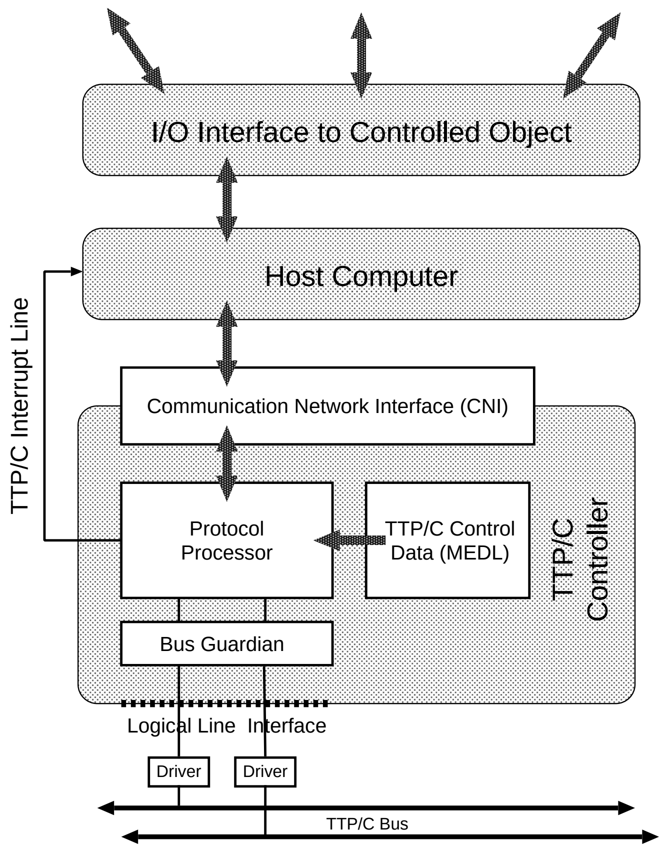

Figure 4.3: Overview of an Electronic Module

As far as a standardization is concerned, an electronic module consists of two main interfaces that have to be standardized in order to ensure interoperability:

1. The *communication network interface (CNI)* is required to ensure interoperability between the host and the TTP/C controller.
2. The *data link layer* is needed for interoperability between different TTP/C controllers.

4.4 Principle of Operation

The controller operates autonomously without any control signals from the host computer. The necessary control information is stored in the controller in a personalized MEDL (see chapter 10 on page 79) containing the following data items:

- For each data transmission (frame) the instant when the frame has to be sent and the address of the frame data field in the CNI where the application data must be fetched.
- The instant when a particular frame has to arrive and the address of the frame data field in the CNI where the received application data must be stored.
- Additional information for the protocol operation.

The number of entries in the MEDL corresponds to the length of the cluster cycles of the defined cluster modes.

During operation, TTP/C synchronizes the local clocks of the nodes to generate a common time base of known precision Π. Whenever the common time reaches an instant that is contained in the MEDL, the actions specified in the MEDL are carried out.

4.5 Frame versus Application Data versus Message

The terms *frame*, *application data* and *message* (see figure 4.4 on the next page) are used in the following chapters. They all describe a kind of ‘data’ but from different standpoints:

Frame A frame is a transmission (bit stream) of a defined length and coding on a TTP channel. TTP/C frames are protected by a CRC, which is being also part of the frame. Besides the application data the frame contains data used by the protocol for operation.

Application Data Application data exist only from the view of the controller: all the frame data that are not necessary for the protocol operation are in the scope of the host – these application data have no semantics for the controller. In case of a reception the controller stores the application data into the CNI or in case of a transmission it fetches them from the CNI.

Message Messages exist only from the view of the host: the application data are divided into parts that are interpreted using a semantic known by the host application or the later described FT-COM layer only. A message describes a state, like a temperature, a switch state or a variable in a distributed control loop. The TTP/C controller does not know anything about messages.

Considering the definitions before the term ‘message descriptor list’ for the static schedule is not correct: not the messages but the frames and the CNI location of the application data are described in the MEDL. Because the term ‘MEDL’ has been established by previous specifications, it is not changed.

4.6 Overview of the Protocol Services

The Time-Triggered Protocol provides the following services:

- Predictable and small bounded delays for all messages, even in the specified peak load and fault scenario.

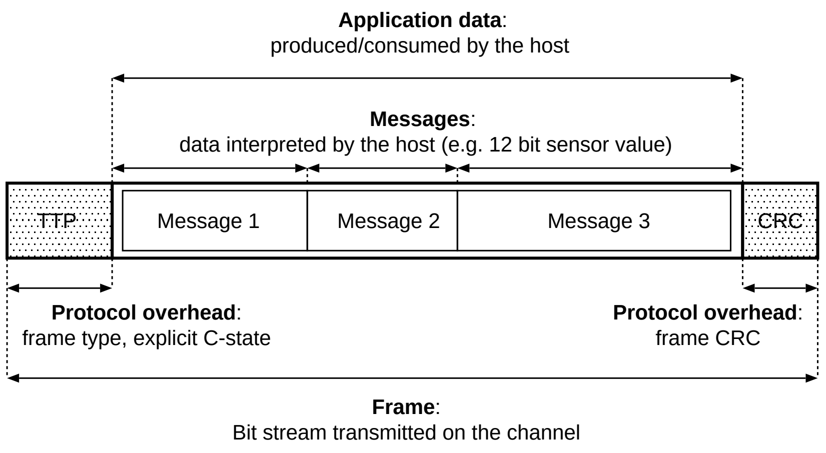
Application data:
produced/consumed by the host
Messages:
data interpreted by the host (e.g. 12 bit sensor value)
TTP
Message 1
Message 2
Message 3
CRC
Protocol overhead:
frame type, explicit C-state
Protocol overhead:
frame CRC
Frame:
Bit stream transmitted on the channel

Figure 4.4: Frame – Application Data – Message

- Temporal encapsulation of the subsystems such that a constructive test methodology is supported and the temporal interference of independently developed subsystems can be ruled out by the properties of the communication architecture.
- Rapid fault detection at the sender and receiver by the provision of a responsive membership service.
- Implicit message acknowledgment in group communication.
- Support of replicated channels and replicated nodes such that fault-tolerant architectures can be implemented. Provision of a distributed redundancy management service.
- Support of cluster mode changes.
- Provision of a clock synchronization service by a distributed fault-tolerant synchronization algorithm without any additional cost in frame size or an increase in the number of transmissions.
- No restriction in the use of network media since no bit arbitration is performed.
- High data efficiency because of low protocol overhead.

4.7 Layers of the Time-Triggered Architecture

The Time-Triggered Architecture is organized as a set of conceptual layers, as shown in figure 4.5 on the facing page.

This is a conceptual model that groups related functions into one layer. It is not an implementation model. The interface between the *protocol service layer* and the FT-COM (see section 4.7.4 on page 24) layer is called the *communication network interface, CNI or TTP CNI*. The interface between the FT-COM layer and the host layer is called the

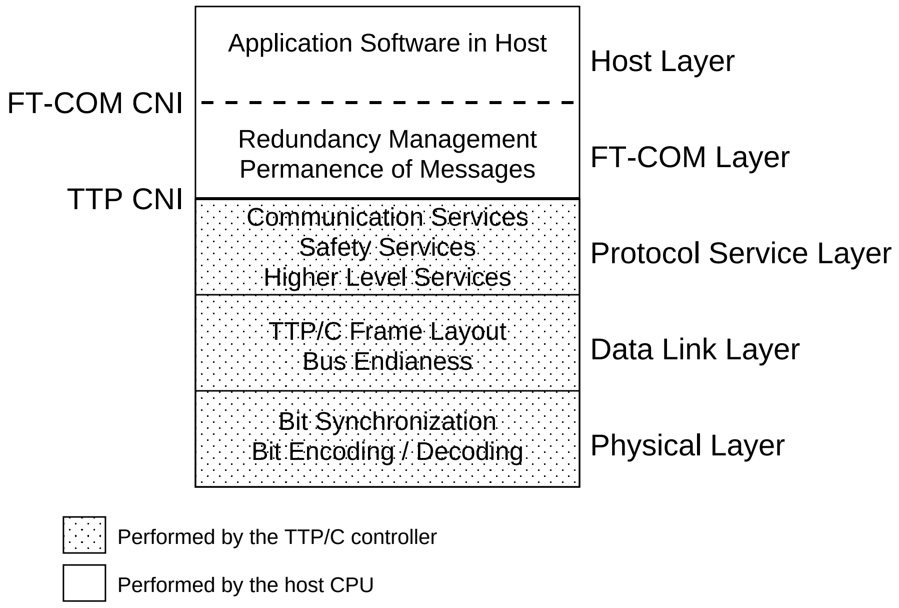

Figure 4.5: Layers of the TTA

FT-COM communication network interface, FT-COM CNI. The FT-COM layer and the host layer are not part of TTP/C but of TTA.

In the following sections the functions performed in each layer are outlined.

4.7.1 Physical Layer

TTP/C does not dictate a specific bit encoding nor the physical media used, but the requirements that must be fulfilled are described in chapter 7 on page 37.

4.7.2 Data-Link Layer

The data-link layer provides the means to exchange frames between the nodes, i.e. the format of the TTP/C frames are defined in this layer. The access scheme to the bus (which consists of two channels) is time-division multiple access (TDMA), controlled by the data stored in the MEDL of the controller.

During operation, two types of nodes are distinguished, *real member nodes* and *virtual member nodes*. A real member node has its own specific node slot which is not shared among other nodes. A virtual member node is a set of nodes that share a single node slot. The term member node is used to denote either a real member node or a virtual member node.

A member node sends exactly one frame on each of the two channels during every TDMA round. The lengths of the sending slots of the different member nodes can be different and correspond to the portion of the overall bandwidth assigned to the individual member node; each slot allows a specific amount of data transmission – the maximum data length depends on the CRC used to protect the frame. The duration of a node slot of a member node is the same in every TDMA round. It follows that all TDMA rounds have the same length.

4.7.3 Protocol Service Layer

The Protocol Service layer provides the protocol operation including all services of TTP/C that can be divided into the main categories:

- Communication services
- Safety services
- Higher level services

The communication services guarantee a reliable data transmission, the startup of a cluster, the reintegration of nodes, the acknowledgment and the fault-tolerant clock synchronization.

The safety services include the node membership, the clique avoidance algorithm and the independent bus guardian.

The higher level services include the mechanism for the deferred mode changes (also the optional distributed alarms, using the mode change mechanism), the external clock synchronization and the optional reconfiguration of a node.

4.7.4 FT-COM Layer

The FT-COM layer is responsible for the management of replicated application subsystems, i.e., it performs the redundancy management and agreement of replicated messages sent by different nodes. A subsystem is a (distributed) application task for a defined functionality of the (distributed) application. A subsystem is therefore located on several nodes and can be replicated on another set of nodes.

The FT-COM layer is described in the OSEK/VDX specification [OSE01] introduced by the OSEK/VDX Steering Committee.

4.7.5 Host Layer

The host layer contains the host application running on a node including the operating system and the control loops. Messages should be accessed from the application via the FT-COM CNI instead via the TTP CNI.

4.8 Protocol Constraints

A TTP/C system is a distributed fault-tolerant system that is designed to operate properly only if a minimum number of nodes is operational within a cluster.

4.8.1 Minimal fully Fault-Tolerant Configuration

In order to be fully fault-tolerant against Byzantine faults a minimal configuration of a TTP/C system consists of four real member nodes using a *fully independent* local or central bus guardian (see section 12 on page 72). If a cluster design contains more nodes than physically present in the cluster[1], special precautions should be taken during MEDL design to avoid that these nodes are assigned essential protocol functionality that cannot be performed by other nodes (e.g., cold start or active clock synchronization).

4.8.2 Minimal non Fault-Tolerant Configuration

If less than four nodes are available for clock synchronization, a cluster operates in a mode that is not fully fault-tolerant. The clock synchronization is not Byzantine resilient anymore[2]. Special precaution should be taken during MEDL design for this minimal configuration.

4.9 Fault Hypothesis

4.9.1 Internal Physical Faults

TTP/C is based on the fault-hypothesis that any single component in the system can fail in an arbitrary failure mode. This assumption is based on the fact that the likelihood of two concurrent independent component failures is remote enough to be considered a rare event that can be handled by an appropriate *never-give-up (NGU)* strategy. However, it should be noted that a very prompt error detection mechanisms is necessary to ensure that two consecutive single faults are not becoming concurrent.

With respect to hardware faults, the TTP/C protocol is designed to isolate and tolerate single node faults in a properly configured cluster [Kop94a] during synchronized operation (‘running’ cluster).

At the architecture level it is ensured that a faulty node cannot prevent correct nodes from exchanging their data by introducing a bus guardian (see section 9.2.4 on page 70) in TTA-bus networks or a central guardian in TTA-star networks. The bus guardian guarantees that a node can only send once in a TDMA round, thus eliminating the problem of

[1]E.g. for further extensions or optional devices

[2]It has been shown in [Pea80] that at least 4 nodes must be present to ensure tolerance of Byzantine faults.

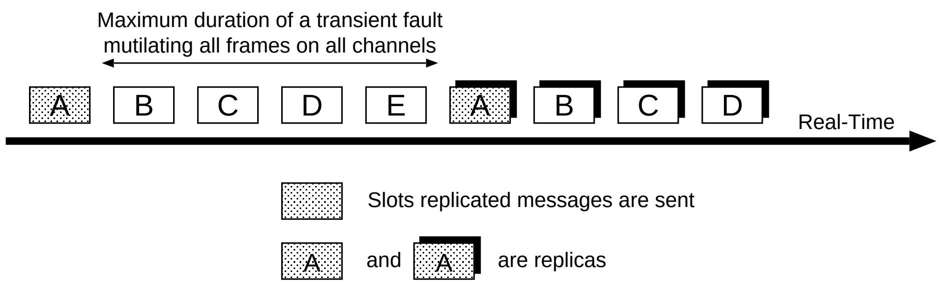
Maximum duration of a transient fault mutilating all frames on all channels
A
B
C
D
E
A
B
C
D
Real-Time
Slots replicated messages are sent
A
and
A
are replicas

Figure 4.6: Maximum Duration of a Transient Fault

babbling idiots that monopolize the communication medium. Using an independent central bus guardian the protocol tolerates also faults during the startup phase and guarantees a successful transition into the synchronized operation.

Finally, the TTP/C protocol implements a never-give-up strategy for multiple fault scenarios: if a node detects faults that are not covered by the fault hypothesis, it informs the application. The application may now decide either to shut-down in fail-safe environments or to restart with an agreed consistent state among all nodes of the distributed system in fail-operational environments. The resilience of a TTP/C cluster with respect to multiple failures depends on the application specific configuration. Many multiple internal faults will be tolerated under normal operating conditions.

4.9.2 External Physical Faults

As long as an external fault impacts only a single logical unit (e.g. a part of a replicated application subsystem, or one of the both channels), the TTP/C system will tolerate the fault.

Assuming a replicated message is sent by two nodes: correlated external faults during the first node's slot that destroy all frames in this slot containing the message will be tolerated in a properly configured cluster [3] if the temporal distance between the end of the node slot of the first replica and the start of the node slot of the second replica (the second node that transmits a frame containing the message) is larger than the duration of the transient fault (figure 4.6).

4.9.3 Design Faults in the TTP/C Controller

The controller design and the cluster design are assumed to be free of design faults.

[3] This means using redundancy (FT-COM layer) for critical messages

The personalized MEDLs for all controllers in the cluster are assumed to be correctly derived from the same cluster design[4] for the specific controllers. Different controller implementations may require different personalized MEDLs for the same role.

The *syntactic* integrity of the personalized MEDL stored in the specific controller is checked at regular intervals in order to detect hardware faults.

4.9.4 Design Faults in the Host Computer Software

A fault (even a malicious fault) in the software of the host computer of a correctly configured cluster can corrupt the data produced by this node but cannot cause any interference with the protocol operation of the other nodes in the cluster. Design faults in the software may, however, render redundancy useless.

4.9.5 Permanent Slightly-Off-Specification Faults

In a distributed system where the individual nodes use local hardware to determine about the correctness or incorrectness of the transmission, there is always a ‘grey area’ of parameters (e.g., bit timing, voltage level, etc.) which will be accepted by some nodes but rejected by others. If a node continuously emits such slightly-off-specification transmissions, a distributed system may never reach a conclusion about the correctness of this node (repeated Byzantine faults).

TTP/C supports tolerance of this kind of fault by means of a bus guardian unit with additional functionality.

4.9.6 Spatial Proximity Faults

This specific kind of external physical fault impacts the network by severing all links to one node (or potentially more than one). This fault can only be tolerated by appropriate network topologies, i.e., a star or ring architecture – a bus architecture cannot tolerate this fault in normal cases.

TTP/C supports tolerance of spatial proximity faults by supporting star and ring topologies with arbitrary propagation delays.

[4]A mechanism for checking this property is provided by the schedule ID that must be encoded into the frame CRC.

5 Media Access

5.1 TDMA Scheme

TTP/C uses time-division multiple access (TDMA) as medium access strategy during the synchronized operation of nodes. The data transport between nodes is not point-to-point oriented but broadcast: every node receives all data transmissions available on the bus.

In TDMA protocols each node is permitted to periodically utilize the full transmission capacity of the bus for some fixed amount of time called *TDMA slot* which is the interval from one transmission start to the next one. Because a node needs a short phase for preparing the actions in the next slot (read transmission parameters from MEDL, initialize transceiver,..) the logical *node slot* starts with this preparation phase and ends at the next one. It has the same length as its correlated TDMA slot, but begins and ends earlier. More details are given in section 5.6 on page 32. In its assigned slot a node sends frames on both channels – the frames on Ch0 and Ch1 do not have to be the same and may differ in their lengths and contents.

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 in a TTP/C cluster and the static bus access schedule that is known by all participants.

The periodic sequence of TDMA slots (or node slots) is called a *TDMA round*. As far as the length of the node slots and the sending sequence of the different nodes are concerned, all TDMA rounds are equal. The only possible difference between two TDMA rounds is the content and the length of the frames sent in the node slots. The pattern of periodically recurring TDMA rounds – with may differ in the frame lengths and the frame contents – is called *cluster cycle*. The *round slot* describes a logical slot in the cluster cycle including the node slot information (e.g. slot length) and the data description parameters (message addresses and size). Figure 5.1 on the next page illustrates the division of a cluster cycle into round slots and the division of TDMA rounds into node slots.

During the startup phase a TTP/C cluster uses an asynchronous media access where collisions may be possible, described in section 9.1.1 on page 48.

5.2 Multiplexed Nodes

It is possible that several nodes can share a single node slot to improve bandwidth utilization. A node sharing a node slot with one or more other nodes is called a *multiplexed node* [Pal97]. Multiplexed nodes are statically assigned to particular TDMA rounds. There

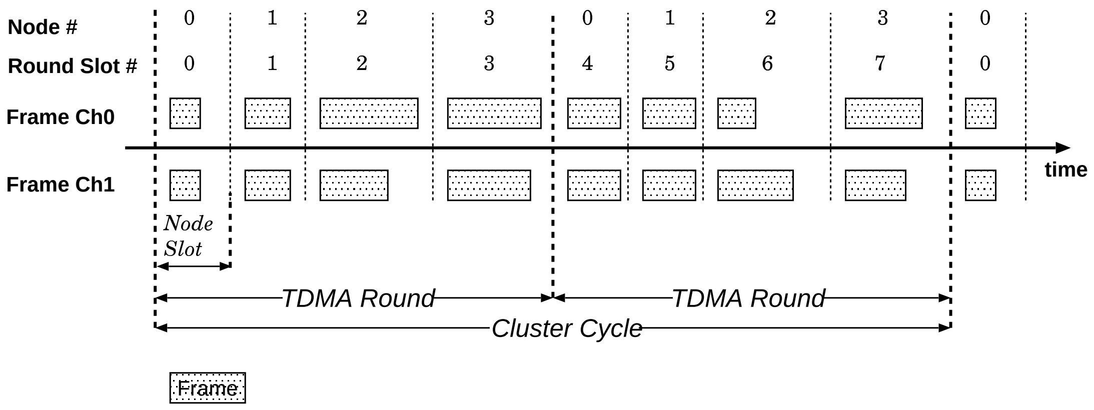

Figure 5.1: Media Access Scheme

is thus no conflict about the point in time when a multiplexed node can send a frame. The set of multiplexed nodes that share a node slot is called a *virtual member node*.

Figure 5.2 shows an example of a cluster cycle consisting of four TDMA rounds. The last slot is shared by the multiplexed 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 the round 3. Node 3 has half the transmission frequency of of a real member node having a sending slot in each TDMA round. The node 4 and node 5 have a quarter of a transmission frequency of a real member node because they send both only once during cluster cycle while e.g. node 0 sends four times.

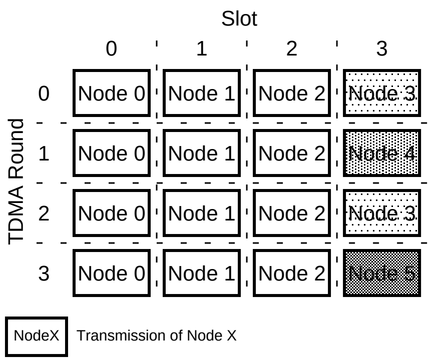

Figure 5.2: Multiplexed Slot Assignment

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

5.3 Shadow Nodes

Shadow nodes are provided to avoid spare exhaustion after permanent failures.

A shadow node is a node that does not own a node slot, but can acquire a node slot if an active node fails.

A shadow node can listen to all frames on both channels but is not allowed to send a frame. After the failure of an active node the host of the shadow can activate a shadow node to acquire the sending slot of the failed node. As soon as a shadow node acquires a node slot it becomes a member node. Without reconfiguration [1] a node can be the shadow of only one other node.

5.4 Passive Nodes

A node is passive on the bus (it does not send in its sending slot) if its host does not update the host life-sign which is used to validate the host availability (from the controller's point of view such a host is not operational and therefore there exists no valid data to transmit). The host can intentionally use this mechanism to prevent its controller from sending.

It is also possible to define nodes to be always passive and never consider transmission; e.g., a monitor or diagnostic node is always passive and must not send.

In order to ensure that a *permanent passive node* does not transmit frames on the bus even if the host starts updating its life-sign, the node can be configured as permanently passive by its MEDL configuration.

The host of a passive node can read all frame data from the CNI, but can never send a frame, neither during cluster startup nor during synchronized operation. A passive node may, however, send frames during download.

5.5 TDMA Round Assignment

Each physical node is assigned statically (off-line) to one or more particular TDMA rounds of the cluster cycle. A controller may only acquire its node slot in a round if it is allowed to reintegrate in its node slot of that round. Different controllers sharing a node slot must not be allowed to acquire this slot in the same round of a cluster cycle. This mechanism avoids that a number of shadow nodes which want to become active at the same instance of time acquire the same node slot.

The set of node slots that may potentially be acquired by a controller is called the *reconfiguration scope* of the controller. If a controller does not support reconfiguration, the reconfiguration scope only consists of the default role of the node.

[1] Reconfiguration is removed from the standard TTP services and therefore not described in this document

5.6 Inter-Slot Timing

It must be differentiated between logical node and TDMA slot: the *TDMA slot* begins with the transmission of a node and ends with the the transmission start of the successor node. The start of the *transmission phase (TP)* is called *action time (AT)*. The action time is reached on all sychronized nodes of the cluster at the same point in time (within precision). The node to which the TDMA slot belongs starts the transmission, the other nodes are enabled to accept a reception. The interval during which no transmission/reception is performed (no traffic on the bus) is called *inter-frame gap (IFG)*. During this period the controller processes the received data, performs the protocol services (clock synchronisation, acknowledgment, etc.) and prepares the action of the next slot. The idle phase is also part of the IFG. During the idle phase the controller does not do anything, it waits for the start of the *pre-send phase (PSP)*.

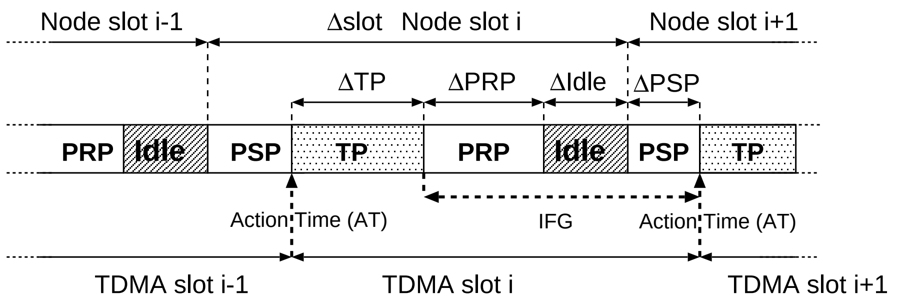

Figure 5.3: Slot Timing

The PSP is necessary to load the schedule information of the next slot from the MEDL and to prepare the transmission/reception of data at the next AT, so it is the start of the *node slot*, which includes all operation belonging to one data transmission. The evaluation of received data and execution of protocol services is called *post receive phase (PRP)*. The durations of the phases (except of the transmission phase) depends of the actions performed in the current slot, so there are no deterministic points in time of the start/end of the phases during the IFG, but maximum duration can be evaluated (e.g. the PRP after sending is shorter than after receiving). Figure 5.3 shows the partitions of a slot.

At first sight the idle phase makes no sense, but it is used to stretch the slot to the duration defined from the schedule designer. During the TP and the idle phase the host also has consistent read access to the status information of the controller. During the TP only messages not affected from the current active transmission can be accessed or modified – during the idle phase the host has full access to all messages and can access controller status information consistently.

If a slot duration and the length of the TP are given in the schedule, the idle phase begins at

$$Idle_{start} \quad = \quad AT + \Delta_{TP} + \Delta_{PRP_{max}} \tag{5.1}$$

and has a duration of

$$\Delta_{Idle} \quad = \quad \Delta_{slot} - (\Delta_{TP} + \Delta_{PRP_{max}} + \Delta_{PSP_{max}}) \tag{5.2}$$

$\Delta_{PRP_{max}}$ and $\Delta_{PSP_{max}}$ are the worst-case post-receive and pre-send phases with the longest durations [2].

[2]Implementation dependent, controllers may have constant phases

6 Controller State

The *controller state (C-state)* is a collection of state variables which describes the internal state of a TTP/C controller. Although the C-state is calculated locally by each controller, it represents a global view of the cluster and must agree with the calculated C-states of the other nodes in a correctly synchronized TTP/C cluster. A C-state disagreement between a node and the majority of the other nodes will mark this node as faulty. Thus the exchange of the C-state between the nodes is one of the basic concepts of TTP/C to support fault detection.

The C-state consists of the following elements (explanations are given later with the according services):

Global Time The global time of the next transmission in node granularity (macroticks), updated at the start of the node slot (begin of PSP).

Round Slot Position The current slot in the cluster cycle (currently processed entry in the static schedule), updated at the start of the node slot (begin of PSP).

Cluster Mode The current active cluster mode the controller is operating in, updated at the start of the node slot (begin of PSP).

Deferred Pending Mode Changes (DMC) If a mode change is requested for the start of the next cluster cycle, the pending request is saved in the C-state, updated at the end of the PRP after the request is received.

Membership Information Consistent view of the activity of all nodes in the cluster (node membership, see section 9.2.1 on page 68). A node is in the memberhship after having correctly sent in its last slot. A node is outside the membership, if having not sent or sent incorrectly. The membership is updated after the acknowledgment algorithm is performed (end of PRP).

6.1 C-state Validity

The C-state of a node becomes valid (assumed as agreed with the C-states of the majority of nodes)

- after integration, that means after reception of the first valid frame with explicit C-state (see section 9.1.2 on page 49).
- after reception of the first correct frame in case of cold starting (see section 9.1.1 on page 48).

If the local C-state of a controller becomes valid, the *C-state valid* signal (section 11.3.2 on page 95) is raised to the host. If the controller turns itself off in case of a detected error (e.g. due to clique detection, see section 9.2.2 on page 69), the C-state becomes invalid.

7 Physical Layer

Although the TTP/C protocol does not rely on a particular physical medium or bus coding scheme, some constraints must be fulfilled:

- The TTP/C bus must consist of two independent physical channels, which may be based on different physical layers.
- TTP/C needs a shared broadcast medium.
- The boundaries of the propagation delay must be known.

The choice of a suitable physical layer depends on the application of a TTP/C network regarding transmission speed, physical environment and dimension.

7.1 Physical Layer Compatibility

Nodes that should communicate in the same cluster must provide interfaces to the same physical layer, that means bus drivers must be available and the bit encoding on the medium must be supported by all nodes. A common transmission speed accepted by all nodes must also be available.

8 Data Link Layer

The data link layer deals with the access to the communication channels and the transmission of frames. Sharing the same data link layer is necessary for the compatibility of controllers, i.e. the controllers can interact in the same cluster.

8.1 Frame Types and Formats

8.1.1 Contents of Frames

TTP/C frames are coherent sequences of data, transmitted as broadcasts from a controller during its slot. For single failure tolerance, frames are transmitted simultaneously on both channels. A frame consists of the following parts:

- Application data prepared by the host
- Explicit C-state for agreement check between the synchronized nodes and integration of unsynchronized nodes
- Mode change requests
- Additional frame description data, e.g. a frame type identifier
- CRC calculated over the frame for detection of transmission faults
- Implicit C-state calculated into the frame CRC, if no explicit C-state is transmitted
- Unique cluster schedule ID that is also calculated into the frame's CRC (not explicit in the frame) to guarantee that only nodes with the same schedule can communicate with each other. To detect a faulty cross-over of the channels it is recommended to choose different parts of the schedule ID for the two channels.

An exception is the so-called *cold start frame*. This frame is used for the cluster startup and must contain the following information:

- Global time of sending (C-state time)
- Round slot position of the sender (must be located in the first TDMA round in the first cluster mode)
- Frame type identifier, which indicates this frame as cold start frame
- CRC calculated over the frame for detection of transmission faults
- Unique cluster schedule ID is also calculated into the frame's CRC

Other nodes may integrate on this frame by taking over the C-state time and the schedule position. The sender is identified by the round slot position and is set into the membership.

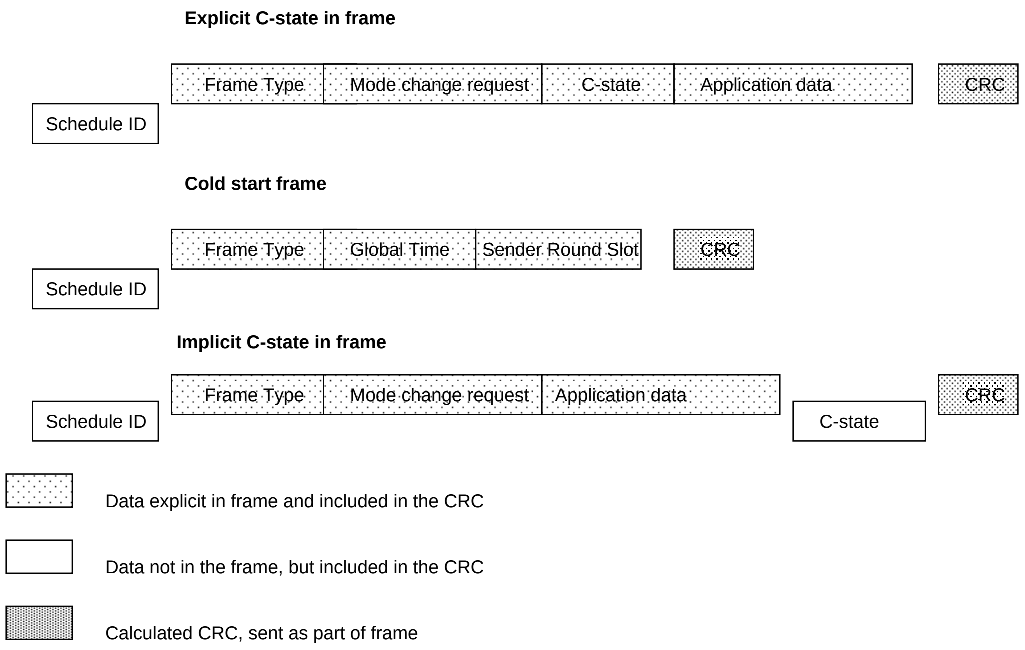

Figure 8.1: Implicit versus Explicit C-state

8.1.2 Explicit versus Implicit C-state

For the frame evaluation we must distinguish between the implicit and the explicit C-state: If an explicit C-state is given, the CRC is only used for detecting transmission errors. The explicit C-state is then compared with the receiver's one. An implicit C-state in the CRC cannot be extracted, but the CRCs can be compared for agreement.

The use of implicit C-states in frames improves data transmission performance, because no bandwidth on the channel is spent for the C-state transmission (which is overhead for the application, but necessary for the protocol operation). Especially at lower transmission speeds the gain of additional bandwith usable for application data is noticeable. On the other hand the logic for comparing the local C-state with the received implicit one is more complex at the receiver and may (depending on the implementation) take a longer processing time than simply comparing two explicit C-states.

Explicit C-states in frames are required for the integration process and for cluster startup: a controller can only integrate on a received frame if it can extract the C-state and take it over for its local one (that is not existent before the integration is finished). The ratio between frames with implicit and explicit C-states and the spreading of the explicit ones determines the integration duration of nodes into a running cluster. It is recommended that during the cluster startup mode (the first cluster mode used to establish a synchronized cluster, normally not intended for application data exchange) only frames containing explicit C-states are transmitted.

Figure 8.1 on the facing page illustrates the difference beween a frame containing an explicit C-state and a frame with an implicit one.

8.2 Frame States

After reception a TTP/C controller determines the *frame status* of each received frame. This is an evaluation of the frame contents to detect transmission faults or C-state disagreement. The controller uses the frame states for the acknowledgment algorithm and the clique avoidance. Figure 8.2 on page 43 shows the algorithm for the frame status calculation.

8.2.1 Null-Frame

If no activity (not even noise) is observed on the channel of the transmission medium during the transmission phase, the expected frame is considered to be a *null-frame*. A missing or fail-silent sender transmits null-frames.

Since transmission outside of the expected time frame for transmission is not observed by the controller, a null-frame is detected if no activity is detected from the opening of the receive window (see section 11 on page 57) until the end of the transmission phase, even if there is noise or other bus activity outside of this time interval.

8.2.2 Invalid Frame

If bus activity is detected by the receiver during the time interval from the opening of the receive window until the end of the transmission phase, but no valid frame is received, the frame is called *invalid*.

8.2.3 Valid Frame

A received frame is syntactically *valid* if the following conditions hold:

- The frame starts during the receive window. The size of this window depends on the precision parameter in the MEDL.
- 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.

8.2.4 C-state Agreement

Received frames are checked for C-state agreement: all parts of the frame C-state (see chapter 6 on page 35) except the membership information must be identical with the local C-state for a positive agreement. For a receiving node there exists up to two test cases for the membership in the C-state of the received frame each depending on the state of its acknowledgment algorithm:

- If the receiver is already acknowledged: receiver's own membership set, if it has sent in its last slot
- Waiting for the first successor: receiver's own membership set – or receiver's own membership unset. The sender must be set in the membership.
- Waiting for the second successor: first successor unset and receiver's own membership set – or first successor set and receiver's own membership unset. In both cases the sender of the frame must be set in the membership.

In all three cases the first scenario is called *C-state scenario 1*, the second (if available) *C-state scenario 2*. Knowledge about the acknowledgment algorithm (see section 9.1.6 on page 61) is necessary to understand the meaning of the test cases.

Both the explict and implict C-state are checked with the tests mentioned before. In the first case a comparison of the C-state data structures is made, in the other case two local CRCs are built with the membership combinations and compared with the received CRC.

8.2.5 Incorrect Frames

An *incorrect frame* is a valid frame with an incorrect CRC check (both C-state scenarios failed) in case of an implicit C-state, or in case of an explicit C-state, with a C-state disagreement (both C-state scenarios failed) or a failed CRC over the frame contents (transmission error).

8.2.6 Tentative Frames

A *tentative frame* is a valid frame with an agreed C-state according to scenario 2 (the node's own membership flag is cleared). Tentative frames can only exist, if the node is still not acknowledged.

8.2.7 Correct Frame

A *correct frame* is a valid frame which passed the CRC check and all additional semantic checks at the receiver. The explict/implicit C-state fully agrees between sender and receiver.

Furthermore, a correct CRC indicates that sender and receiver have been connected correctly, since otherwise the different CRC init values (see section 8.3 on page 44) would result in CRC errors.

8.2.8 Other Error Frame

A frame marked with *other error* 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 10.3.4 on page 83).

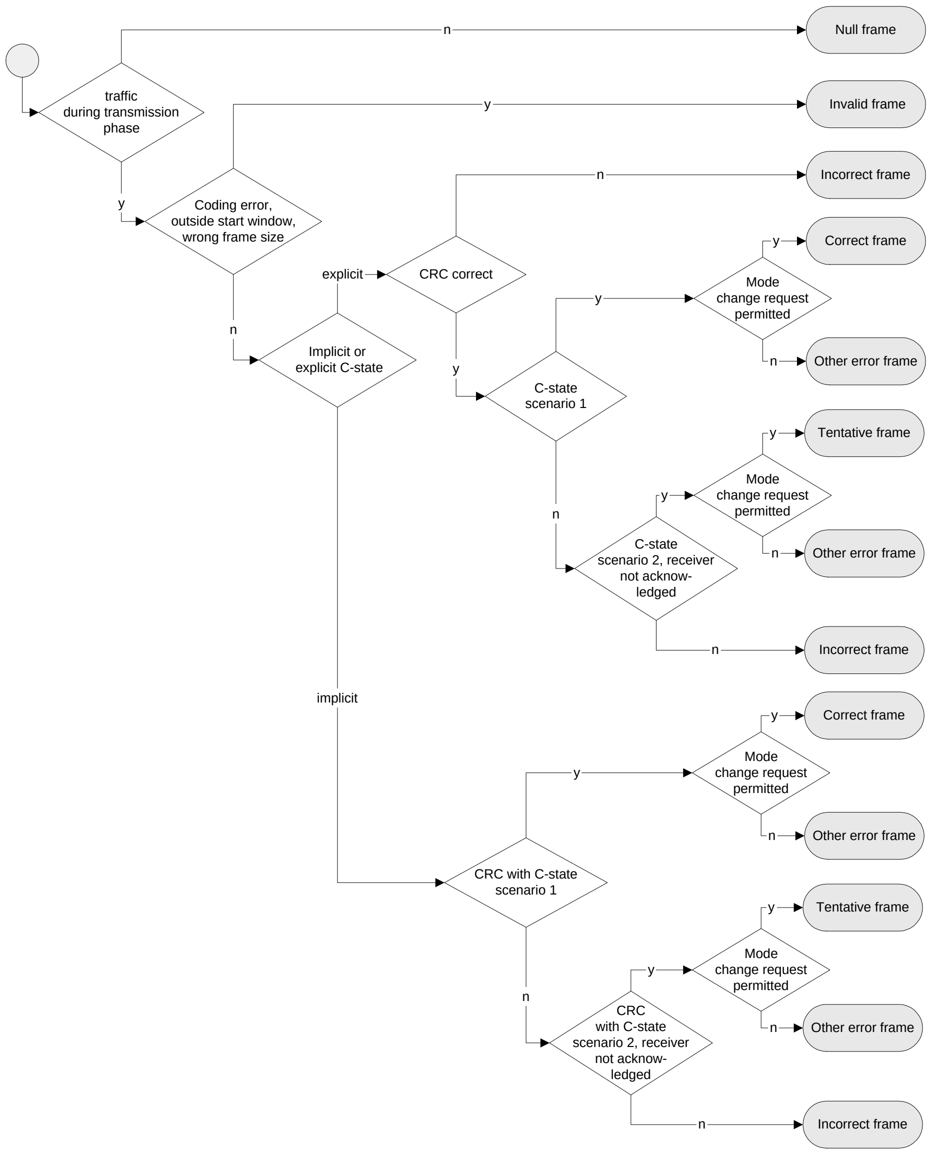
Null frame
Invalid frame
Incorrect frame
Correct frame
Other error frame
Tentative frame
traffic during transmission phase
Coding error, outside start window, wrong frame size
CRC correct
Implicit or explicit C-state
explicit
implicit
Mode change request permitted
C-state scenario 1
C-state scenario 2, receiver not acknow-ledged
CRC with C-state scenario 1
CRC with C-state scenario 2, receiver not acknow-ledged
y
n

Figure 8.2: Frame Status Calculation

8.3 Frame CRC Calculation

Every frame contains a CRC field which protects its contents. The CRC is only used for error detection, not for error correction.

The CRC calculation is initialized with the schedule ID parameter contained in the MEDL. This parameter is divided into two CRC initialization values, one for channel 0 and a different one for channel 1. This prevents that a node which

- contains an incompatible MEDL[1], and therefore a different schedule ID
- is connected to the bus with crossed-out channels[2], and therefore uses the CRC initialization value of Ch0 on Ch1 and vice versa

receives or sends correct frames.

A controller that has a different schedule ID stored in its MEDL than the sender cannot receive the frame correctly because of this CRC calculation method. This ensures that controllers with different schedule IDs – which in turn indicates that their MEDLs were not derived from the same cluster design – cannot run together in a cluster.

8.3.1 Polynomial

The CRC is specified by its polynomial. The used polynomial depends on the maximum frame length used by the protocol and must have a Hamming distance of 6.

8.3.2 CRC Calculation of TTP/C Frame

There are two different cases of CRC calculation:

- If the frame contains an explicit C-state, this C-state is part of the frame's data and the CRC is calculated over the complete frame data initialized with the schedule ID part for the channel. Since the C-state of the sender is *not* incorporated in the CRC over the data part, a CRC error of the frame at the receiver is never caused by a C-state disagreement with the sender. Cold start frames are also treated as frames with explicit (but shortened) C-state.
- In case of an implicit C-state, the C-state is not part of the frame's data but the CRC is calculated over the frame data and the C-state initialized with the schedule ID part for the channel. The sender incorporates a copy of its local C-state whereas the receiver uses a copy of its own local C-state. This mechanism enforces that frames are only accepted by the receiver if the C-state of the sender is the same as the C-state of the receiver (regarding both C-state scenarios).

The CRC calculation is illustrated in figure 8.1 on page 40.

[1]This constitutes a design fault and is not covered by the fault hypothesis.

[2] This mis-configuration can lead to problems if the propagation delays differ significantly between the channels, or if the frames on the two channels are not fully redundant.

8.3.3 CRC Calculation of Download Frames

Download frames are an exception: because they must be accepted by all nodes independent of the cluster, the CRC is not initialized with the schedule ID but with a constant value that is known to all nodes that are compatible to a specified data link layer.

8.4 Slot Status

Because TTP/C uses at least two transmission channels, the controller combines 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:

Correct ← Tentative ← Incorrect ← Other Error ← Null-frame ← Invalid

This means if the controller receives a correct frame on Ch0 and an invalid one on Ch1, the slot status is 'correct'. The slot status is used for the acknowledgment and the clique avoidance algorithms.

8.5 Data Link Layer Compatibility

A set of nodes that incorporates in the same cluster must have a compatible physical layer, a compatible data link layer and the same static schedule. In details that means:

- The physical medium must be supported by the bus drivers of all nodes.
- The bit encoding on the physical medium must be the same on all nodes.
- A common transmission speed accepted by all nodes must be available.
- The bus endianess and the data granularity for the bus access must be compatible.
- The structural format of the used TTP/C frames must be supported by all nodes.
- The semantics of the transmitted TTP/C frames must be the same in all nodes.
- All nodes must use the same CRC polynomial for frame protection.
- The static schedule information in the nodes (MEDLs) must describe the same scheduled TTP/C network. This is necessary for the nodes to have the equal a priory knowledge about frame arrival times, slot lengths, frame lengths and frame types.

9 Protocol Service Layer

The TTP/C protocol provides different groups of services to the higher layers:

Communication Services Responsible for the data exchange, cluster startup and integration, bus-noise tolerance, the implicit (and explicit) acknowledgment scheme (see section 9.1.6 on page 61), and the fault-tolerant clock synchronization necessary for a temporal correct bus access according the TDMA scheme. The communication services are the basic services of the protocol that provide the functionality of a network protocol for distributed real-time systems and establish a temporal firewall to the host for message exchange. Also the fault-tolerance aspect is covered by the communication services that operate correctly in all single-fault scenarios.

Safety Services TTP/C is a network protocol for safety critical real-time systems – therefore safety services are necessary. The safety services establish the node membership, the clique avoidance algorithm, the independent bus guardian functionality (central or peripheral) and the host/controller life sign algorithm.

These services should guarantee a fail-silent behavior of a faulty node in the time domain and prevents the existence of different node cliques with unequal C-states.

Higher Level Services Higher level services are requested by the host and are used for switching between transmission schedules at run-time (cluster mode changes), synchronizing the cluster with an external clock or an other TTP/C cluster (external clock correction) or for a role-change of a node (optional reconfiguration).

Both the communication and the safety services provide a fail-operational behavior of a running TTP/C cluster: The safety services prevent the distribution and impact of faults in the cluster, the communciation services tolerate single-faults and detect faulty nodes. The acknowledgment algorithm is part of the communication services, but it is based on the membership service. The higher level services are for a more flexible application design and may also be used for fault-tolerance behavior implemented in the host layer.

Other services, e.g. used for maintenance are not topic of the TTP/C specification, but must be considered if they have influence on the protocol process.

9.1 Communication Services

9.1.1 Cluster Startup

The change from an unsynchronized cluster to a synchronized one is termed *cluster startup*. This process is performed at power-on or reset of the whole cluster, which means that a minimum of two synchronized nodes does not exist.

The startup of each node consists of three steps:

- Initialization of the host and the TTP/C controller.
- Listening on the channels for frames with explicit C-state for the duration of the listen timeout. If such a frame is received, the controller starts the integration to adopt the C-state.
- If no running cluster is detected (no frames with explicit C-state have been received), it checks the conditions for performing a cold start:
  - the host has updated its life-sign
  - the *Cold Start Allowed flag (CF)* is set in the MEDL
  - the maximum number of allowed cold starts for this node is not reached

  If these conditions are fulfilled, the node sends a cold start frame (see section 8.1.1 on page 39), increases the *cold start counter* and processes the TDMA scheme.

A more detailed description of the cluster startup process is given in the protocol state overview (section 12.2 on page 103).

In architectures without independent bus guardian the number of a node's allowed cold starts should be limited, because a TTP/C controller with incoming link fault may prevent the cluster from a successful startup.

Timeouts

For the startup of a cluster several timeouts are of utmost importance to prevent nodes from repetitious collisions during the asynchronous bus access.

Startup Timeout

The startup timeout $\tau_i^{startup}$ of a node$_i$ must be cluster-unique for all nodes which have the permission to perform a cold start. Its value is equal to the duration of all TDMA slots prior to the sending slot of the node with the slot position i:

$$\tau_i^{startup} = \begin{cases} 0 & \text{if } i = 0 \\ \sum_{j=1}^{i} \tau_{j-1}^{slot} & \text{if } i > 0 \end{cases} \tag{9.1}$$

where τ_{j-1}^{slot} indicates the duration of the node slot of the $(j-1)^{th}$ controller.

Listen Timeout

The listen timeout (τ_i^{listen}) of a specific controller is the sum of the controller startup timeout ($\tau_i^{startup}$) and the duration of two TDMA rounds[1] (τ^{round}):

$$\tau_i^{listen} = 2 \cdot \tau^{round} + \tau_i^{startup} \tag{9.2}$$

This choice for the 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 listen timeout.

9.1.2 Integration

Integration is the process of a node to adopt the C-state of a running cluster to synchronize on and to acquire a sending slot.

After the initialization of the node is finished, the TTP/C controller behaves like during the cluster startup but should never perform a cold start: it listens on the channels to receive a frame with explicit C-state or a cold start frame and then integrates on it. The first cold start frame received during the listen timeout – the *big bang* – is rejected by a controller.

The cluster schedule must provide at least a minimum of one frame per channel with explicit C-state every two TDMA rounds (minimum listen timeout). If this is not given, an integrating controller will cold start in case of a permanent failure of one channel (incoming link error).

A controller will never integrate, if no frames with explicit C-state are sent by the rest of the cluster - this scenario must be prohibited by the schedule designer.

Big Bang

A mechanism called the *big bang* ensures 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 is not needed, but in case of long propagation delays $\Delta_{prop} > \Delta_{cold\ start\ frame\ duration}$ subsets of nodes may integrate on different cold starters. To prevent such startup cliques the first received correct cold start frame is rejected by all nodes and the listen timeout is restarted. The cold starting nodes will not detect any traffic during their TDMA round and start their startup timeouts after they get a communication blackout error.

Because the difference between any startup timeouts is at least a slot duration and therefore longer than a transmission phase or the maximum propagation delay, no further inconsistent collision will appear. Figure 9.1 on page 51 shows a startup scenario with nodes *A* and *E* as cold starter. 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.

[1]If the TDMA rounds differ in duration, the longest TDMA round is chosen

Using the big bang 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*.

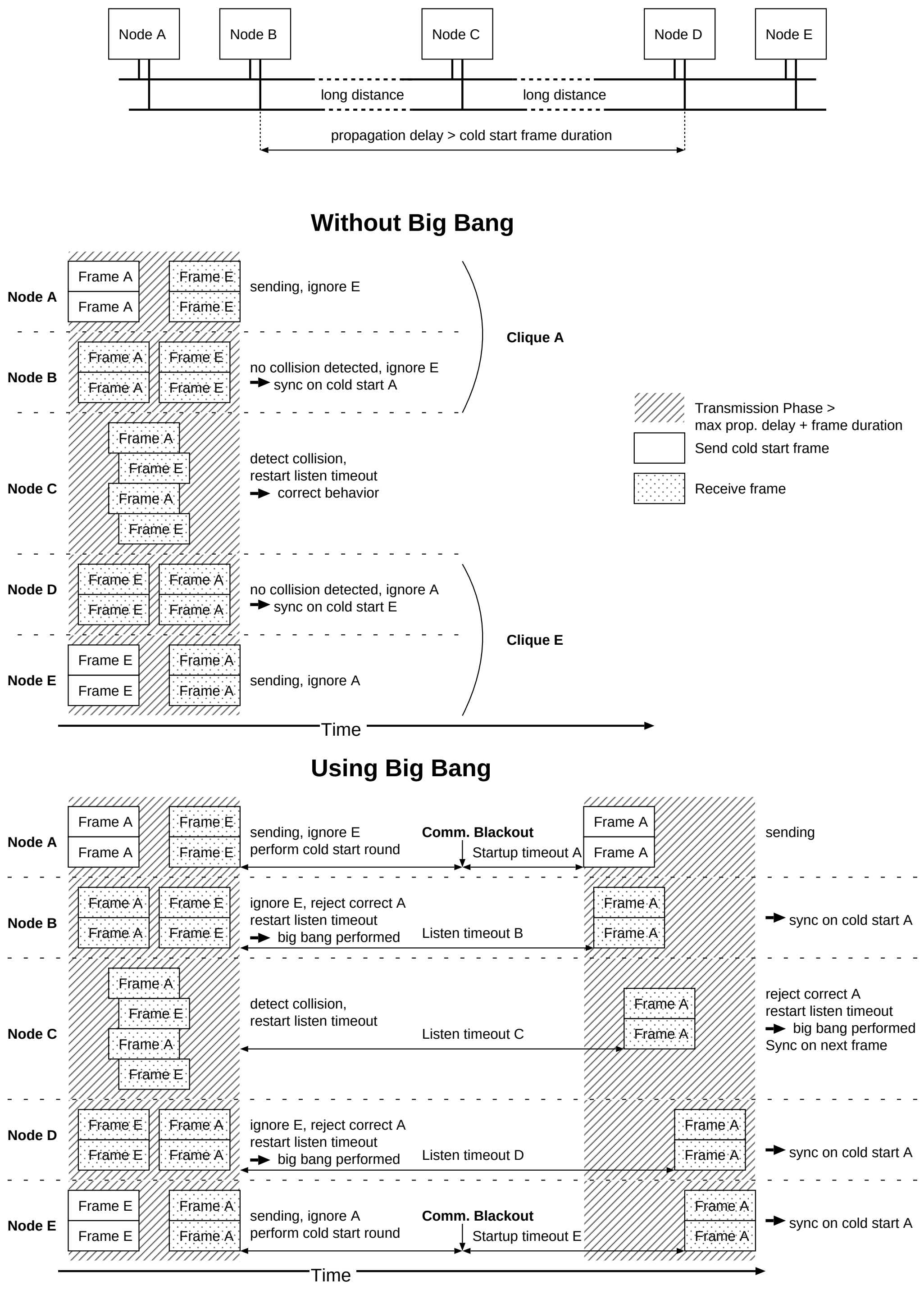

Figure 9.1: Startup Collision Scenario – Big Bang

Slot Acquirement

A node only acquires its node slot in a round if it is allowed to reintegrate in its node slot of that round, the host is active, the integration counter condition is fulfilled and the slot is not occupied:

- Different controllers sharing a node slot (shadows) must not be allowed to acquire this slot in the same round of a cluster cycle. The *Reintegration Allowed flag (RA)* in the MEDL (section 10.3.4 on page 83) is used to define if reintegration is possible for a controller in this round, so shadow nodes must have their RA flags distributed over different rounds.
- A node can only acquire a slot, if the host life-sign is updated correctly. An exception is the so called *free-shot*: a node may also acquire the slot with invalid host life-sign, if it is its first sending slot, after the C-state becomes valid and the current cluster mode is the startup mode. The slot acquirement is finished, if the node has sent successfully and is active in the membership.
- The integration counter condition is fulfilled, that means a minimum number of correct frames was received (see section 9.1.2).
- An unoccupied slot means that the memberhip flag of this slot is not set, except if a multiplexed membership is used: a node with multiplexed membership does not check the membership flag, so nodes with multiplexed membership must not have shadows!

Integration Counter

To avoid cyclic integration on a faulty frame a mechanism called *integration counter* is used: it counts the number of received correct slots after integration and only if the *minimum integration count* (see section 10.3.1 on page 80) is reached, the node is allowed to send for the first time. The frame integrated on is rated as first correct slot.

This mechanism only affects the integration on a non-cold start frame (synchronized on a cold start frame, the node is allowed to send immediatly); the minimum integration count should be set at least to the value of 2 (the behavior of older TTP/C protocols meets a value of 1).

A more detailed description of the integration is given in the protocol state overview (section 12.2 on page 103) in the listen and passive states and in figure 9.2 on the facing page.

9.1.3 Data Transport

Data transport means the distribution of application data via the TTP/C network without regarding the protocol information overhead (data used by the TTP/C protocol like the C-state) and data provided by other services like the global time base.

The host provides its data via the CNI to the TTP/C controller which is sent in the node's transmission slot. Because TTP/C is a broadcast protocol all nodes receive a transmission

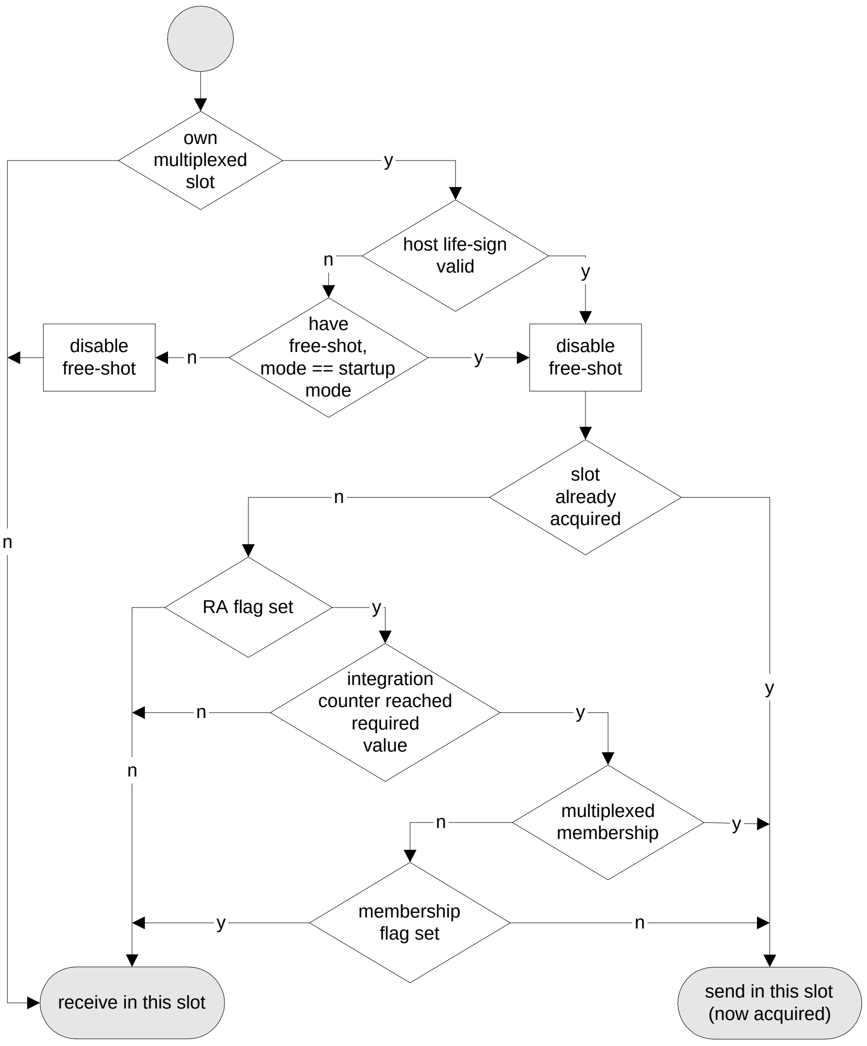

Figure 9.2: Slot Acquisition of a Node

and store the application data into the CNI if a node needs it for operation (frames can be marked in the MEDL as ‘ignore’ to save the CNI space for their receiving buffers, if their contents are valueless for a node).

Because only valid data should be transmitted the host must confirm a transmission frame buffer once before it can be sent by the controller. If the controller accesses an unconfirmed buffer, it must not transmit the frame but signalizes the host an error. This validation flag is called *Ready Status flag (RS)*. The status of a received frame should also be viewable by the host to detect errors and reject the data. The range of the frame status is described in section section 8.2 on page 41.

Non-Replicated versus Replicated Messages

It is possible that a controller sends different application data on the replicated channels during a single sending slot. Non-replicated messages will be lost in case of a single channel fault: therefore safety critical messages should be scheduled into the application data of both channels.

Concurrency Control

As described in section 5.6 on page 32 host and controller must access parts of the CNI in a mutual exclusive way. Using a time triggered operating system based on the global TTP/C time base, the host knows the intervals when it is allowed to access CNI data (synchronous access). Figure 11.2 on page 96 gives a detailed description of the intervals when the host is allowed to access the CNI without interfering the controller.

For asynchronous access other mechanisms must be implemented on the host and TTP/C controller like the *Non-blocking Write Protocol (NBW)* or buffers protected by the hardware to ensure data consistence and prevent overlapping access.

9.1.4 Global Time Base – Clock Synchronization

Distributed clock synchronization among an ensemble of clocks proceeds according to the following three distinct phases:

1. Every clock reads the time values of a well-defined ensemble of clocks.
2. Every clock calculates a correction term for its clock using a clock synchronization algorithm.
3. Every clock applies the correction term to its local clock to bring the clock into better agreement with the ensemble.

Clock Synchronization Requirements

The following list of requirements [Kop96] is satisfied by the clock synchronization system used:

- The algorithm must handle at least one asymmetric (Byzantine) fault in each TDMA round.
- The algorithm must operate in an ensemble of a varying number of nodes: if less than 4 nodes are operational, then the requirement to handle a Byzantine fault is waived.

- A set of so-called *master clocks* which is used to generate the global time must exist. These are nodes with precise resonators having a nominal maximum drift ρ better than 10^{-4} sec/sec which is an average value of the standard quartz. The nominal maximum drift required to fulfil the synchronization requirements of a specific application may differ from this value. A static rate correction must be performed on these nodes and the resonator frequency should be long-term stable. To handle Byzantine faults at least 4 master clocks must participate in the cluster.
- Some nodes may have inexpensive resonators (e.g. no long-term stabilty) and should therefore not participate in the generation of the global time (marked in the MEDL). These nodes are called *slave clocks*. It may be possible that these clocks perform dynamic rate correction.
- A node should terminate its operation if the absolute value of the correction term calculated for its local clock is greater than $\frac{\Pi}{2}$ (Π denotes the precision [Kop97a]).
- Only correct frames are used for the clock synchronization.
- External synchronization must be supported.
- It must be possible that different nodes have resonators of differing frequency.
- It is not required that a macrotick contains an integer number of microticks.

Reading the Time Values

This section describes how the start of the transmission, which occurs at the beginning of the TDMA slot (the *action time*, or AT), is used for exchanging the clock information between the nodes in the cluster. The described parameters are considered for one channel only – in an implementation both TTP/C channels must be regarded because of their different propagation delays.

The sender s starts transmission at time $t_{AT'_s}$ (from the view of the sending node), which is exactly the beginning of the action time (as perceived by the sender) plus a delay (*send delay*) that guarantees that no correctly synchronized receiver receives the transmission before the action time (as perceived by the receiver). This point in time is also called *delayed action time, AT'* of the sender.

$$t_{AT'_s} = t_{AT_s} + \Delta_{delay_s} \tag{9.3}$$

Δ_{delay_s} of the sender s must be selected so that no correctly synchronized receiver receives the transmission from the sender before its own perception of the action time.

The time difference between the expected (from the MEDL) and actual arrival time of every frame (including a controller's own frames) is measured in locally used microticks at the receiver.

If the sender is not part of the master clocks (SYF flag (section 10.3.4 on page 83) in the MEDL round slot entry not set) or the frame is not correct, the measured time difference is discarded.

In each round slot entry the MEDL contains a delay correction term $\Delta_{corr_{s,r}}$ for the transmission between sender s and the receiver r in microticks of the receiver. This delay correction term is added to the expected time of arrival of the frame.

If two correct frames from the same node (normal case) are received, the average of the two time difference values is calculated and used as measurement value. If only one correct frame from a node is received, this one value is used.

The value is stored in a push-down stack of depth four. At startup the value of the four stack entries are initialized with zero in order to prevent an unwanted drift in any direction due to uninitialized time difference values. During operation it is ensured that no invalid time difference values are contained in the stack, since old values are only discarded when new values obtained from correct frames are present (i.e., old values get pushed out of the stack by new ones).

If no correct frames were received, the push-down stack is left unchanged.

Synchronization Algorithm

The Fault-Tolerant Average (FTA) algorithm [Kop87] is used for clock synchronization in the TTP/C controller.

The slot at which a new correction term is calculated is recorded in the flag ClkSyn in the MEDL (section 10.3.4 on page 83). This ensures that all nodes correct their clocks at the same time. If the sender in this slot is part of the subset of nodes selected for synchronization (i.e., if in this slot both the SYF and the ClkSyn flags are set), the time measurement taken from that received frame is stored in the push-down stack before this calculation is performed.

The smallest and the largest measurements of the four measurements in the push-down stack are discarded.

The average of the two remaining measurements is the *clock state correction term* for the local clock.

An external rate correction term may be contained in the external rate correction field of the CNI. The external rate correction term is expressed as the number of microticks that have to be corrected in each synchronization interval.

Using an external time reference (e.g. GPS) a synchronization protocol in the host layer can be implemented: The external correction term is calculated in the time gateway host (that one with the reference clock) and sent to every host of a cluster as application data in a normal frame. The hosts will extract the correction terms from the frames and write them into the external rate correction fields of their controllers. The controllers add this term to their clock state correction term to arrive at the *total correction term* of the node for the next synchronization interval.

If the absolute value of the total correction term is larger than $\frac{\Pi}{2}$, the node raises a synchronization error and freezes.

Correcting the Local Clock

A defined amount of microticks[2] is corrected after $frMTs$ (*free-running macroticks count*) macroticks ($frMTs$ is defined in the MEDL, if it is supported by the controller).

Correction of the local clock is initiated no later than at the next action time; the duration for performing the complete clock correction depends on the total state correction term and the value of $frMTs$.

After the total correction term is exhausted, the clock runs free until the next synchronization instant marked in the MEDL.

Parameter Selection

This section deals with the constraints imposed on the different parameters of the clock synchronization. In order to provide a *reasonable timebase* [Kop97a], the precision Π has to be less than the duration of one macrotick Δ_{MT}. Both parameters are defined in the MEDL.

$$\Pi < \Delta_{MT} \tag{9.4}$$

The expected receive time $t_{AT'_r}$ (delayed action time) of the receiver r depends on the expected frame's send time $t_{AT_r} + 2\Pi$ (from the view of the receiver) and the channel dependent delay correction term $\Delta_{corr_{s,r}}$ specified in the current MEDL round slot entry:

$$t_{AT'_r} = t_{AT_r} + 2\Pi + \Delta_{corr_{s,r}} \tag{9.5}$$

Assuming the sender s transmits with a send delay of Δ_{delay_s}, the following relation between s and all receivers must be fulfilled. The propagation delay between s and any r is $\Delta_{prop_{s,r}}$, the receiver's correction term is $\Delta_{corr_{s,r}}$:

$$\begin{aligned} \forall r \in Cluster \setminus \{s\} : \Delta_{prop_{s,r}} + \Delta_{delay_s} &= \Delta_{corr_{s,r}} + 2\Pi && (9.6)\\ \Delta_{corr_{s,r}} &\geq 0 && (9.7) \end{aligned}$$

The cluster designer must take care to fulfill this conditions and may optimize the parameters depending on the propagation delays and the precision value.

The receive window Δ_{rw_r} of the receiver defines the interval of valid receptions around the expected receive time $t_{AT'_r}$ of a particular frame. This means the reception is expected in the middle of this symmetric window:

[2] As default one tick, but if the microticks-per-macrotick ratio is very large, a greater amount than one microtick may need to be corrected in order to exhaust the complete correction term before the next synchronization instant. A value greater than 1 may not be supported by all controller implementations.

$$\Delta_{rw_r} = [t_{AT'_r} - 2\Pi, t_{AT'_r} + 2\Pi] \tag{9.8}$$

The captured time difference Δ_{dif} represents the difference between the clocks of sender s and receiver r:

$$\begin{aligned} \Delta_{dif} &= t_{reception_r} - t_{AT'_r} && (9.9) \\ \Delta_{dif} &= t_{AT_s} - t_{AT_r} && (9.10) \end{aligned}$$

For a reception to be valid, it must start in Δ_{rw_r}. Receptions not started in this window are treated as invalid and the captured value is not used by the clock synchronization algorithm. The context of the parameters is shown in figure 9.3.

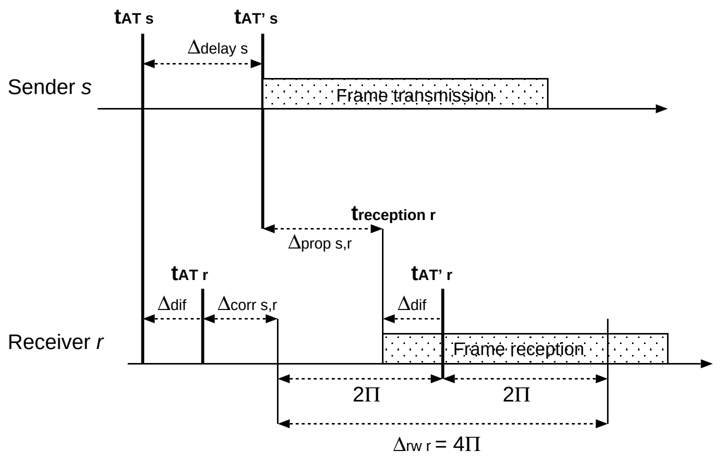

Figure 9.3: Time Difference Capturing

The interval between two successive invocations of the clock synchronization algorithm (i.e., two successive node slots with ClkSyn flag set) Δ_{cs} is bounded by the following two conditions:

$$\begin{aligned} \Delta_{cs} &\leq R_{int} && (9.11) \\ \Delta_{cs} &> maxcorr \cdot ((frMTs + 1) \cdot \Delta_{MT} + \Delta_{mt}) && (9.12) \end{aligned}$$

where R_{int} denotes the *resynchronization interval* [Kop97a], *maxcorr* specifies the maximum number of microticks that can be inserted or deleted in order to correct the controller's local clock, $frMTs$ is the number of free running macroticks taken from the MEDL and Δ_{mt} is the duration of a single microtick. Hereby the following restriction applies to the values of *maxcorr*:

$$0 \leq maxcorr \leq \frac{\Pi}{2\Delta_{mt}} \tag{9.13}$$

Considering the *synchronization condition* imposes additional constraints on the values of R_{int}:

$$\Gamma = 2\rho R_{int} \tag{9.14}$$
$$\Pi \geq \Gamma + \Phi \tag{9.15}$$

where Γ denotes the *drift offset*, Φ denotes the *convergence function* of the used clock synchronization algorithm and ρ specifies the *drift rate* of the local clock. This leads to the following condition for R_{int}:

$$0 < R_{int} \leq \frac{\Pi - \Phi}{2\rho} \tag{9.16}$$

When using the fault tolerant average clock synchronization algorithm, the value of Φ is the following:

$$\Phi = \frac{\Pi + 2\varepsilon}{2} \tag{9.17}$$

where ε specifies the *reading error* (i.e., the difference between minimum and maximum propagation delay). The reading error depends mainly on the characteristics of the physical layer (line drivers, cabling, repeaters, star couplers etc.).

This yields the following condition for R_{int}:

$$R_{int} \leq \frac{\Pi - 2\varepsilon}{4\rho} \tag{9.18}$$

Additionally there must be at least 4 slots per TDMA round where the SYF flag is set in the corresponding MEDL entry. The number of slots with the ClkSyn flag set must be greater than or equal to 1 per cluster cycle. The maximum number of slots with the ClkSyn flag set is restricted by the fact that a minimum of 4 slots with SYF flag set must be present between two successive slots with ClkSyn flag set. The clock synchronization algorithm of TTP/C is formally verified in [Pfe99].

9.1.5 Noise Tolerance

A design requirement of the TTP/C protocol implementation is the tolerance of one (permanent) noisy channel during synchronized protocol operation and startup.

Synchronized Operation

The TTP/C protocol determines the status of a slot from the more ‘positive’ status of the both channels, e.g. if on one channel an ‘invalid’ and on the other a ‘correct’ frame was received, the slot is regarded as ‘correct’. In older implementations this leads to a faulty scenario in case of a permanent noisy channel: Only a subset of the nodes in a full configured TTP/C cluster is present (cluster startup phase, or some node slots are left ‘empty’ for further extensions), so in a noise-free environment these slots are detected as ‘null-frames’ and are not used for the clique avoidance (the failed counter is not increased). The subset stays in synchronized operation.

If one channel is permanently noisy, the slot status of such an empty slot is marked as ‘invalid’ - an ‘invalid’ frame surpasses a null-frame - and the failed counter is increased in each empty slot. If the number of empty (now noisy) slots exceeds the number of ‘correct’ slots (the slots, that are acquired by nodes) the cluster will shutdown with all running nodes having a Clique Error. This behavior is not acceptible.

To avoid this scenario a simple modification of the slot status is made:

A slot with one frame status ‘null-frame’ and the other ‘invalid’ is treated as ‘null-slot’ (empty slot).

Startup and Integration

It is necessary to avoid permanent noise corrupting the startup of a TTP/C cluster.

Cold start frames are sent on both channels to shorten the duration of the startup. To avoid a noisy channel preventing controllers from integration on a frame of the working channel the following algorithm is used: Starting integration, both channels are rated equal. The first channel on which traffic is detected becomes the observed one. The controller tries to receive a frame on this channel. If a correct frame is received, the controller integrates on it (regarding the CIA flag and the big bang) and the integration is finished.

If the integration fails, the controller checks whether a complete frame with explicit C-state was received on the unobserved channel. If this frame was correct the controller integrates on it and the integration is finished.

If the integration fails or no frame was received on the unobserved channel, the controller swaps the observed channel (assuming the previous observed one is a noisy channel) and restarts the listen timeout.

If the observed channel is silent and a complete frame reception is detected on the unobserved one, the frame is checked and the controller integrates on it if the check is passed. If the check fails the controller does not change the observed channel and continues with its listen timeout. The algorithm used by TTP/C is described in figure 9.4 on the next page.

During the startup timeout after an unsuccessful cold start round only the observed channel is monitored for traffic detection to abort the timeout and to enter the listen state. If an observed channel was determined during the listen timeout, that one is used, else the channel with the first traffic detected becomes the observed one.

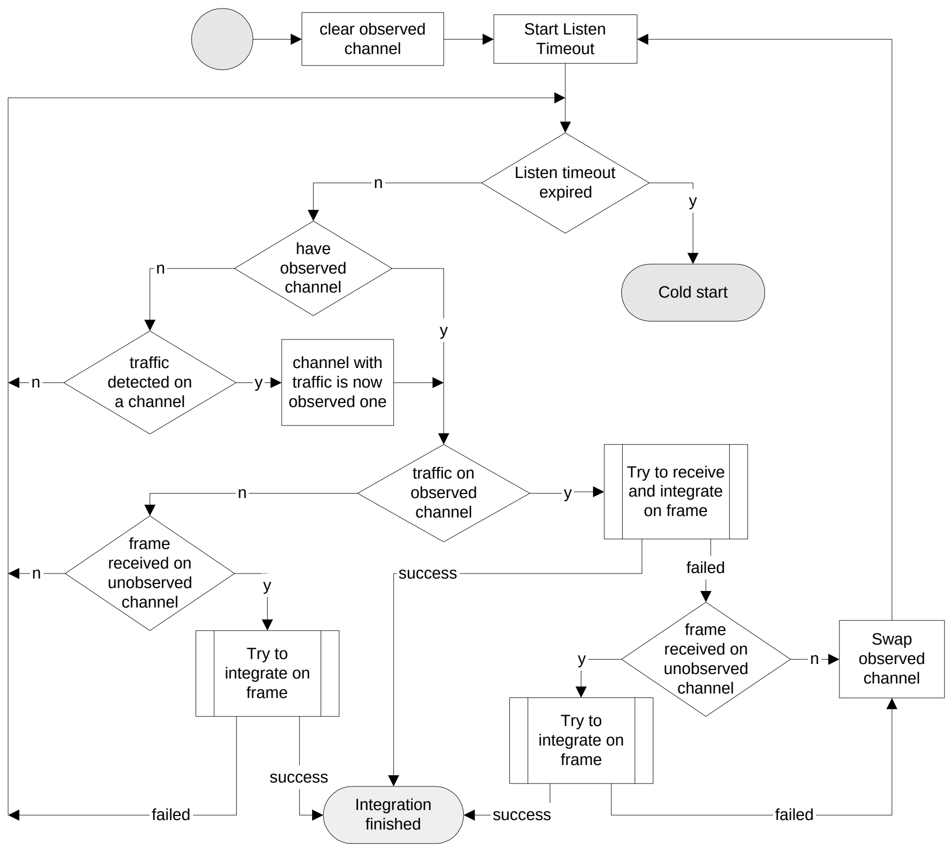
clear observed channel
Start Listen Timeout
Listen timeout expired
n
y
Cold start
have observed channel
n
y
traffic detected on a channel
n
y
channel with traffic is now observed one
traffic on observed channel
n
y
Try to receive and integrate on frame
frame received on unobserved channel
n
y
success
failed
Try to integrate on frame
frame received on unobserved channel
y
n
Swap observed channel
Try to integrate on frame
success
failed
Integration finished
success
failed

Figure 9.4: Noise Tolerance during Integration

Noise on one channel is tolerated in any case, noise on both channels will prevent the cluster from cold starting except when there is silence on one channel for at least the duration of the shortest listen timeout of a node with cold start permission. Using an intelligent central bus guardian will also tolerate incoming link faults of a node (endless cold starter, can be treated as periodic bus noise on both channels).

9.1.6 Acknowledgment

The TTP/C acknowledgment service is based on the membership service described in section 9.2.1 on page 68 (an example for the dependency between the different service groups).

Acknowledgment of the receipt of a frame by the successor of the sender is performed implicitly through the node membership in the C-state. During a sending slot a node sends two frames. If any of these two frames is received correctly by a receiver, the receiver considers the sender to be active at its membership point.

Only (syntactical) *valid* frames are analyzed. Invalid frames and null-frames are discarded.

The analysis of frames with implicit C-state is based on the CRC calculation. The analysis of frames with explicit C-state carried in the data field is different: A CRC error of a frame does not indicate a C-state disagreement between sender and receiver, but rather a disturbance of the transmission or a schedule ID mismatch between sender and receiver. If both frames of a sender were corrupted during transmission or contain incorrect C-state data, the sender loses its membership.

The first and second successors of the sender determine whether the sender has succeeded in sending at least one correct frame. The successor relationship is dynamic, depending on the membership at the time of evaluation.

The point in time when the decision about the membership of a node is final is called the *membership recognition point* related to this node.

MEMB(A) denotes the membership flag of node A at the particular controller. Before starting to send, the sending node, say node A, sets its membership flag MEMB(A) to TRUE. Therefore the reception of a correct frame from a sending node requires that the receiver sets the membership flag of the sender to TRUE before checking the frame CRC (or the C-state explicitly transmitted).

Normal Operation

A node A that decides to send assumes itself alive and fully operational.

If the node assumes itself alive and fully operational, it sets its own membership flag MEMB(A) to ‘TRUE’ and sets its agreed slots counter (see section 9.2.2 on page 69) to ‘one’.

If the successor of the sender node A, say node B, has received at least one correct frame from node A during the transmission phase of the sending slot of node A, MEMB(A) is TRUE for the transmission of B’s frame. Therefore A can use B’s transmission for acknowledgment.

A will only consider transmissions from B for acknowledgment if at least one valid frame (see section 8.2.3 on page 41) is received. If this condition is not fulfilled, MEMB(B) is set to FALSE, and the next sender C is tried as (first) successor.

During normal operation, the sender A performs two CRC checks over the frame received from its successor B, either in parallel or sequentially.

Check Ia A sets MEMB(A) to TRUE and MEMB(B) to TRUE. It then performs the CRC check over the received frame, concatenated with its local C-state.

Check Ib A sets MEMB(A) to FALSE and MEMB(B) to TRUE. Then it performs the CRC check over the received frame, concatenated with its local C-state.

If the CRC check Ia is OK, node A assumes that the transmission was correct and remains in the membership. Node A increases its agreed counter by one. The *membership failure counter* is set to zero. The decision about A's membership in this round is final.

If the frame has an explicit C-state, the C-states between sender and the receiver have to be explicitly compared. If these two C-states differ, then the same case distinction as with a C-state error of an implicit C-state has to be made.

If the CRC check Ia is not OK, then at least one failure has occurred. If both checks (check Ia and check Ib) fail, it is 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, B is considered to have transmitted and failed, and the failed slots counter is incremented (see section 9.2.2 on page 69).
- If an invalid transmission activity was observed only on one channel, and silence on the other channel, transient noise is assumed, and neither the agreed slots nor the failed slots counter is incremented (see section 9.1.5 on page 59).

Comment: *Check Ib will also fail if A transmitted 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.*

B loses its membership and A continues to look for a (first) successor.

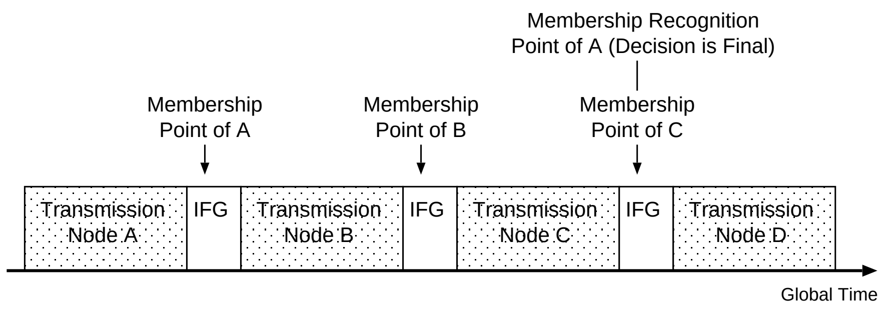

Figure 9.5: Membership Decisions

Failure Scenarios

It is possible for check Ia to fail and check Ib to pass. This occurs if either A's transmission was faulty or corrupted, or if B made some error. The data transmitted by B is 'suspicious' at this time, as it is unknown whether A or B is correct. A therefore marks the frame data from B as 'tentative' in the frame status field (see section 8.2.6 on page 42) and does not change the membership flag of B. A final decision of the cause is made after receiving the next frame from A’s second successor — which is also the successor of B —, say C. If A decides not to use B’s data anyway in this case, this decision will be correct if A is correct, which is in accordance with the confidence principle.

In the next slot, the original sender A performs two CRC checks over the valid frames received from C.

Check IIa A sets MEMB(A) to TRUE and MEMB(B) to FALSE. It then performs the CRC check over the received frame, concatenated with its local C-state.

Check IIb A sets MEMB(A) to FALSE and MEMB(B) to TRUE. It then performs the CRC check over the received frame, concatenated with its local C-state.

If the CRC of check IIa is OK, node A assumes that its original transmission was OK and successor B was in error. Node A remains in the membership. Node A increases the agreed slots counter and the failed slots counter by one. The membership failure counter is set to zero. The decision is final in this round.

If the CRC of check IIb is OK, node A assumes that the original transmission was erroneous and the successor B was correct. The membership flag of node A is set to zero, the membership flag of node B is set to one. A increases the agreed slots counter and the failed slots counter by one and marks the frames from C as correct. The membership failure counter is increased by one; if it reaches the maximum membership failure count value set in the MEDL, the controller raises a membership error (ME) and freezes. Otherwise the controller only raises a membership loss interrupt (ML) and looses its slot acquirement. The decision is final for this round.

Comment: *The frame status fields for the frames from node B are not changed to ‘correct’ at this time. Even so, the host CPU can decide that B was correct.*

If neither check IIa nor IIb worked, or if C transmitted null-frames, then C is removed from the membership, the failed slots counter is increased by one, and the next node becomes the second successor.

Figure 9.6 on page 66 illustrates the temporal sequence of the different CRC checks performed.

Table 9.1 on the next page summarizes the cases that have to be distinguished.

Sequence of checks

If Check Ia succeeds for the frame on at least one channel, the result of Check Ib is not regarded anymore for this frame and the frame on the other channel. In the same manner, if Check IIa succeeds for the frame on at least one channel, the result of Check IIb will not be used by the controller. In this way, a contradiction between the channels is solved in accordance with the confidence principle.

	CRC of Check				Action							
	Ia (B)	Ib (B)	IIa (C)	IIb (C)	MEMB (A)	MEMB (B)	MEMB (C)	Agreed slots	Failed slots	Other	Comments	Decision after
a	T				T	T		+1	0		A acknowledged	1st succ
b	F	F			T	F		0	+1	do check I with next successor	Frames corrupted	
c	F	T	T		T	F	T	+1	+1		1st succ failed	2nd succ
d	F	T	F	T	F	T	T	+1	+1	slot acquirement lost	A failed	2nd succ
e	F	T	F	F	T	F	F	0	+1	do check II with next 2nd successor	Frames corrupted	

Table 9.1: Actions after the CRC Check

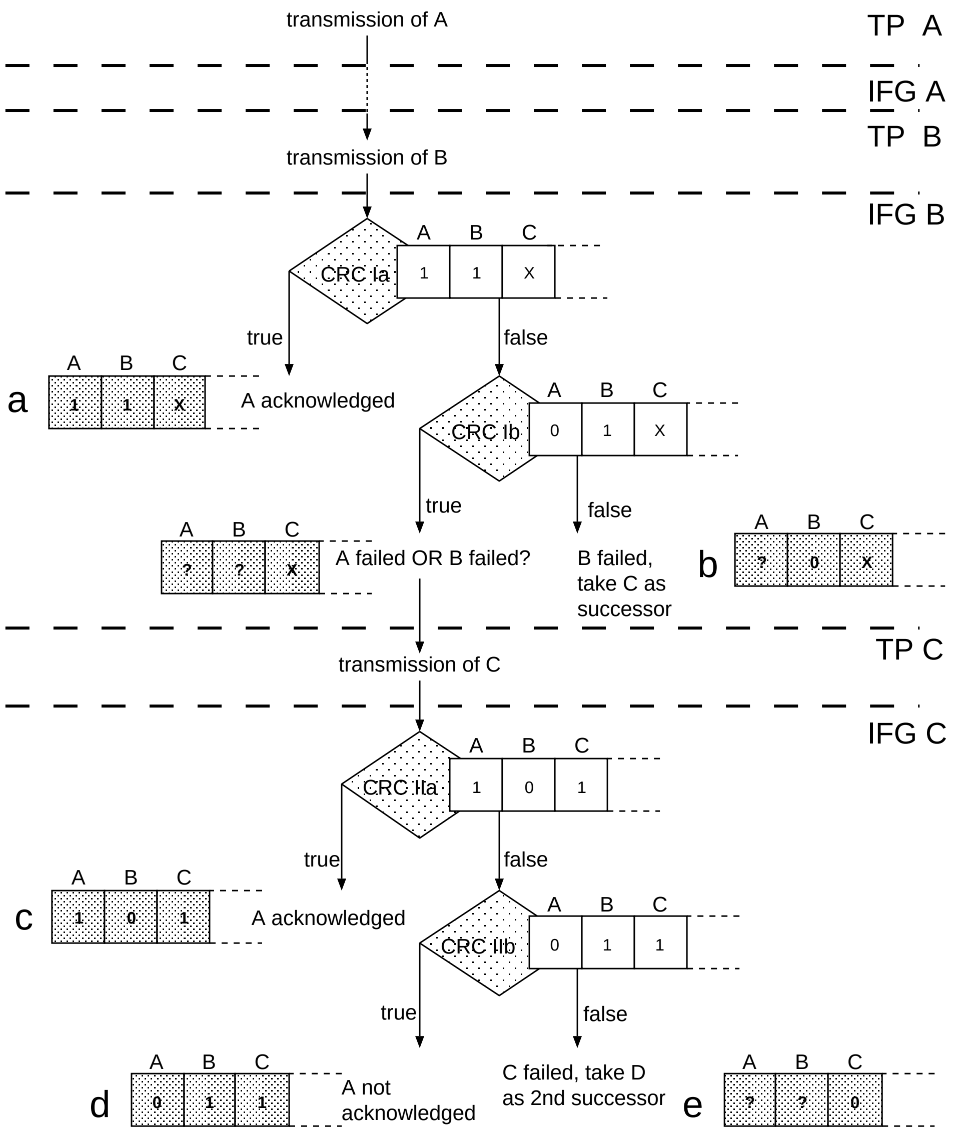

Figure 9.6: Acknowledgment Scheme

Repeated faults

If a node loses its membership in *n* (*maximum membership failure count (MMFC)*) successive sending slots, then the controller terminates its operation by raising a membership protocol error and freezing the controller.

The MMFC is defined in the MEDL. If the MMFC is set to zero, this condition is not checked.

If a controller reaches the pre-send phase (PSP) preceding its next sending slot *before* it is acknowledged, i.e., while it is still looking for a first or second successor, the clique detection mechanism (section 9.2.2 on page 69) or the check for communication blackout will detect an error.

9.2 Safety Services

9.2.1 Membership

The node membership service (shortly called *membership service*) informs all nodes of a cluster about the operational state of each node within a latency of about one TDMA round. The membership service of TTP/C is formally verified in [Pfe00].

A node is operational at time *now* if

- the host computer of the node has updated its life-sign within the last TDMA round and
- the communication controller is operating and synchronized with the rest of the cluster.

If the host computer has not updated the life-sign, the host is considered non-operational and the communication controller does not send a frame. The controller remains synchronized by receiving all frames, but does not send.

The Membership Vector

A node becomes a member node after it has acquired a node slot.

The membership status of each *member node* is recorded in the membership vector. The membership vector can be implemented as a vector of flags with a length that is equal to the number of member nodes of a cluster. Each member node is assigned a membership flag within the membership vector. If the membership flag in the membership vector is set, then the corresponding member node is considered to be alive, otherwise it is considered not to be alive.

The Membership Point

The post-receive phase (PRP) after the transmission phase of the sending slot of a member node is the membership point of the member node. After a final decision about the membership status of a member node has been reached at the *membership recognition point* (see section 9.1.6 on page 61), the membership flag remains unchanged until the next membership point. The membership recognition point is the moment when an observing node makes a final decision about the membership of a sending node. In a failure scenario, the membership recognition points at different observing nodes can be different.

In the best case, the membership recognition point of a node A is right after the PRP of the node slot of A's successor. In the worst case, the membership recognition point is postponed until the PSP preceding A's next sending slot — in this case, node A failed to find acknowledgment.

Due to the *confidence principle*, a sending node considers itself active as soon as all requirements for transmission in the next transmission phase are met. The node internally reaches its own membership point one node slot earlier than the rest of the cluster. A sending node sends two frames immediately before its (global) membership point, one frame on each of the replicated channels.

A receiving node considers a sending node alive at the membership point of the sending node if at least one of the two frames sent immediately before the membership point of the member node has arrived correctly at the receiver. If none of the two frames of a member node has arrived at the receiver immediately before the membership point, the receiver considers the sending member node as not alive.

Multiplexed Nodes

A multiplexed node can either have a multiplexed membership flag or its own membership flag, depending on the MEDL. The meaning of a multiplexed membership flag can change with every TDMA round. Each safety critical multiplexed node should have its own membership flag.

From the membership point of view, a multiplexed node with its own membership flag 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 non-multiplexed nodes.

Permanent Passive Nodes

Because permanent passive nodes are not allowed to send, they do not own a membership flag.

9.2.2 Clique Detection

To avoid permanent clique formation and to detect inconsistencies in the cluster as fast as possible, the number of nodes that agree on the current C-state is monitored during every TDMA round. For the evaluation the slot status (see section 8.4 on page 45) is checked:

- If the slot status is ‘correct’, the *agreed slots counter* is incremented.
- If the slot status is ‘incorrect’ or ‘invalid’, the *failed slots counter* is incremented.
- If the slot status is ‘null frame’, the slot is not considered for clique detection.
- After having sent successfully the agreed slots counter is set to 1.
- If the acknowledgment is under progress and the first or second successor is evaluated, the counters are not changed, but updated after the acknowledgment is finished according to table 9.1 on page 65.

During integration the agreed slots counter is initialized with 2, the failed slots counter is reset to 0.

Clique Error

Once per TDMA round, in the PSP before its own (or shared) sending slot, the controller checks whether the value of the agreed slots counter is greater than the failed slots counter. If this condition is not fulfilled, i.e., the node detects that it is not consistent with the majority of the nodes, a *clique error* (section 20 on page 88) is raised and the controller freezes.

Communication System Blackout

If the sum of agreed slots counter and failed slots counter is less or equal than one — which indicates that no correct transmission activity, except possibly for the node's own, has been observed during the last TDMA round — the TTP/C controller raises the *communication system blackout* error (section 20 on page 88) and freezes. This condition is checked once per TDMA round, in the pre-send phase prior to the controller's (own or shared) sending slot.

9.2.3 Host/Controller Life-sign

The host must provide the controller with a life-sign that must be periodically updated in order to notify its activity. The specific life-sign protocol is implementation dependent.

The update of the host life-sign has to be performed between the transmission phase of the node's sending slot and the beginning of the pre-send phase preceding the node's next sending slot. It is verified by the controller in the PSP of its sending node slot. If all TDMA rounds of a cluster cycle have the same length, the update period equals the length of a TDMA round. In order to provide an accurate life-sign information, the host should update the host life-sign field as close to the next transmission phase as possible.

If the host life-sign does not agree with the expected value, the controller does not send and switches temporary to passive: it remains synchronized and receives but will not send until the life-sign is updated correctly. An updated host life-sign is also a pre-condition for a controller to startup a cluster by sending a cold start frame.

The controller provides a life-sign to the host by the changes in the C-state global time and schedule position.

9.2.4 Bus Guardian

The bus guardian is an autonomous subsystem that protects the communication channels from temporal transmission failures. Additionally, the bus guardian can serve to protect the bus against transmissions that are apt to lead to ambiguous results at the receivers (so-called 'slightly-off-specification' faults).

The bus guardian functionality may be implemented locally or centrally:

A local bus guardian may reside on the same silicon die as the TTP/C protocol controller; it can also be implemented as an independent device. In any case, a node must have a bus guardian to achieve fail-silence in the temporal domain. Such a local bus guardian has only the knowledge of its node‘s schedule (sending slot) and must use an independent external clock source.

A central bus guardian that also protects the cluster against additional fault classes[3] is a device with complete knowledge of the system communication and can be effectively implemented in the star coupler of a star network – nodes participating in a bus architecture must use a local bus guardian.

Figure 9.7 shows the layout of an architecture with local and central bus guardian. Due to the single fault hypothesis a central bus guardian must consists of two independent bus guardians for the two TTP/C channels while a local bus guardian can process both channels.

Because of the extended fault protection, fail-operational systems should use a central bus guardian with autonomous clock synchronization.

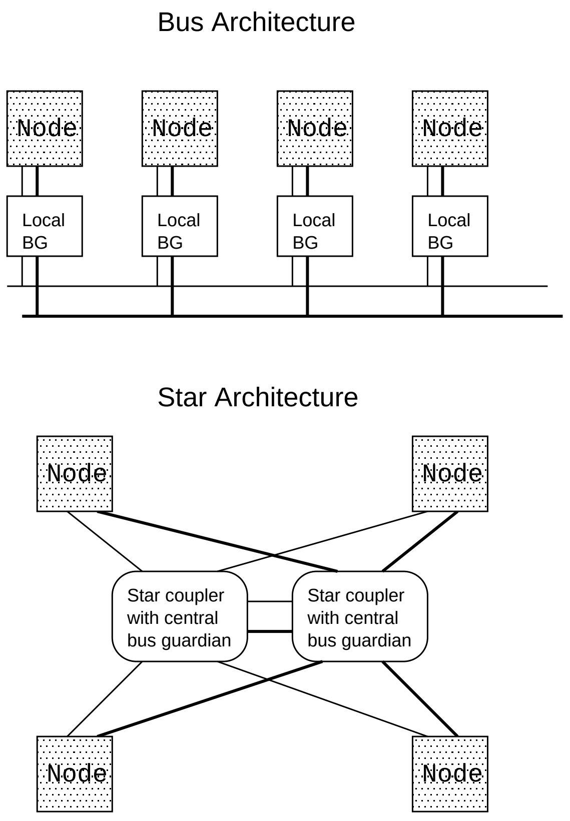

Figure 9.7: Central versus Local Bus Guardian

[3]especially spatial proximity faults and permanent slightly-off-specification faults

Requirements

This section presents the basic requirements for a bus guardian unit.

Babbling idiot avoidance The bus guardian must protect the bus from monopolization by a babbling idiot (i.e., a node sending more often than specified or even all the time).

Fail-silence The bus guardian has to protect the bus from a transmission of the controller outside of its sending slot, enforcing fail-silence of the controller in the temporal domain.

Common mode failure avoidance The bus guardian must use a clock source that is independent from the oscillator of the controller, preventing temporal coupling of the two subsystems, and thus a common mode failure[4].

Unless the bus guardian performs autonomous clock synchronization, some kind of periodic synchronization signal from the controller to the bus guardian, commonly referred to as *Arm signal*, is required to keep the bus guardian window (an example is shown in figure 9.8 on the next page) synchronized with the cluster communication. The location of the Arm signal in reference to the own sending slot can be static (in the controller design) or configurable (in the MEDL). In this case the bus guardian uses the clock synchronization from the controller and may not prevent the controller from sending outside its slot but from sending more than once in the TDMA round. Such a bus guardian is not fully independent although it has its own external clock source.

Continuous function The bus guardian must be operative during all controller states – it is not sufficient to have a bus guardian only in the active states.

Error detection In case of a local bus guardian the bus guardian should be able to immediately tell the controller when it detects an error. If the controller does not get an error signal from the bus guardian in this case, the protocol mechanisms will still tell the controller that it has failed, because the other controllers will not receive any frames from it – this may take much longer to detect and report to the host, but is the only possibility in a star architecture with a central bus guardian.

Slightly-Off-Specification Fault Prevention Permanent faults that occur within the defined sending slot of a node but lead to clique formation in the cluster can only be contained by an ‘intelligent’ unit that prevents such transmissions from propagating into the network. A bus guardian unit that meets this requirement offers protection against such ‘slightly-off-specification’ faults[5].

[4]If the bus guardian and the controller share a clock, the worst case fault would be a sudden drop of the clock speed during transmission, creating a huge transmission phase that is accepted by the bus guardian. Such a fault results in a de-facto babbling idiot.

[5]The faults have this name because typically they manifest themselves in transmissions that are just slightly out of the specified timing/voltage level etc. and therefore are ‘just barely’ accepted by some nodes and ‘just barely’ rejected by some others.

Bus Guardian Window

The bus guardian should protect the bus from any transmission outside of the defined node slot, which could invalidate a correct transmission. However, the bus guardian must not prevent the controller from transmitting at correct times.

To achieve the described functionality, the bus guardian should allow transmission for the node at the right time and for the complete frame transmission (*slot duration*), but must prevent any transmission from this node for the remaining duration of the TDMA round.

The resulting round pattern contains a so-called bus guardian window for the transmission of the node.

To allow for the inevitable differences in the independent timing of the bus guardian and the sending controller, the bus guardian has to open the window 'a little earlier' than the expected start of transmission, and close it only 'a little later' than the expected end of transmission. The amount of time required to ensure that the controller is always permitted to send in the fault-free case depends largely on the synchronization mechanisms used by the bus guardian, and can be a determining factor for the inter-frame gap.

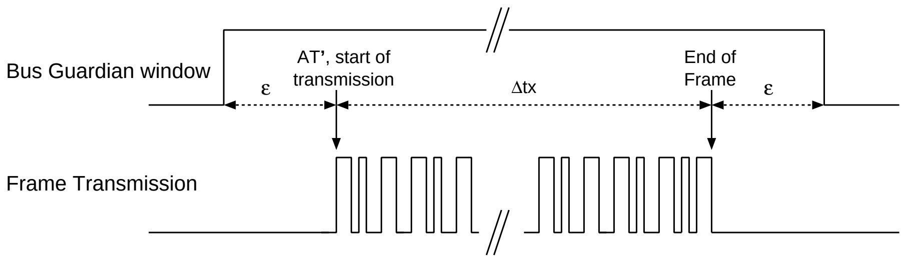

Figure 9.8: Bus Guardian Window Timing

The lower limit for the parameter ε in figure 9.8 can be derived from the physical properties of the system:

Local bus guardian without autonomous clock synchronization It must allow the maximum possible drift between controller clock and bus guardian clock. It must also take into account the maximum clock state correction that can occur between the *Arm* signal (i.e., the last synchronization point between controller and bus guardian) and the closing of the window. A bus guardian without autonomous clock synchronization cannot prevent the controller from sending outside its TDMA slot but can only limit the bus access for the slot duration.

Bus guardian with autonomous clock synchronization The maximum deviation between the controller and the bus guardian is limited by the precision of the clock synchronization. A lower limit for ε can be deduced from the following requirement: The bus guardian must protect the bus from a node transmitting a correct frame at a point in time close to the allowed precision; such a transmission can result in a Byzantine fault if the frame is received correctly by some nodes – because for them it started within the allowed receive window –, but be rejected as invalid by others – because for them it started outside of the receive window.

In order to prevent this inconsistency, the bus guardian must limit the times of frame transmission to an interval so that frames transmitted too early or too late are discarded and result in an invalid transmission for *all* receivers. With a transmission duration of Δ_{tx} and the transmission start at AT', the bus guardian window must not start after $AT' - \Pi$ and must not end earlier than $AT' + \Delta_{tx} + \Pi$. If ε is greater than Π, the receive window in the receiving nodes (described in section 11 on page 57) needs to be expanded so that it starts at least Π before the bus guardian opens. Further investigations about the bus guardian window in relation to the transmission timing can be found in [Rus01].

9.3 Higher Level Services

These services provides extra functionality to the host application and are not necessary for a working basic TTP/C communication.

9.3.1 Cluster Modes

Many real-time systems exhibit mutually exclusive phases of operation and control. For example, an aircraft can be on the ground, in take-off, or landing. We call every one of these mutually exclusive phases a *mode*.

Every mode has its own schedule parameters in the MEDL but TTP/C requires that all modes are based on the same TDMA slot sequence. All synchronized nodes of a TTP/C cluster must operate at the same time in the same mode; this condition is checked by the C-state agreement. Therefore a TTP/C mode is called *cluster mode*.

If the operational environment (host application) changes from one mode mode to another one, the TTP/C controller may change from one cluster mode to another cluster mode – this is called *mode change*. A cluster schedule must contain at least one cluster mode, called the *startup mode*, which is used during the cluster startup.

A mode change is requested by the host computer via the CNI. In the PSP of its sending slot the TTP/C controller checks whether this mode change request is valid by checking if the requested successor mode is enabled in the *Mode Change Permissions (MCP)* located in the schedule information of the sending slot. After the request is read from the controller, it is cleared in the CNI regardless of whether the request was permitted or not.

If the mode change request is valid, the controller sets the mode change requests in its frame and distributes the information to all other nodes in the cluster.

During the PRP all controllers check whether a mode change was requested by the previous transmission. If the frame with the request set is valid and the request is permitted by the MCPs of the receivers MEDL, the mode change request is valid and stored in the C-state‘s DMC field until the beginning of the next cluster cycle. If another cluster mode is requested before the first one is executed, the new request overwrites the old one.

A frame carrying an invalid mode change request is marked with ‘other error’ in the frame status and considered as incorrect.

Cluster mode changes are performed not immediatly but at the start of a new cluster cycle and are therefore called *Deferred Mode Changes*. TTP/C also supports a special mode change request (*Clear Pending Mode change (CPM)*) that clears a pending mode change from the C-state‘s DMC field and prevents the cluster from changing its cluster mode at the next cluster cycle start.

9.3.2 External Clock Synchronization

External clock synchronization is in the responsibility of the host. For this purpose the external rate correction term is provided by the host via the CNI. The correction term is given in units of microticks per resynchronization interval which is added to the correction term of the TTP/C internal clock synchronization algorithm. The external clock correction value is bounded by the $\frac{\Pi}{2}$ interval.

At the start of the resynchronization interval (i.e., the action time of each slot with the ClkSyn flag in the MEDL set) the number contained in this field is added to the clock state correction term calculated by the clock synchronization algorithm, giving the total clock state correction term. The total clock state correction term yielded by this computation is bounded by the $\frac{\Pi}{2}$ interval (section 9.1.4 on page 56).

The controller clears the term in the CNI after reading it, requiring the host processor to reprogram the current rate correction value at each resynchronization interval.

To enable external clock correction, the MEDL has to be designed accordingly, i.e., the *Allow External Rate Correction flag (AR)* must be set in the MEDL.

When using external clock correction, the time gateway node – a node which has access to the cluster time and to the external time and can compare the drift rate of these two time bases – periodically reevaluates the drift and broadcasts the result to all nodes. Each node writes the drift correction value into the local controller 'external rate correction field', so that the cluster drift is modified by this amount. This requires some (small) coordinated effort in the application and some additional bandwidth for the periodic broadcast of the current drift correction term.

For fault-tolerant external clock synchronization, redundant time gateways are required.

9.4 Maintenance Services

Although not part of the core TTP/C protocol, controller implementations may support maintenance services, like built-in self test or asynchronous bus download. This protocol specification provides a transition into the download service by receiving download frames during the integration phase in the listen timeout. These frames must differ from normal TTP/C frames and should be recognized from all controller implementations that are compatible to a defined data link layer.

Comment: *Transmitting download frames from a service node will prevent a whole cluster from its regular startup and/or may prevent single nodes from re-integration. Therefore no node participating in a regular cluster should be able to generate download frames*

It is recommended to implement maintenance by using the synchronized communication, e.g. by performing a mode change into a dedicated service mode using the TTP/C mode change service.

10 Message Descriptor List

10.1 Cluster Design versus Personalized MEDL

The cluster design is a data model describing the communication pattern of a cluster. It is also known as the ‘abstract MEDL’. The cluster design defines the TDMA round layout, cluster modes and the valid changes between them, and the parameters required for clock synchronization. It does not contain node-local information about the application data storage in the CNI of individual nodes.

Each controller must have a personalized MEDL, which is derived from the cluster design, and additionally contains node-local information; furthermore, it may contain special setup data required for internal purposes of specific TTP/C controller implementations. Personalized MEDLs of individual controllers of the same cluster may differ in any or all of the node-local aspects; however, they must not differ in any cluster-wide aspect. Otherwise, a basic assumption of the protocol fault hypothesis [1] (see section 4.9 on page 25) is violated.

10.2 Overview of the Structure of the MEDL

This section presents the functionality that a personalized MEDL offers – it does *not* describe the actual implementation in terms of field layouts, field sizes, etc. that a controller expects — this layout is not defined by the protocol specification but by the controller specification. Therefore binary MEDLs may not be interchangeable between different makes of controllers.

Since online checking of MEDL semantics[2] is not required, the construction of personalized MEDLs from a cluster design requires to be correct in all steps; however, CRC fields or similar error detection mechanisms allow for detection of hardware failures in the MEDL memory once the MEDL has been correctly stored.

Online checking of the syntactic correctness of the personalized MEDL data within the controller, provided by CRC fields or similar mechanisms, must be implemented in each controller to ensure error detection.

[1]No design faults in the controller or the MEDL

[2]Such as that the layout of each TDMA round is the same for all rounds in the cluster cycle, or that the sum of all slot lengths indeed equals the explicitly stored length of the TDMA round

10.2.1 Size of the MEDL structures

The size of the MEDL memory is not specified. A cluster design is therefore limited by the amount of memory that each controller offers for storage of the MEDL data structures, and by the efficiency of these structures. A general-purpose controller offers sufficient MEDL memory for schedules with dozens of nodes and several cluster modes, while low-cost implementations may offer only just enough MEDL memory to support one application-specific setup.

10.3 Contents of the MEDL

10.3.1 Schedule Parameters

These parameters describe the basic communication behavior of a node and are necessary to integrate or startup a cluster.

Cold Start Allowed Flag (CF) If this flag is set the node can enter the cold start state. Only non-multiplexed full-member nodes that send frames with explicit C-state in the first cluster mode are allowed to perform a cold start.

Cold Start Integration Allowed Flag (CIA) The controller may only integrate on a cold start frame if this flag is set. Otherwise the controller must discard the frame and wait for an regular frame with explicit C-state. Only non-multiplexed full-member nodes that send frames with explicit C-state in the first cluster mode are allowed to integrate on a cold start frame.

Allow External Rate Correction Flag (AR) This flag enables or disables external rate correction performed by the host. If it is not set, the host cannot influence the clock synchronization of the controller.

Minimum Integration Count (MIC) A controller needs a frame with explicit C-state from another node to integrate into a cluster. To ensure that controllers do not integrate on faulty frames and re-transmit the faulty C-state in their own subsequent transmissions, a specific number of correct frames (only the first one has to have an explicit C-state) has to be received before a node can transmit actively. This number is defined with a minimum of 1 and a recommended value of 2 in this field.

Maximum Cold Start Entry This field specifies the maximum number of cold start attempts by the controller (i.e., the maximum number of attempts to send a cold start frame). If the field is set to 0, the controller will not check the number of cold start frames and has an unlimited number of retries for cold-start.

Maximum Membership Failure Count (MMFC) This field specifies the number of successive membership failures at which the controller freezes. If the field is set to 0, the controller will not check the number of membership failures and thus it will never turn itself off due to successive membership failures.

Macrotick Generation Parameters Parameters that describe the generation of a macrotick of a defined length from the microtick source. All nodes in a cluster must use the same macrotick length.

Precision The Π parameter of the clock synchronization (see section 9.1.4 on page 54) typical in units of microticks. This parameter influences the detection of synchronization errors of the local controller and other nodes and must be selected in accordance with the physical properties of the cluster to allow for correct operation under normal conditions and the reliable detection of errors. If the value selected for this parameter is smaller than the actual precision of the current cluster setup, protocol operation is not possible. An upper limit for this parameter is given by the macrotick duration for generation of a reasonable timebase.

Communication Bit Rates Used to define the bit rates on the two channels needed for transmission and reception in the cluster; the contents depend on the bit rates and their setup supported by the controller.

Startup Timeout Startup timeout for the node (see section 9.1.1 on page 48).

Listen Timeout Listen timeout for the node (see section 9.1.1 on page 49).

10.3.2 Role Section

The information of the role section associates a node with a specific sending slot. The *logical name* identifies the node slot in which the node is allowed to send and to check the clique avoidance algorithm.

Logical name slot position The slot position part of the logical name field indicates the position of the – own or shared – sending slot that corresponds to the specific role within the TDMA round[3].

Logical name multiplexed ID The multiplex ID part of the field is used to distinguish between several multiplexed nodes which share the slot determined by the slot position part. Multiplexed nodes do not transmit every round — the rounds in which they transmit are determined by the round slot entries and may therefore differ for different cluster modes — but they do perform the checks determined by the slot position part in every round, regardless of the cluster mode.

If the logical name specifies a non-multiplexed role, the multiplex ID part of the logical name is set to zero.

Passive Flag Marks the node as permanent passive (i.e., a role without sending slot) , used by a monitoring/diagnosis node.

Multiplexed Membership Flag (MM) Marks the node as a membership sharing multiplexed node. Nodes without their own membership flag are not intented for safety critical tasks.

Flag Position in Membership Vector Own flag position in the membership vector.

Send Delays Send delays of the channels described in section 9.1.4 on page 55.

10.3.3 Identification Section

The MEDL Identification Section identifies the MEDL (Schedule design, CNI layout) to provide external tools and the host with this information.

[3]This determines when the checks for clique detection etc. are performed.

Cluster Schedule Identification This field contains an identifier of the abstract MEDL, i.e. the cluster design. The cluster designer (tool) has to ensure that the schedule IDs differ for different designs. The contents are used for the frame CRC calculation (see section 8.1 on page 39) to guarantee that only nodes based on the same cluster design can participate in the same cluster.

Application Identification Unique ID used by the host application to verify the actual MEDL is compatible to it. Although the application ID is not used by the TTP/C protocol, it ensures a correct cooperation between host and controller. The application ID may consist of several subparts (e.g. CNI layout checksum) that are generated by the cluster schedule and node design tools.

10.3.4 Round Slot Section

A cluster cycle is a sequence of round slots. Each round slot is assigned to a node slot, but parameters of round slots can vary in different TDMA rounds. For each cluster mode a cluster cycle description (list of successive round slots) must exist. An example of a MEDL layout structure is given in figure 10.1 on the next page.

The following parameters must be defined for each round slot:

Logical Sender Slot Position This field contains the slot position part of the logical name of the sending node. If the slot position of the current logical name of the controller matches the value contained in this entry, then a non-multiplexed controller is the sender in the current slot; a controller sharing this slot with other nodes must additionally have a multiplex ID equal to the multiplex ID stored in the current round slot entry. If the controller is not the sender in this slot, it is a receiver.

The value of this field represents the number of the slot in the TDMA round (zero based) and can be calculated as the round slot modulo the number of slots in the TDMA round.

Logical Sender Multiplex ID This field holds the multiplex ID part of the logical sender name in this slot. A controller sends in this slot only if the logical sender slot position entry *and* this field match its current logical name.

Slot Duration The slot duration entry contains the duration of the current node slot in units of macroticks. Slot durations are chosen to fit the timing requirements of the application. At least the transmission phase, the pre-send phase and post-receive phase must be processed within the defined slot duration.

Transmission Phase Duration Duration of the transmission phase in the node slot. The length of the transmission phase includes the maximum transmission duration of both channels with jitter and delay.

Delay Correction Terms Propagation delays of frame transmission between this node and the sending node on the respective channels in units of microticks. If the node is sending, the values are ignored and the send delays from the protocol parameters are used to delay the transmissions on the two channels.

The combination of the delays (send delay and delay correction term) of the receiving and transmitting nodes will amount to the delay correction of this virtual point-to-point connection.

Frame Types Marks the type of the frames transmitted in the round slot (e.g. with implicit or explicit C-state)

CNI Addresses and Length of Application Data Defines the Source/Destination address and the length of the application data transmitted in the slot. May also be used to mark data as ‘ignore’, if received data is not needed.

Mode Change Permissions (MCP) Permission for distribution/acception of a mode change request to a specified cluster mode.

Reintegration Allowed Flag (RA) Marks that the node is allowed to acquire a round slot, needed in case of shadows to avoid concurrent slot acquirement.

Clock Synchronization (ClkSyn) Flag If this flag is set, the clock synchronization algorithm is executed in this slot. It marks the resynchronization intervals for the cluster-wide clock synchronization.

Synchronization Frame (SYF) Flag If the synchronization frame flag is set, the frames arriving in the slot are used by the clock synchronization algorithm: for these frames the deviation between the expected and the actual arrival time of the frame is measured and used for calculating the clock correction term.

Flag Position in Membership Vector This entry contains the flag position of the logical sender in the membership vector.

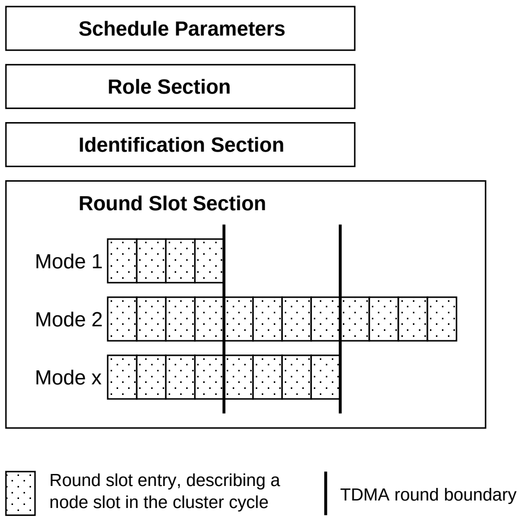

Figure 10.1: MEDL Layout Example

10.4 MEDL Design Guidelines

When designing a MEDL for an application, the following constraints have to be ensured by the design:

- The schedule ID field must contain identical values for all controllers in the cluster. This is achieved by deriving the personalized MEDLs of all controllers from a single cluster design.
- For fault-tolerant re-integration, there must be at least two nodes which transmit frames with explicit C-state in each cluster cycle.
- In startup mode, as many nodes as possible must send frames with explicit C-state; the only valid reason for transmission of implicit C-states in startup mode is to reduce the slot length (see calculation of slot duration below).
- A minimum of four nodes per TDMA round is recommended in order to provide fault-tolerant protocol operation.
- The minimum integration count should be set at least to 2 to prevent the cluster from successive integration faults.
- The startup timeout parameter must be unique for all nodes in the cluster which have the permission to perform cold start to avoid multiple collisions during cluster startup.
- If all TDMA rounds have the same slot lengths, the duration of each slot must be at least the time required to transmit
  - the longest frame transmitted by any node in this slot (regarding jitter and propagation delays) in any cluster mode
  - plus the longest computation time (PRP+PSP) of any controller in the cluster (i.e., the maximum IFG of the slowest controller in the cluster),
  - plus the duration of one precision interval (Π),

  rounded up to full macrotick length. If a slot is shorter than necessary to transmit a frame with explicit C-state, nodes sending in this slot have to send frames with an implicit one even in startup mode.
- The Π (precision) parameter must be smaller than the macrotick duration in order to provide a reasonable time base with macrotick granularity, but must be large enough to allow for the physical properties of the cluster (clock drift and propagation delay jitter).
- If a cluster is designed for which not all member nodes are going to be physically present all the time, the sending slots of these nodes should not be used for clock synchronization (i.e., the SYF flag should not be set in their sending slots).
- The ClkSyn flag must be set so that in any resynchronization interval (which is determined by the slots with ClkSyn flag set) at least four slots with the SYF (synchronization flag) set are present. If a slot is intentionally left empty for future expansion or is intended for a node that is not present all the time, the SYF must not be set for this slot. The calculation of the resynchronization interval is described in section 11 on page 57.

Comment: *As a rule of thumb, the ClkSyn flag should be set at least every eight SYF slots, or once per TDMA round if there are less than eight SYF slots in a TDMA round.*

- The mode change permissions for each slot must be set so that a mode change request to an undefined cluster mode is prohibited. Furthermore, the Mode Change Permissions for all slots must be consistent within the cluster.
- If one node slot can be occupied by different controllers (i.e., it is in the redundancy scope of different controllers), it has to be ensured that only one of them is allowed to *acquire* this slot at a time by setting the RA flag in only one of the corresponding control section entries. This restriction limits the number of shadow nodes for any TDMA slot to the number of rounds in the cluster cycle.
- The flag position in the membership vector of real member nodes must be the same in all TDMA rounds in all cluster modes. For multiplexed nodes with their own membership, the flag position in the membership vector must be associated with the multiplex ID of each node. Multiplexed nodes with mixed membership must have the same flag position in the membership vector throughout all TDMA rounds in all cluster modes.
- Multiplexed nodes with shared membership flag cannot decide whether their slot is occupied upon integration; therefore such nodes must not have shadows.
- There must be at least one node that is capable and allowed (by the MEDL) to cold start in order to start up cluster communication.

  It is recommended to plan the cold start scenario in a way that other nodes (without cold-start permission) are already waiting and can synchronize on the cold-starting node immediately.

  It is recommended to limit the cold start entries because a node with incoming link error[4] may prevent a cluster startup if cold-starting is repeated infinitely often.
- Multiplexed and passive nodes, and nodes that transmit frames with implicit C-state even in startup mode, must not be allowed to send a cold start frame or to answer to cold-start frames, i.e., their CF and CIA flags must be cleared.
- Permanent passive nodes must not integrate into a cluster, i.e., the RA flag must not be set in any slot.
- A set of shadow nodes must be in physical vicinity, because the delay correction used by the receiver nodes is equal for all shadows.

[4]The controller cannot detect traffic, but is able to transmit

11 Communication Network Interface

The basic TTP/C communication network interface (CNI) is the interface of the controller that is visible to the system programmer of the host computer. The CNI is a memory area that allows simultaneous random access by the controller and the host CPU with some areas read-only for the host.

The implementation of the CNI can vary, but some parts should be physically located on the controller.

Conceptually the CNI can be divided into three major areas:

Status Area The status area is used to display information of controller and the TTP/C network to the host processor. It contains the CNI status fields, which are updated by the controller and are read-only for the host. It is recommended that the status area is physically located on the controller.

Control Area The control area is used for control commands from the host to the controller, like mode change request or the host life-sign. This area may be written by the host but fields can be reset by the controller after consumption. It is recommended that the control area is also physically located on the controller.

Message Area The message area contains all application messages that are sent or received by the node. The frame data field contains the application data transmitted by the controller. Each frame data field has an associated frame status field. For frames received by the node, the frame status field contains information on the reception of the frame that contained the data; for frames sent by the node, the frame send status flag contains information on whether the data in the frame data field is valid for transmission.

The message area may be part of the controller or host memory[1] but in both cases a concurrency control mechanism must be implemented.

An asynchronous interrupt line (or more lines) from the controller to the host must be implemented for supporting time-triggered operating systems on the host computer.

11.1 Description of the CNI Fields

A detailed CNI layout is controller implementation dependent and cannot be given in the abstract TTP/C specification. Some fields described in this section are optional but recommended.

[1]e.g. located on a DPRAM in the controller or located in the host main memory and accessed via DMA transfers

11.1.1 Status Area

The information shown from the controller to the host can vary in different implementations and depends on the level of diagnosis information the host wants to evaluate. It is in the scope of the host to get a consistent view of the status area by using a concurrency control mechanism implemented in the controller or accessing the status area in the intervals when it is stable (see also section 5.6 on page 32): during the idle phase and during the transmission phase the contents of the status area does not change [2]. The status area is cleared by the controller at the begin of the initialization.

The listed status fields and flags should be provided by the protocol and controller implementation.

C-state The host should have a view of the actual C-state whose contents are described in chapter 6 on page 35. The C-state information becomes valid when the controller integrates on a frame with explicit C-state or is acknowledged after sending a cold start frame.

Clock State Correction Term The clock state correction term is the current value of the correction term that has been computed by the clock synchronization algorithm (usually in units of microticks).

The clock state correction term must be evaluated by the host if external clock correction is used: the external clock correction will cause a synchronization error if the total clock state correction term (sum of the clock state correction term by the clock synchronization algorithm and the external rate correction) exceeds the $\frac{\Pi}{2}$ interval.

Protocol State The protocol state field contains the state that the TTP/C protocol currently executes (see table 12.1 on page 100).

Protocol Error Flags The following flags denote internal protocol errors detected by the controller. They are updated as soon as the appropriate error condition is detected by the controller, and at the same time an interrupt may be raised (implementation dependent). The flags are cleared at the controller initialization; the controller immediately freezes upon the occurrence of any protocol error.

Clique Error (CE) The acknowledgment logic has detected that the controller is in disagreement with the majority of the cluster (see section 9.2.2 on page 69). This either indicates that an asymmetric (Byzantine) error has occurred and the controller is a member of the minority clique, or that a majority of the transmissions of the last TDMA round was disrupted on both channels.

Synchronization Error (SE) The clock synchronization subsystem has detected an error (see section 9.1.4 on page 56).

Membership Error (ME) The maximum number of successive membership failures specified in the maximum membership failure count (section 9.1.6 on page 63) has been reached.

[2] except the progress of the global time

Communication System Blackout (CB) This flag is set if a blackout of the communication system occurs (i.e., no bus activity except the controller's own transmission has been observed during the last TDMA round; see section 9.2.2 on page 70).

Bus Guardian Error (BE) This error is optional in case of a local bus guardian with the ability to inform the controller about detected faults.

Host Error Flags These flags are set by the controller. They are cleared during initialization and in the PSP of the controller's own transmission. Host errors can only occur in the controller's own node sending slot – if a host error is detected the controller will not send in its slot and loses the slot occupation. It therefore must acquire its slot after integration in the startup as described in section 9.1.2 on page 49.

Slot Occupied Error (SO) When the controller tried to acquire a slot, it detected that the sending slot was already occupied by another controller (the membership flag is already set). This indicates that

- this node is a shadow node, the host is updating the life-sign, but another node has already acquired this slot, or
- another controller has, presumably by mistake, got the same personalized MEDL.

Mode Violation Error (MV) The host requested a mode change that it is not allowed to request, because this mode change is prohibited by the mode change permissions in the controller's own sending slot.

Frames Not Ready (NR) The RS flag has not been set properly by the host for frames to transmit (see section 9.1.3 on page 52).

Concurrency Control Error (CC) The controller detected a concurrency access violation by the host, i.e. the host wrote data into a frame data field while the controller was reading that field in the pre-send or in the transmission phase.

Built-In Self Test (BIST) Error Flags During startup and run-time of a TTP/C controller several self tests can be performed. Because these tests depend on the controller implementation various error flags may be implemented. The CRC check over the MEDL at startup and a periodic CRC check during the protocol process are a requirement of the TTP/C protocol:

MEDL CRC Error (MC) The CRC check over the MEDL has failed, the controller freezes.

TTP/C Controller Information This section contains information of the TTP/C controller and the protocol version used. Each make of TTP/C controller (not each individual TTP/C controller) must have its own unique identifier and reports it to the host. A serial number of each individual controller is optional and also belongs to this section like additional information describing the controller, e.g. a byte-order marker (BOM) .

Cluster Schedule Identification This field contains the schedule ID value that is computed by the cluster designer and put into the MEDL.

TTP/C implicitly checks whether all controllers have the same schedule ID in their personalized MEDLs by initializing the frame CRC calculation with this value (see section 8.3 on page 44). Additionally, the host application can read that value from this field and compare whether the expected schedule ID matches the reported one.

Application Identification The cluster designer can specify an arbitrary application ID value and put it into the MEDL. The host application can then read that value from this field and check if the expected application ID matches this value.

Unlike the schedule ID, the application ID is a node-local identifier. A change in the application may be node-local and affects the MEDL of the local node only; such a change would be reflected in a change of the application identification, while the schedule ID (which is a cluster-wide common identifier) remains the same.

Current Logical Name The logical name of a member node that determines the current role of the node. The contents of the logical name are described in section 10.3.2 on page 81.

Controller Life-sign This field is used by the controller to provide a periodic life-sign information to the host, if no other frequently updated fields are used for a life-sign mechanism (e.g. C-state time or round slot position). This life-sign should be updated in its own sending slot.

Cluster Time Field This field contains the globally synchronized time with a granularity of one macrotick. The cluster time field is read only for the host. It is initialized according to the current globally synchronized cluster time (derived from the C-state time of the frame the controller integrates on) and from then on it is continuously synchronized with the globally synchronized cluster time by the clock synchronization mechanism. At each action time, the cluster time field contains the same value as the C-state time field; during the TDMA slot, the cluster time field advances in steps of macroticks, while the C-state time field is only updated each action time.

The contents of this field are undefined if the C-state is not valid.

Interrupt Status Field The interrupt status field contains information on the interrupt conditions that are currently triggered by the controller. The information is typically encoded in a flag vector where each flag represents an interrupt condition. All flags are set by the controller and are cleared by an interrupt acknowledgment mechanism; this mechanism must ensure that no TTP interrupts are lost if the host acknowledges one or more TTP interrupts concurrently with another occurrence of the same or other TTP interrupts.

The interrupts can be disabled by clearing the corresponding flags in the interrupt enable field (see section 21 on page 92). The individual flags in the interrupt status field are set whenever the appropriate interrupt condition occurs, regardless of whether this interrupt condition is enabled in the interrupt enable field. Therefore the application can poll for the occurrence of any of the interrupt conditions without having to enable interrupts. At power-on the interrupt status is cleared.

A detailed list of all non-optional TTP/C interrupts is given in section 11.3 on page 94.

11.1.2 Control Area

The fields of the control area can be read and written by the host at any time, some of them are cleared by the controller after reading. These fields are used from the host to request services from the TTP/C protocol. The control area must not be cleared by the controller at initialization, but is in the scope of the host – the host must have the opportunity to initialize the fields before it turns the controller on.

Controller On Flag (CO) This flag is used to turn the controller on or off. During normal operation the flag is set. The host can turn the controller off at any time by clearing the flag – the controller then freezes. Whenever the controller freezes autonomously, e.g., due to a protocol error, this flag is cleared by the controller. The host can turn the controller on again by setting this flag. Setting this flag to a value it already contains has no effect. At power-on this flag is cleared.

Controller Await Flag (CA) This flag is used by the host processor to initiate the controller's transition into the Await state. If it is cleared, the controller will start regular protocol execution. A set CA flag forces the controller into the Await state after startup. The CA flag must therefore be set before the host sets the CO flag to activate the controller.

Built-in Self Test Flag (BIST) This flag is used by the host processor to initiate the controller's transition into the test state. If it is cleared, the controller will start regular protocol execution. A set BIST flag forces the controller into the test state after startup. The BIST flag must therefore be set before the host sets the CO flag to activate the controller.

Host Life-sign This field is used by the host to provide a periodic life-sign information to the controller. If the host does not perform a correct periodic update of this field, the controller will not transmit data and lose its slot acquirement. A detailed description is given in section 9.2.3 on page 70.

External Rate Correction Field The external rate correction field allows the host processor to specify a common mode rate correction term in units of microticks per resynchronization interval, which is added to the correction term of the TTP/C internal clock synchronization algorithm. It is needed by an external clock synchronization algorithm executed by the host processor.

The controller will regard the contents of this field only if the Allow External Rate Correction flag (AR) in the MEDL is set. The contents of the field are bounded by the $\frac{\Pi}{2}$ interval – a synchronization error is raised in case the host violates the condition.

At the start of the resynchronization interval (i.e., the action time of each slot with the ClkSyn flag in the MEDL set) the number contained in this field is added to the clock state correction term calculated by the clock synchronization algorithm, giving the total clock state correction term. The total clock state correction term yielded by this computation is bounded by the $\frac{\Pi}{2}$ interval (section 9.1.4 on page 56). The controller clears this field after reading it, requiring the host processor to reprogram the current rate correction value in each resynchronization interval.

Mode Change Request Field The host subsystem writes a mode change request into the mode change request field to indicate that a request for a change of the current cluster mode should be transmitted by the controller. When assembling frames for transmission (i.e., during the PSP), the controller reads this information and then clears this field.

Then the controller checks whether the request is valid according to the information stored in the mode change permissions for the current round slot in the MEDL. If this condition is fulfilled, the controller transmits the mode change request in both frame headers. After transmitting a deferred mode change request, the controller sets the C-state DMC (deferred mode change) field to the value of the transmitted mode change request.

If the mode change request from the host is invalid according to the MEDL, the controller raises a host interrupt and sets the Mode Violation Error (MV) in the host error flags.

Timer Field The timer mechanism allows the host processor to choose a specific point in time at which an interrupt should be raised. For this purpose, the host processor writes the desired value of the global time to this field. When the global time reaches this value, a *timer interrupt* (see section 11.3.3 on page 95) to the host processor is raised.[3]

Interrupt Enable Field This field corresponds to the interrupt status field. It usually consists of a mask that allows to enable or disable each of the interrupt conditions separately. By disabling all interrupt conditions in this field, the host CPU ensures that it will never receive an interrupt from the TTP/C controller.

Timer Overflow Counter Field This field presents a counter which is initialized with 0 upon a reset of the controller and is incremented each time an overflow of the cluster time occurs.

This field can be set by the host processor to any value and can therefore be used to build a larger ranged time base with macrotick granularity, but only the synchronized cluster time part is equal on all nodes; the timer overflow counter can be synchronized by the host applications.

Time Startup Field This field is used for synchronized cluster startup. The C-state time value transmitted in the first cold start frame of the controller contains this value. The controller reads this value at a defined point in time before transmitting the cold start frame (this point in time is available in the controller data-sheet), the host must therefore update the field accordingly. Synchronized cluster startup is typically used in conjunction with inter-cluster synchronization (see section 9.3.2 on page 75).

[3]The latency of the interrupt hardware in the host CPU adds an unknown amount of jitter to the processing of this interrupt; for highly jitter-sensitive applications, this must be taken into account when using the TTP time interrupt.

11.2 Message Area

The message area contains the received application data and the data the node sends. The layout of this area is determined by the data layout parameters in the round slot entries of the personalized MEDL. This layout can be selected individually for each node in a cluster without impact on the personalized MEDLs of other nodes. The initialization of the message area is in the scope of the host.

An entry of the CNI message area consists of the frame status and its associated application data.

An example of a message area layout is shown in figure 11.1.

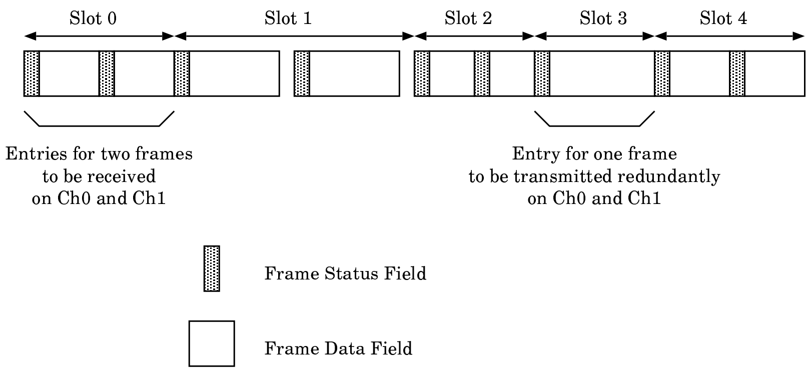
Slot 0
Slot 1
Slot 2
Slot 3
Slot 4
Entries for two frames to be received on Ch0 and Ch1
Entry for one frame to be transmitted redundantly on Ch0 and Ch1
Frame Status Field
Frame Data Field

Figure 11.1: Example Layout of a Message Area

During the transmission phase the controller reads and updates the message area according to the transmission schedule.

The reception of frames containing application data overwrites data previously stored in the message area at the memory location stated in the CNI address fields of the actual MEDL round slot entry – that can happen even if the frames are not correct. The frame status field tells about the status of the data stored in each of the message area entries.

11.2.1 Frame Status Field

The frame status field provides information on the transmission and reception of the frame containing the associated application data. It allows the host processor to detect whether or not the frame was correctly received. The frame status fields can also be parts of the status area, the frame send status (section 9.1.3 on page 52: RS flag) can be located in the control area – the locations of frame data and frame status are independent (figure 11.1 shows only a possible implementation).

Frames from Host to Controller

For frames transmitted by this node, the frame status field is interpreted as Ready Status flag (RS) described in section 9.1.3 on page 52.

The host application has to make sure the RS flag is set before the controller prepares to transmit this frame.

When the controller reads a frame from the CNI, it examines the value of the RS flag. If this flag is not set, the NR host error is set, a host error interrupt is raised, and the controller does not send. Only if the RS flag is set, a transmission of the frame takes place[4].

The controller does not change the contents of the RS flag, i.e., it does not automatically reset the RS flag for frames after transmission.

Frames from Controller to Host

For frames received by this node, the frame status is set according to the frame states described in section 8.2 on page 41. The frame status for this frame is never recalculated once it has been written to the CNI unless a new reception is stored at the same message area entry.

11.3 Interrupt Signal

The interrupt is the only control signal that passes the CNI. It is raised by the controller and received by the host. For this purpose there is a single physical interrupt line from the controller to the host. The host must read the interrupt status field (section 20 on page 90) to determine the cause of an interrupt. If a host error or protocol error occurred, a more detailed analysis of the cause of the interrupt is contained in the protocol/host error flags. The various interrupt conditions are enabled or disabled in the interrupt enable field (section 21 on page 92).

There exist three classes of interrupts depending on the point in time when an interrupt raised by the controller becomes active to the host.

11.3.1 Asynchronous Interrupts

These interrupts are used to inform the host about errors or other important events with a very small latency. Thus the occurrence of these interrupts is not restricted to the macroticks of the global time base.

BIST Error (BR) An internal error of the controller has been detected by the self-test logic. The controller freezes – the cause of the error can be identified by accessing the BIST error flags.

[4]Transmission will only take place if the RS flag(s) is set for *all* frames to be transmitted in this slot. It is not possible to send frames selectively on only one channel by setting just one RS flag.

Protocol Error (PE) The TTP/C protocol detected a fault. The controller freezes – the cause of the error can be identified by accessing the protocol error flags.

Host Error (HE) When the controller detects a host error, it sets the corresponding flag in the interrupt status field and does not send (loses slot acquirement). Detailed information about the error is contained in the host error flags.

Controller Ready (CR) This interrupt is raised when the controller has finished its internal initialization, the MEDL is checked and the status area contains the information of the schedule and the controller personality.

11.3.2 Action Time Synchronous Interrupts

Interrupts of this type become active to the host at the next action time. If several of these interrupts occur at the same action time, they raise a single interrupt only at the host, but all appear in the interrupt status field.

C-state valid (CV) Notify the host that the C-state is valid and the controller is synchronized.

Membership Loss (ML) The acknowledgment decided that the controller has lost its membership in the cluster.

User Interrupt (UI) If this flag is set, the interrupt was triggered by the programmable timer. This interrupt is commonly used to inform operating system schedulers in time-triggered systems of a schedule event.

11.3.3 Macrotick Synchronous Interrupts

This type of interrupt always coincides with a full macrotick of the global time base.

Timer Interrupt (TI) If this flag is set, the interrupt was triggered by the programmable timer. This interrupt is commonly used to inform operating system schedulers in time-triggered systems of a schedule event.

11.4 Power-On Defaults

The controller on flag (CO) is cleared at power-on. The interrupt status field has a power-on default of zero. Furthermore, whenever the CO flag is changed from ‘cleared’ to ‘set’, the interrupt status field is cleared immediately.

11.5 Concurrency Control

Like the status area the message area entries must be protected by a concurrency control mechanism to avoid simultaneous access from host and controller, e.g. the host must not have permission to manipulate data that is currently sent by the controller.

In this case the controller detects a concurrency control violation, a host error interrupt is raised and the CC host error flag is set – the controller loses its slot acquirement.

The update of the CNI fields that change with the progression of the TDMA round takes place in the IFG of each node slot. During that time it is not guaranteed that the host reads a consistent view of the controller status.

Once the update has been performed, the values for the new node slot (except the C-state membership vector, which contains the membership calculated in the previous slot) are used for further operation.

Upon a mode change special care must be taken when accessing the application data contained in the message area, since these data might have been received in the old cluster mode (and not in the new cluster mode defined by the current absolute mode number) and thus have different semantics compared to the new cluster mode.

The intervals in which the host is allowed to access the message area (writing application data to send, reading received data) and can read the status area consistently are shown in figure 11.2.

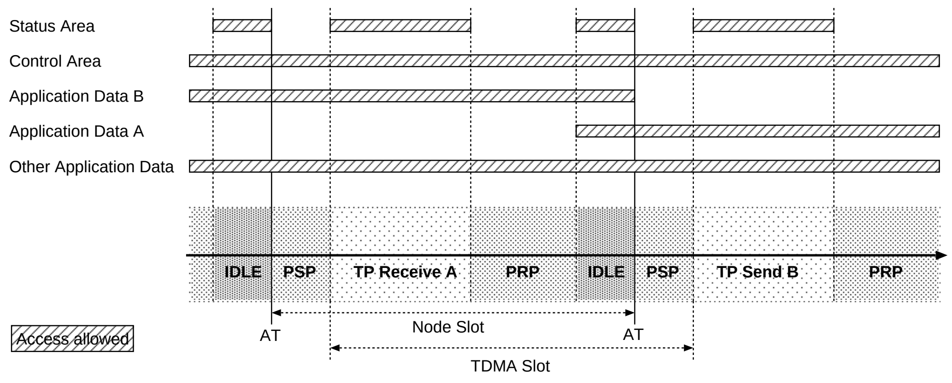

Figure 11.2: Allowed Host CNI Access for Node B

11.6 CNI Update Rates

Table 11.1 summarizes the update rates of the CNI status area fields.

CNI Field	Update Rate
Clock State Correction Term	every resynchronization interval
Protocol State	upon a state transition
Error flags	continuous
Logical Name	upon startup
Controller life-sign	each own sending slot
Cluster Time Field	every macrotick
Interrupt Status Field	continuous

Table 11.1: CNI Status Area - Update Rates

For fields in the CNI control area, no update rate can be stated, as they are written by the host and read by the controller [5]. The CNI message area is updated according to the transmission schedule in the round slot entries: the update rate for any field in the message area can be as long as a whole cycle or as short as a single inter-frame gap, determined by the MEDL.

[5]Except for the time overflow field, which is updated once every overflow of the cluster time field, and whenever the host writes it.

12 Protocol States

The process of the TTP/C protocol can be divided into several operational states (shown in figure 12.1 on page 101).

12.1 Protocol States and State Transitions

12.1.1 Protocol State Overview

Init (1) The TTP/C controller enters the init state after it is switched on (CO flag set by the host). In this state all CNI status fields are initialized.

Listen (2) The TTP/C controller enters the listen state when the initialization has been completed and tries to integrate on a received frame. It listens for any activity on both channels until the listen timeout expires.

Cold Start (3) The TTP/C controller enters the cold start state if it is allowed to (CF flag set and host alive) and assumes it is the first TTP/C controller in a recently powered on cluster since no frames were observed during the listen timeout. It facilitates the integration of other controllers by periodically sending cold start frames until it receives a response from another controller, or if it runs out of cold start retries; in this case, it goes back to the listen state.

Active (4) The controller transmits frames according to the contents of the MEDL, the host is active.

Passive (5) The controller is synchronized but the TTP/C controller has not yet acquired a node slot or it has lost the slot acquirement due to a detected fault. Reasons for that are:

- The host is inactive (no life-sign update).
- A failed host (Host Error).
- Loss of membership due to acknowledgment algorithm decision (if the MMFC is not reached).
- Have still not sent after integration.
- The node is permanent passive.

No frames are sent in this state but the controller is still receiving.

Await (6) Upon request of the host, the TTP/C controller is waiting for download or other implementation dependent asynchronous services instead of executing the TTP/C protocol.

Test (7) The TTP/C controller is performing self tests upon a request of the host processor, or is tested by the host processor. Like the await state behavior depends on the controller implementation.

Download (8) The TTP/C controller executes TTP/C download, i.e., it retrieves a new MEDL or new application data for the host processor via the TTP/C bus. It does not execute the TTP/C protocol at that time. The download protocol is not part of TTP/C and may vary with the controller implementations.

Freeze (0) The execution of the TTP/C protocol is halted until the controller is turned on by setting the CO flag. This state can be used by the host to retrieve diagnostic information about a previous error. It is also the initial state of the controller at power-on.

No.	State	Host	CO flag	Sending	Synchronized
1	Init	Don't Care	On	No	No
2	Listen	Don't Care	On	No	No
3	Cold Start	Up	On	Cold start frames	Yes [1]
4	Active	Up	On	Normal frames	Yes
5	Passive	Down	On	No	Yes
6	Await	Don't Care	On	No	No
7	Test	Don't Care	On	No	No
8	Download	N.A.	On	Download	N.A.
0	Freeze	Don't Care	Off	No	No

Table 12.1: States of the TTP/C Controller

[1] after having received one correct frame

12.1.2 TTP/C Controller State Transitions

Figure 12.1 depicts the possible states and state transitions of the TTP/C controller. Hereby the states drawn with gray background denote states where the controller sends frames on both channels whereas the shading indicates states where the host is updating the host life-sign in the CNI and is therefore considered to be alive.

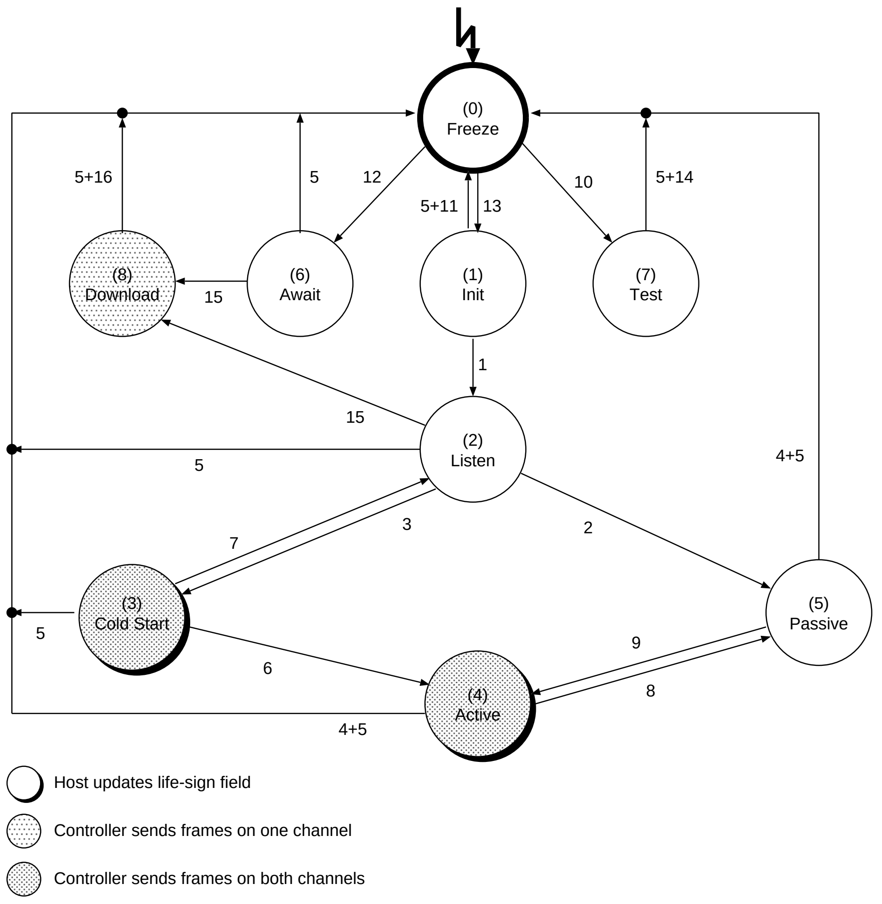

Figure 12.1: State Transitions of the TTP/C Controller

The transitions between these states are caused by the events listed in table 12.2.

No.	From	To	Event
1	Init	Listen	Initialization complete
2	Listen	Passive	Valid frame containing explicit C-state received
3	Listen	Cold Start	Listen timeout expired, cold start allowed and host life-sign updated
4	Active/Passive	Freeze	Clique error, Communication blackout, Sychronization error, Membership error, periodic MEDL CRC check failed
5	Listen/Cold Start/ Active/Passive/ Init/Await/ Test/Download	Freeze	CO flag turned off
6	Cold Start	Active	Controller in majority clique, at least 1 correct frame received (min. 2 controllers alive)
7	Cold Start	Listen	Clique error, host life-sign not updated, max. cold start entries exceeded or traffic detected on observed channel during startup timeout
8	Active	Passive	Host life-sign not updated, host error, membership loss
9	Passive	Active	Node slot acquired, host alive
10	Freeze	Test	Self tests requested (BIST flag set), CO flag set
11	Init	Freeze	MEDL CRC check failed
12	Freeze	Await	CA flag set, CO flag set
13	Freeze	Init	CO flag set, BIST flag and CA flag cleared
14	Test	Freeze	Tests completed
15	Listen/Await	Download	Frame indicating Download Mode
16	Download	Freeze	Download completed

Table 12.2: State Transitions of the TTP/C Controller

12.2 TTP/C Protocol States

12.2.1 Protocol Variables

The following variables must be used by the protocol state machine at run-time. They are not implementation dependent but are necessary for the algorithms of the TTP/C services:

Cold start counter Counts the number of performed cold starts of a controller.

Integration counter Counts the correct slots after integration on a non-cold start frame.

Agreed slots counter Counts the correct slots received during a TDMA round. The controller's own sending slot is also assumed to be correct. The counter is reset to 0 in the PSP of its own shared slot. It is initialized with 2 during integration and with 1 after successful sending (also a cold start frame).

Failed slots counter Counts the failed slots received during a TDMA round. The counter is reset to 0 in the PSP of its own shared slot and during integration.

Membership failure counter Counts the successive losses of the membership due to negative acknowledgment. The counter is reset after the controller is positive acknowledged.

Acknowledgment state State of the acknowledgment algorithm:

- Wait for first successor
- Wait for second successor
- Acknowledgment finished

First successor membership flag The saved position of the first successor's membership flag, needed by the acknowledgment algorithm if checking the second successor (see section 9.1.6 on page 63).

Big Bang flag Tags the performed big bang: after initialization the first received correct cold start frame is ignored and the listen timeout is restarted (see section 9.1.2 on page 49).

Free-shot flag Tags a controller's free-shot. A controller may send without updated host life-sign, if

- the node was not the cold starting one.
- it is the node's first transmission after integration.
- the cluster mode is the startup mode.

Observed channel Channel treated as the observed one during integration in the listen state and during the startup timeout in the cold start state. See section 9.1.5 on page 59 for more details.

Announcing these variables to the host via the CNI status area is optional and depends on the controller implementation.

12.2.2 Freezing

One state transition exists for all states: if the host clears the CO flag, the controller transits into the *freeze* state and stops operation.

12.2.3 Boot Process

As soon as the controller is switched on via the CO flag, it clears the interrupt status field and clears the CNI Status Area. The controller version is written into the CNI. The host is responsible for setting or clearing all fields in the CNI Control Area appropriately before starting the controller – the host may set the CA flag if it knows that the controller has a faulty MEDL, or it may set the BIST flag to perform some tests.

- If the CA flag is set, the controller transits immediately to the await state.
- If the host has commanded a controller (self-)test by setting the BIST flag field, the controller transits to the test state.

Otherwise the controller transits to the *init* state.

12.2.4 Init State

The controller checks the MEDL CRCs and if any error is detected, including a wrong version (or no) MEDL, the controller raises the MEDL CRC error (MC) and enters the freeze state.

Then it prepares the protocol execution, setting up hardware parameters like transmission speeds and macrotick length.

After the schedule and application IDs are copied into the CNI, the controller raises the controller ready interrupt (CR) and transits to the listen state.

12.2.5 Listen State

In the *listen* state, the controller waits for the duration of the listen timeout to receive a correct frame to integrate on. This can only be a frame with explicit C-state or a cold start frame.

If anything other than a correct frame with explicit C-state[2] is received, the controller behaves according to the integration noise tolerance described in section 9.1.5 on page 59: if a reception fails, not in all cases the listen timeout is restarted, but the observed channel is changed.

If a frame with explicit C-state and correct CRC is received on one of both channels, the controller uses one of them for the following checks. If two correct frames are received on both channels, the controller checks if both carry the same C-state. If the C-state differs

[2]Cold start frames do not contain a real C-state, but their global time and round slot information are treated as degraded C-state.

in both frames, the controller rejects both and restarts the listen timeout, in the other case the controller uses one for the integration.

At next the controller checks the type of the frame:

- The first correct cold start frame is discarded by the receiver; the receiver restarts the listen timeout. This mechanism called the big bang is described in section 9.1.2 on page 49.
- If the frame is a correct cold start frame and the controller has the ‘cold-start integration allowed’ (CIA) flag in the MEDL set, the controller tries to integrate – any other controller without CIA set discards this frame and restarts the listen timeout.

  The controller takes the MEDL position and the time field, sets the startup mode and marks the sender as active in the membership. Because a cold start frame has no real C-state, the position of the sender’s membership flag is derived from the round slot position of the received frame. The agreed slots counter is initialized with 2, the failed slots counter is reset to 0. From this moment, the controller is synchronized with the sender of the cold start frame and the C-state valid (CV) interrupt is raised at the next action time; it then transits into the passive state.
- If the frame contains a download signature, the controller transits into the *download* state[3].
- If the frame is not a cold start but normal frame with explicit C-state, the controller initializes its own C-state with the contents of the frame. In this case the cluster mode from the frame is used as it is, the agreed slots counter is initialized with 2, the failed slots counter is reset to 0. From this moment on the controller is synchronized with the sender of the frame and the C-state valid (CV) interrupt is raised at the next AT. Then the controller transits into the *passive* state.

If upon expiration of the listen timeout the controller is allowed to enter the *cold start* state (the CF flag in the controller’s personalized MEDL is set, the current value of the cold start counter is less than the maximum cold start entry in the MEDL, and the host has updated the host life-sign field), it immediately transits into cold start state.

If the controller is not allowed to perform cold start, it restarts the listen timeout and stays in the listen state.

12.2.6 Cold Start State

The controller sends cold start frames on both channels: these frames contain a shortened C-state consisting of the time value from the time startup field in the CNI control area – this value is copied into the controller’s C-state time for this purpose – and the MEDL round slot position of the controller’s first sending slot in the startup mode.

Afterwards the cold start counter is incremented by one. This counter counts the number of cold starts performed.

[3]This transition is optional and – like the download state – not part of TTP.

The controller will then step through the TDMA round according to the timing of the schedule and tries to receive frames (acknowledgment is not performed, a cold starting controller assumes itself as ‘acknowledged’) as described in section 12.2.7.

After reception of the first correct frame, the C-state valid interrupt (CV) is raised.

Reaching its own sending slot, the controller performs the clique avoidance test (if it has received more correct than incorrect answers) and checks for communication blackout (no answer was received). If both checks are positive, the controller transits to the active state.

If it is in the minority clique, it will transit to the listen state restarting the listen timeout, in case of communication blackout, it starts the startup timeout. Detecting bus traffic on the observed channel during this timeout will cause the controller to transit to the listen state, restarting the listen timeout and waiting for a reception.

When the startup timeout expires and the cold start counter is equal to or greater than the maximum cold start entry, the controller transits into the listen state. If the maximum cold start entry is zero, this check is not performed[4].

Otherwise the controller performs again all actions described in the cold start.

12.2.7 Passive State

Once the communication system operates correctly, the controller waits in the *passive* state until it can acquire the node slot corresponding to its current logical name. During the passive state a controller does not send any frames; the membership flag of a passive controller is unset. Nevertheless the controller will receive all frames from other nodes and continue to update the CNI.

Following actions have to be performed in every PSP:

- Reload the new round slot information from the MEDL. At the start of a cluster cycle, the deferred pending mode changes are checked in the C-state. If one is pending, the controller switches to the requested cluster mode by setting the mode into the C-state and selecting the corresponding round slot schedule.
- Check if the slot part of the current logical name of the node is equal to the current logical sender name at the current round slot position of the MEDL. If this is the case, i.e., the node is preparing for its own (or shared) sending slot, the following further actions are performed:
  - Perform the clique avoidance algorithm (section 9.2.2 on page 69) and check for a communication system blackout (section 9.2.2 on page 70). If one of the checks fails, the controller freezes.
  - Reset the agreed slots and failed slots counters to 0.
- Check if the whole current logical name of the node (including multiplex ID) is equal to the current logical sender name at the current position of the MEDL. If this

[4]This setting must only be used for experimental purposes.

is the case, i.e., the node is scheduled to acquire a slot and to send, the following additional actions are performed:

A controller with its own membership acquires a node slot and transits out of the passive state if:

- the host life-sign is updated, or the controller uses its free-shot[5],
- the controller is allowed to integrate in the slot (RA flag in the current slot in the MEDL is set),
- the controller has received 'minimum integration count' correct frames, if integrated on a non-cold start frame,
- and the membership flag is cleared, indicating that no other node (i.e., a shadow node) has acquired this slot.

A controller with shared membership acquires a node slot and transits out of the passive state upon the static reintegration flag (RA) and otherwise simply assumes that its node slot is available, since the membership is shared and does not contain the information required for the other checks. Therefore such controllers must not have shadows with which they could collide upon integration.

If a controller with its own membership tries to acquire a slot, but the membership flag indicates that some other controller has already occupied this slot, a slot occupied host error (SO) is raised and the controller remains in the passive state.

Upon positive acquiring a slot, the controller transits to the *active* state and sends in the slot.

- If a valid mode change is requested by the host, it will be distributed with the next transmission. If the request is not permitted (mode change permissions in the MEDL) the mode violation (MV) host error is set and the controller remains in the passive state.

Following actions have to be performed every PRP:

- Acknowlegdment is not performed because a passive controller does not send.
- The clock synchronization algorithm is executed.
- Incoming frames are checked for valid mode change requests. A valid request is set in the C-state 'deferred pending mode changes' (DMC) part.

Usually the controller is in passive state because the host temporarily was not ready to perform normally, and therefore ceased to update the host life-sign in the CNI control area. As soon as the host restarts updating the host life-sign, the controller tries to acquire a slot and transits to the active state.

Permanent passive nodes (section 5.4 on page 31) will never transit to the active state.

[5]The current C-state cluster mode is the startup mode and it is the node's first sending slot since the C-state became valid. This effectively grants the controller one transmission without the host setting the host life-sign. In the event of integrating into a cluster, the time until the first own sending slot may be too short for a host to set the host life-sign correctly. By allowing one single transmission without correct host life-sign during cluster startup, the host never needs to react faster than one TDMA round for the life-sign update.

12.2.8 Active State

The controller sends frames according to the round slot descriptions in the MEDL that corresponds to the current C-state cluster mode.

Following actions have to be performed in every PSP:

- Reload the new round slot information from the MEDL. At the start of a cluster cycle, the deferred pending mode changes are checked in the C-state. If one is pending, the controller switches to the requested cluster mode by setting the mode into the C-state and selecting the corresponding round slot schedule.
- Check if the slot part of the current logical name of the node is equal to the current logical sender name at the current round slot position of the MEDL. If this is the case, i.e., the node is preparing for its own (or shared) sending slot, the following further actions are performed:
  - Perform the clique avoidance algorithm (section 9.2.2 on page 69) and check for a communication system blackout (section 9.2.2 on page 70). If one of the checks fails, the controller freezes.
  - Reset the agreed slots and failed slots counters to 0.
- Check if the whole current logical name of the node (including multiplex ID) is equal to the current logical sender name at the current position of the MEDL. If this is the case, i.e., the node is scheduled to send in this slot, the following additional actions are performed:
  - Check whether the host computer is operating correctly by testing the host life-sign. If the life-sign is invalid, the controller switches to the passive state. The host life-sign is not required for the first transmission of the controller during startup (free-shot).
  - If a valid mode change is requested by the host, the request is transmitted in the slot. If the request is not permitted (mode change permissions in the MEDL), the mode violation (MV) host error is set and the controller transits into the passive state.

Following actions have to be performed every PRP:

- The controller performs the acknowledgment algorithm. If the node loses membership due to a negative acknowledgment, a membership loss (ML) interrupt is raised and the controller transits into the passive state. If the maximum membership failures are reached, the controller sets the membership error (ME) and freezes.
- The clock synchronization algorithm is executed.
- Incoming frames are checked for valid mode change requests. A valid request is set into the C-state 'deferred pending mode changes' (DMC) part. If the controller has sent in the slot and a valid mode change was requested, the request is set in the DMC field.

The controller also stays in active state when changing to another cluster mode.

12.2.9 Freeze State

Upon entering the freeze state, the controller clears the CO flag and suspends the protocol execution until the host turns the controller on again via setting the CO flag. In this case, the boot process is restarted.

When the controller is in the freeze state, the host must be able to read the contents of the CNI status area in order to determine the cause of a previous error of the controller.

12.3 Implementation Dependent States

These states are optional and their behavior depends on the protocol and controller implementation. Because actions performed in these states are independent of the TTP/C protocol, only a common behavior of existing TTP/C controllers is given.

12.3.1 Await State

The controller enters the *await* state if the host requested that transition of the controller by setting the CA flag in the CNI control area before starting the controller (setting the CO flag). This is used to bring a controller into download mode even if the controller is unable to start up normally because it does not have a correct MEDL.

The controller remains in the await state until a frame indicating download mode is received[6]. In this case the controller transits into the download state.

12.3.2 Test State

In this state the controller is subjected to tests to ensure correct functionality (e.g., DPRAM test). The tests can be performed by the controller autonomously – in this case they are requested by the BIST flag (see section 11.1.2 on page 91)[7] – or by the host.

If the controller performs autonomous self tests, once the self tests have been completed, the status of the performed test is written into the CNI [8].

After the test procedures, the controller must enter the freeze state, either by an autonomous transition or by clearing the CO flag. Based on the test results, the host application decides whether the controller is working properly and can restart the controller by setting CO.

12.3.3 Download State

In the *download* state, any MEDL can be downloaded from a download node using a stand-alone point-to-point protocol.

[6]The properties of such frames do not depend on the current MEDL
[7]Which self tests are actually available depends on the controller design.
[8]The location depends on the controller implementation.

Bibliography

[Ang98] H. Angelow. Implementation of the TTP/C Protocol with Advanced Services. Master's Thesis, Institut für Technische Informatik, Technische Universität Wien, Vienna, Austria, August 1998.

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

[Kop94a] H. Kopetz. Fault–Management in the Time–Triggered Protocol. In *Multiplexing and Fiberoptics (SP-1012)*, pages 77–84, Detroit, MI, USA, Feb. 1994. Society of Automotive Engineers, SAE Press. SAE Paper No. 940140.

[Kop94b] H. Kopetz and G. Grünsteidl. TTP — A Protocol for Fault-Tolerant Real-Time Systems. *IEEE Computer*, pages 14–23, January 1994.

[Kop96] H. Kopetz, R. Hexel, A. Krüger, D. Millinger, and A. Schedl. A Synchronization Strategy for a TTP/C Controller. In *Application of Multiplexing Technology (SP-1137)*, pages 19–27, Detroit, MI, USA, Feb. 1996. Society of Automotive Engineers, SAE Press. SAE Paper No. 960120.

[Kop97a] H. Kopetz. *Real-Time Systems*. Kluwer Academic Publishers, 1997.

[Kop97b] H. Kopetz. Specification of the Basic TTP/C Protocol, IP Board Version 4.0. Research Report 27/97, Institut für Technische Informatik, Technische Universität Wien, Vienna, Austria, December 1997. Confidential.

[Mes89] M.D. Mesarovic and Y. Takahara. *Abstract Systems Theory*. Springer Verlag, 1989.

[OSE01] *OSEK/VDX Fault-Tolerant Communication Specification 1.0*. OSEK/VDX Steering Committee, http://www.osek-vdx.org, July 2001.

[Pal97] R. Pallierer and T. M. Galla. Multiplexed SRUs in TTP/C – Concepts and Approaches. Research Report 19/97, Institut für Technische Informatik, Technische Universität Wien, Vienna, Austria, October 1997. Confidential.

[Pea80] M. Pease, R. Shostak, and L. Lamport. Reaching Agreement in the Presence of Faults. *ACM*, 27(2):228–234, 1980.

[Pea82] M. Pease, R. Shostak, and L. Lamport. The Byzantine Generals Problem. *ACM, Transactions on Programming Languages and Systems*, 4(3):382–401, 1982.

[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.

[Rus01] J. Rushby. Formal Verification of Transmission Window Timing for the Time-Triggered Architecture. Research report, Computer Science Laboratory, SRI International, Menlo Park CA 94025 USA, March 2001.

A Documents Structure

The specifications and description of TTP/C controllers are divided into three main layers (see figure A.1 on the following page):

High-Level / Abstract Layer Specifications in this layer describe algorithms and mechanisms with scientific and mathematical background without details about implementation approaches.

Compatibility Layer Specification in this layer define interfaces for the interoperability of TTP/C controllers, e.g. frame formats, bus encoding and endianess.

Implementation Layer Documents in this layer describe real controller implementations with all the necessary details to get the controllers running.

Each layer also contains a collection of requirements documents and test strategies for the contents of the related specifications and descriptions.

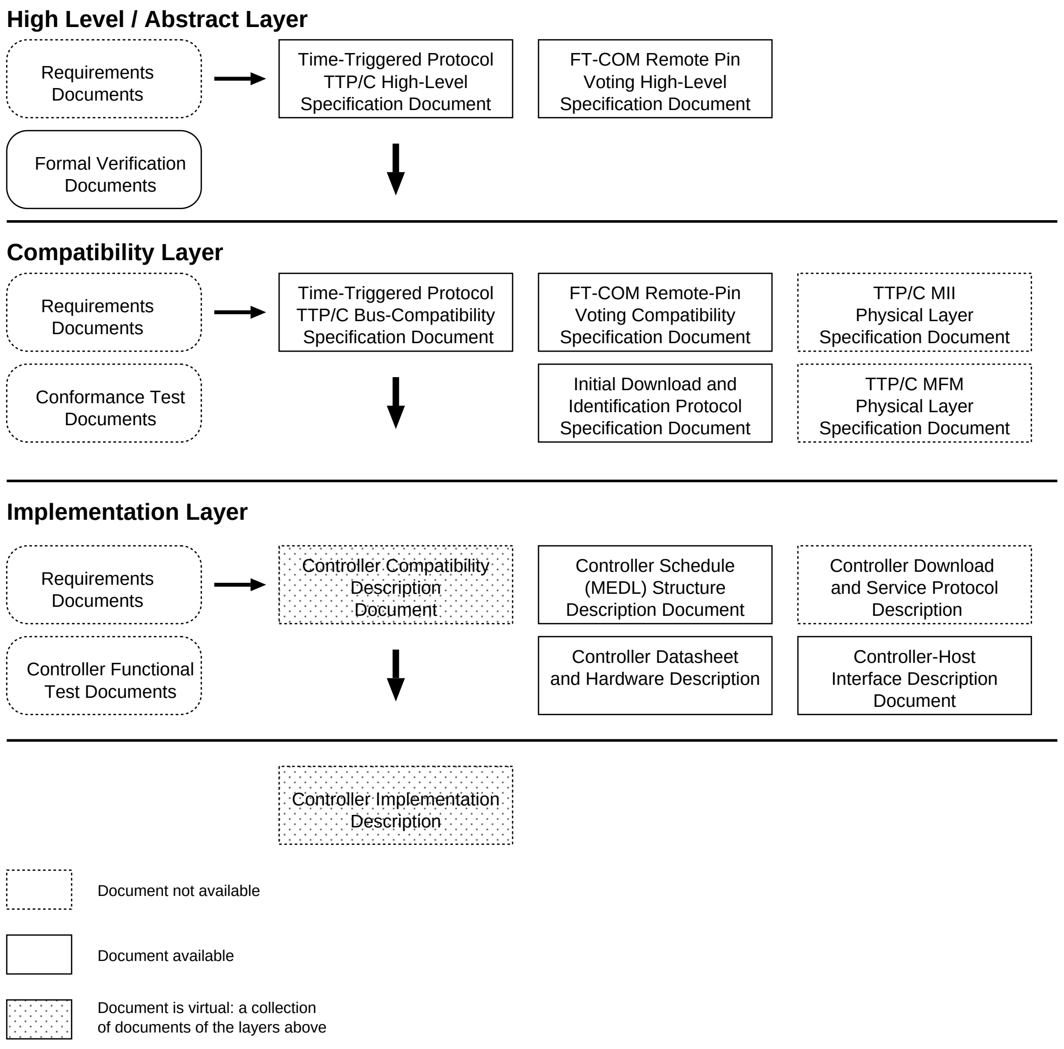
High Level / Abstract Layer
Requirements Documents
Time-Triggered Protocol TTP/C High-Level Specification Document
FT-COM Remote Pin Voting High-Level Specification Document
Formal Verification Documents
Compatibility Layer
Requirements Documents
Time-Triggered Protocol TTP/C Bus-Compatibility Specification Document
FT-COM Remote-Pin Voting Compatibility Specification Document
TTP/C MII Physical Layer Specification Document
Conformance Test Documents
Initial Download and Identification Protocol Specification Document
TTP/C MFM Physical Layer Specification Document
Implementation Layer
Requirements Documents
Controller Compatibility Description Document
Controller Schedule (MEDL) Structure Description Document
Controller Download and Service Protocol Description
Controller Functional Test Documents
Controller Datasheet and Hardware Description
Controller-Host Interface Description Document
Controller Implementation Description
Document not available
Document available
Document is virtual: a collection of documents of the layers above

Figure A.1: Specification Documents Structure

Index