INTERNATIONAL STANDARD

ISO 21806-14

First edition
2021-05

Road vehicles — Media Oriented Systems Transport (MOST) —

Part 14:
Lean application layer

Véhicules routiers — Système de transport axé sur les médias —

Partie 14: Couche d'application allégée

Reference number
ISO 21806-14:2021(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 31, *Data communication*.

A list of all parts in the ISO 21806 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Introduction

The Media Oriented Systems Transport (MOST) communication technology was initially developed at the end of the 1990s in order to support complex audio applications in cars. The MOST Cooperation was founded in 1998 with the goal to develop and enable the technology for the automotive industry. Today, MOST[1)] enables the transport of high quality of service (QoS) audio and video together with packet data and real-time control to support modern automotive multimedia and similar applications. MOST is a function-oriented communication technology to network a variety of multimedia devices comprising one or more MOST nodes.

Figure 1 shows a MOST network example.

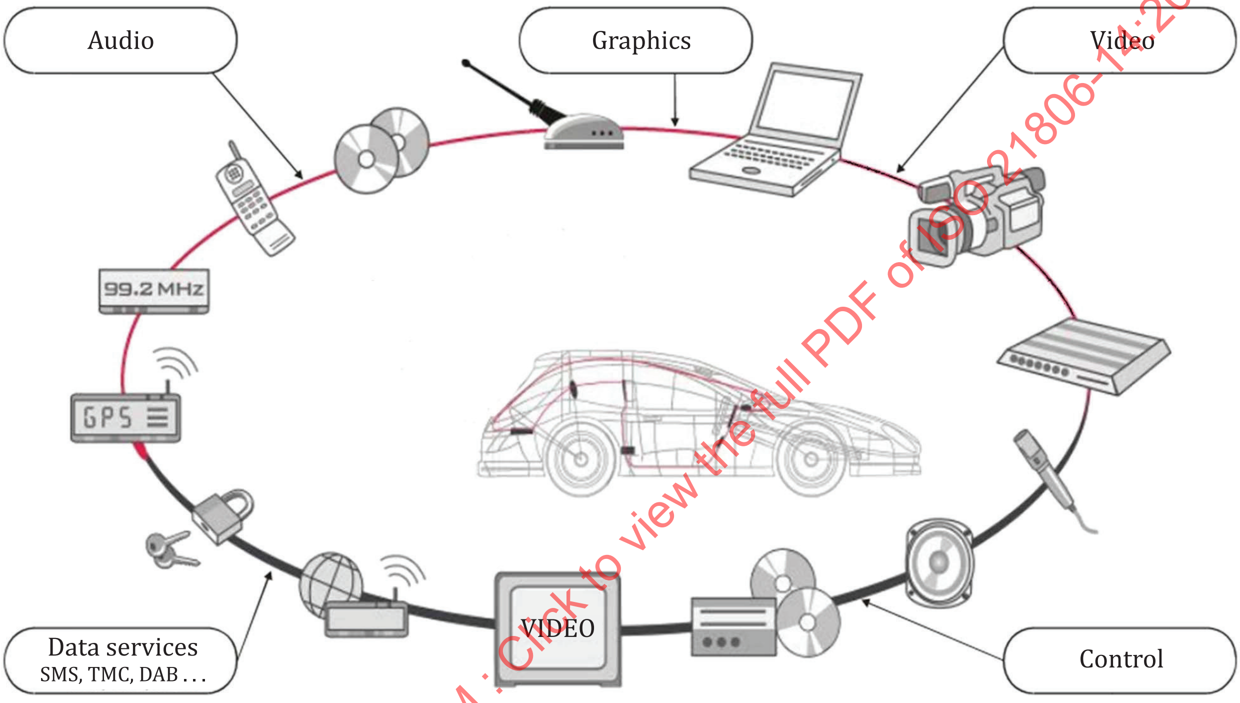


Figure 1 — MOST network example

The MOST communication technology provides:

- synchronous and isochronous streaming,
- small overhead for administrative communication control,
- a functional and hierarchical system model,
- API standardization through a function block (FBlock) framework,
- free partitioning of functionality to real devices,
- service discovery and notification, and
- flexibly scalable automotive-ready Ethernet communication according to ISO/IEC/IEEE 8802-3[2].

MOST is a synchronous time-division-multiplexing (TDM) network that transports different data types on separate channels at low latency. MOST supports different bit rates and physical layers. The network clock is provided with a continuous data signal.

[1)] MOST® is the registered trademark of Microchip Technology Inc. This information is given for the convenience of users of this document and does not constitute an endorsement by ISO.

Within the synchronous base data signal, the content of multiple streaming connections and control data is transported. For streaming data connections, bandwidth is reserved to avoid interruptions, collisions, or delays in the transport of the data stream.

MOST specifies mechanisms for sending anisochronous, packet-based data in addition to control data and streaming data. The transmission of packet-based data is separated from the transmission of control data and streaming data. None of them interfere with each other.

A MOST network consists of devices that are connected to one common control channel and packet channel.

In summary, MOST is a network that has mechanisms to transport the various signals and data streams that occur in multimedia and infotainment systems.

The ISO standards maintenance portal (https://standards.iso.org/iso/) provides references to MOST specifications implemented in today's road vehicles because easy access via hyperlinks to these specifications is necessary. It references documents that are normative or informative for the MOST versions 4V0, 3V1, 3V0, and 2V5.

The ISO 21806 series has been established in order to specify requirements and recommendations for implementing the MOST communication technology into multimedia devices and to provide conformance test plans for implementing related test tools and test procedures.

To achieve this, the ISO 21806 series is based on the open systems interconnection (OSI) basic reference model in accordance with ISO/IEC 7498-1[1] and ISO/IEC 10731,[3] which structures communication systems into seven layers as shown in Figure 2. Stream transmission applications use a direct stream data interface (transparent) to the data link layer.

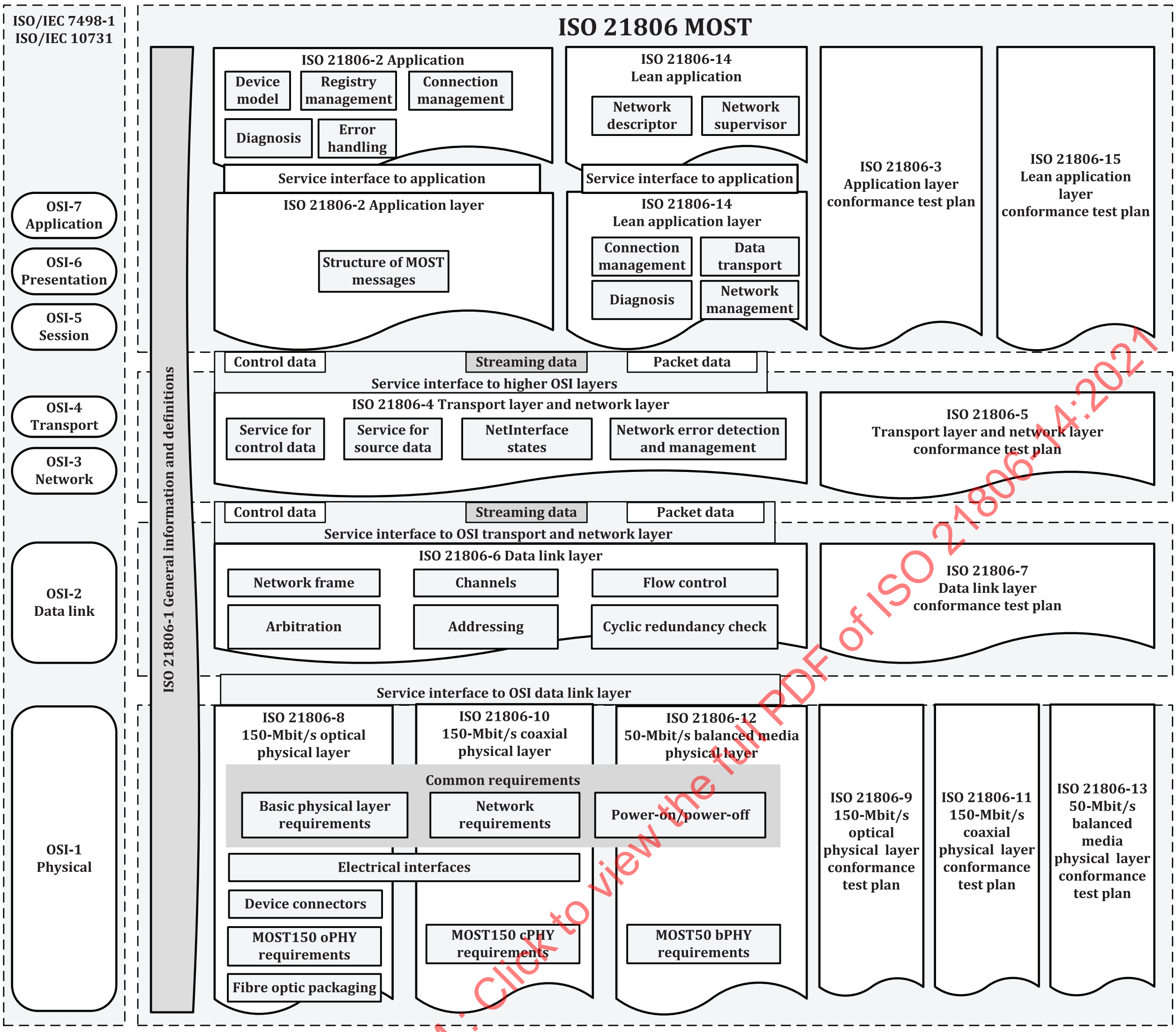

ISO/IEC 7498-1
ISO/IEC 10731
ISO 21806 MOST
OSI-7 Application
OSI-6 Presentation
OSI-5 Session
OSI-4 Transport
OSI-3 Network
OSI-2 Data link
OSI-1 Physical
ISO 21806-1 General information and definitions
ISO 21806-2 Application
Device model
Registry management
Connection management
Diagnosis
Error handling
Service interface to application
ISO 21806-2 Application layer
Structure of MOST messages
ISO 21806-14 Lean application
Network descriptor
Network supervisor
Service interface to application
ISO 21806-14 Lean application layer
Connection management
Data transport
Diagnosis
Network management
ISO 21806-3 Application layer conformance test plan
ISO 21806-15 Lean application layer conformance test plan
Control data
Streaming data
Packet data
Service interface to higher OSI layers
ISO 21806-4 Transport layer and network layer
Service for control data
Service for source data
NetInterface states
Network error detection and management
ISO 21806-5 Transport layer and network layer conformance test plan
Control data
Streaming data
Packet data
Service interface to OSI transport and network layer
ISO 21806-6 Data link layer
Network frame
Channels
Flow control
Arbitration
Addressing
Cyclic redundancy check
ISO 21806-7 Data link layer conformance test plan
Service interface to OSI data link layer
ISO 21806-8 150-Mbit/s optical physical layer
ISO 21806-10 150-Mbit/s coaxial physical layer
ISO 21806-12 50-Mbit/s balanced media physical layer
Common requirements
Basic physical layer requirements
Network requirements
Power-on/power-off
Electrical interfaces
Device connectors
MOST150 oPHY requirements
Fibre optic packaging
MOST150 cPHY requirements
MOST50 bPHY requirements
ISO 21806-9 150-Mbit/s optical physical layer conformance test plan
ISO 21806-11 150-Mbit/s coaxial physical layer conformance test plan
ISO 21806-13 50-Mbit/s balanced media physical layer conformance test plan

Figure 2 — The ISO 21806 series reference according to the OSI model

The International Organization for Standardization (ISO) draws attention to the fact that it is claimed that compliance with this document may involve the use of a patent.

ISO takes no position concerning the evidence, validity and scope of this patent right.

The holder of this patent right has assured ISO that he/she is willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of this patent right is registered with ISO. Information may be obtained from the patent database available at www.iso.org/patents.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those in the patent database. ISO shall not be held responsible for identifying any or all such patent rights.

Road vehicles — Media Oriented Systems Transport (MOST) —

Part 14: Lean application layer

1 Scope

This document specifies the technical requirements for the lean application layer for MOST, a synchronous time-division-multiplexing network.

The lean application layer includes a specification of the services available to control the configuration and behaviour of the network management.

The lean application layer covers the definition of:

- node kinds,
- node addressing,
- data transport,
- network configuration,
- lean network services interface,
- network management, including connection management,
- diagnosis, and
- timing definitions.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 21806-1, *Road vehicles — Media Oriented Systems Transport (MOST) — Part 1: General information and definitions*

ISO 21806-4, *Road vehicles — Media Oriented Systems Transport (MOST) — Part 4: Transport layer and network layer*

ISO 21806-10, *Road vehicles — Media Oriented Systems Transport (MOST) — Part 10: 150-Mbit/s coaxial physical layer*

ISO 21806-12, *Road vehicles — Media Oriented Systems Transport (MOST) — Part 12: 50-Mbit/s balanced media physical layer*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 21806-1 and the following apply.

ISO and IEC maintain terminological databases for use in standardisation at the following addresses:

— IEC Electropedia: available at http://www.electropedia.org/

— ISO Online browsing platform: available at https://www.iso.org/obp

3.1
coupling
tie between a streaming *socket* (3.11) and a network socket

3.2
lean network service
implementation of a software library that provides mechanisms to support *node discovery* (3.6) and establish a *network configuration* (3.3)

3.3
network configuration
nodes currently connected to the network and *paths* (3.7) established for transmission

3.4
network descriptor
list of nodes allowed to participate in network communication, including valid *paths* (3.7) with definition of *couplings* (3.1) and streaming connections

3.5
network supervisor
entity that is responsible for monitoring the *network configuration* (3.3)

3.6
node discovery
determination of the presence of the expected *remote nodes* (3.8)

3.7
path
two *couplings* (3.1) and a streaming connection

3.8
remote node
node that participates in a MOST network that conforms to this document and does not implement the *lean network services* (3.2)

3.9
root node
node in a MOST network that controls the network (discover, configure, and connect)

3.10
signature
unique identification of a node

Note 1 to entry: The signature includes the logical node address, DiagID, MAC address, etc. It is provided at design time. The signature is matched against the information contained in the *network descriptor* (3.4).

3.11
socket
interface of the MNC

Note 1 to entry: A socket is either a streaming socket or a network socket.

3.12
source drop
detection of unavailability of a streaming connection by a sink

Note 1 to entry: This could, for example, be caused by network or source malfunction.

4 Symbols and abbreviated terms

4.1 Symbols

---	empty table cell or feature undefined

4.2 Abbreviated terms

LNSI	lean network services interface
NSAI	network supervisor/application interface

5 Conventions

This document is based on OSI service conventions as specified in ISO/IEC 10731[2].

6 Basic principles

The MOST network consists of nodes. The node that provides the network clock and is responsible for startup and shutdown of the network, as well as network management is called root node. The other nodes that participate in the network communication are called remote nodes.

Figure 3 shows the MOST network example.

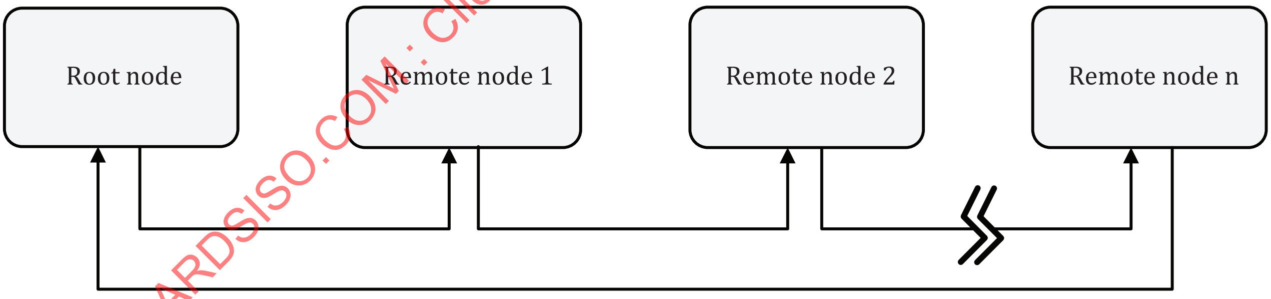


Figure 3 — MOST network example

Network management includes the process for startup and shutdown of the network.

Node discovery is a network management process that identifies all nodes in the network, which present a unique signature to the root node. Node discovery determines the presence of the expected remote nodes.

The nodes that currently exist in the network and the streaming connections between those nodes are called network configuration. The network supervisor in the root node is responsible for monitoring the network configuration.

The lean network services implement a software library that provides the root node with mechanisms for node discovery and establishing a network configuration. A remote node reacts to any lean network services request, for example, it answers to `Hello_Get` commands.

The connection management, as part of the lean network services, is a process that establishes paths, couplings, connections, and sockets based on the network descriptor.

7 Node kinds

7.1 General

REQ	**7.1 AL – Root node and remote nodes**
There shall be one root node and up to 63 remote nodes.	

The maximum number of nodes may be further reduced by physical layer restrictions.

7.2 Root node

7.2.1 General

The root node in a MOST network controls the network (discover, configure, and connect). This node is the one that implements the mandatory network supervisor.

REQ	**7.2 AL – Root node is TimingMaster**
The root node shall be the TimingMaster.	

REQ	**7.3 AL – Root node structure**
The root node shall contain: — the network supervisor, and — the lean network services.	

Figure 4 shows the root node. The application consists of software and/or hardware. It uses the network supervisor/application interface (NSAI) to communicate with the network supervisor. The application exchanges streaming data and packet data with the MNC.

The network supervisor uses the network descriptor to configure the lean network services by means of the lean network services interface (LNSI). The LNSI, in one direction, transports requests from the network supervisor to the lean network services and, in the opposite direction, provides responses to the network supervisor.

The lean network services exchange network management messages with the MNC. The lean network services send commands to and receive reports from the MNC.

The MNC of the root node is the TimingMaster of the MOST network.

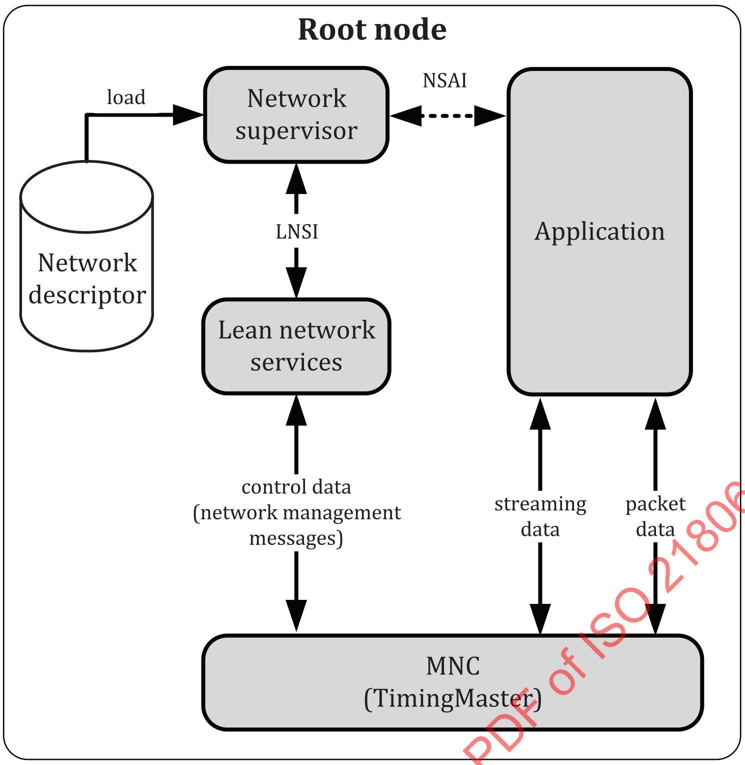


Figure 4 — Root node

7.2.2 Lean network services

The lean network services are implemented in the root node, exclusively.

<table>
<tr><th>REQ</th><th>7.4 AL – Lean network services scope</th></tr>
<tr><td colspan="2">The lean network services shall implement:
— startup;
— shutdown;
— node discovery;
— connection management; and
— diagnostic functions.</td></tr>
</table>

7.2.3 Network descriptor

<table>
<tr><th>REQ</th><th>7.5 AL – Network descriptor</th></tr>
<tr><td colspan="2">The network descriptor shall contain the definition of:
— nodes allowed to participate in network communication;
— valid paths;
— couplings;
— streaming connections;
— network sockets; and
— streaming sockets.
NOTE The network descriptor is static and known a priori.</td></tr>
</table>

REQ	**7.6 AL – Network supervisor relies on network descriptor**
Based on the network descriptor, the network supervisor shall determine which nodes are permitted to join the MOST network.	

An example of a network descriptor is provided in Annex C. The status of Annex C is informative.

7.3 Remote node

REQ	**7.7 AL – Remote node structure**
A remote node shall not contain the network supervisor or the lean network services.	

Network management and connection management are provided by the MNC.

REQ	**7.8 AL – Remote node reacts on requests**
The MNC of a remote node shall answer lean network service requests.	

Figure 5 specifies the remote node. The application consists of software and/or hardware. It exchanges streaming data and packet data with the MNC.

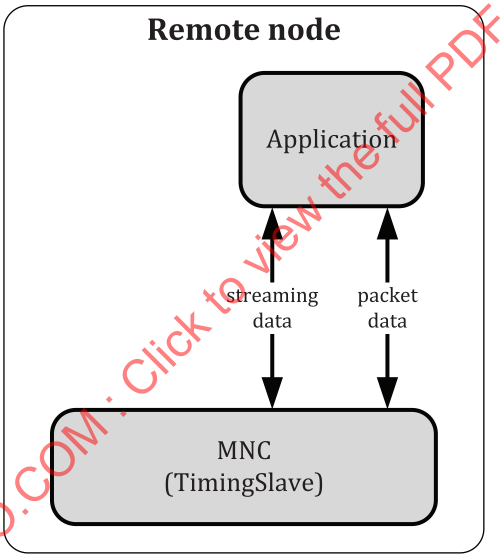


Figure 5 — Remote node

7.4 Listen-only node

For network analysis purposes, a certain node kind exists that does not change the content of network frames. A listen-only node is invisible to other nodes.

8 Node addressing

REQ	**7.9 AL – 16-bit address area structure**
For 16-bit addressing, the address area of an MNC shall be structured as specified in Table 1.	

Table 1 — 16-bit Address range

Address range	Description
0000_{16} to $000F_{16}$	Address range for internal communication between local network controller and lean network services
0010_{16} to $02FF_{16}$	Logical node address range
0300_{16} to $03C7_{16}$	Group address range
$03C8_{16}$	Blocking broadcast address
$03C9_{16}$ to $03FE_{16}$	Group address range
$03FF_{16}$	Non-blocking broadcast address
0400_{16} to $043F_{16}$	Node position address range
0440_{16} to $04FF_{16}$	Reserved
0500_{16} to $0EFF_{16}$	Logical node address range
$0F00_{16}$ to $0FEF_{16}$	Address range for administrative purposes
$0FF0_{16}$	Debug address
$0FF1_{16}$ to $0FFD_{16}$	Address range for administrative purposes
$0FFE_{16}$	Un-initialised logical node address
$0FFF_{16}$	Address for administrative purposes
1000_{16} to $FFFF_{16}$	Reserved

9 Data transport

The application may use the packet channel for communication with network devices. The application should not use the control channel.

REQ	7.10 AL – Administrative communication
The control channel shall be used for administrative communication, that is, network management and connection management.	

10 Network configuration

10.1 General

The network descriptor determines the network configuration, the nodes and the streaming connections between them. It can be either edited text-based or by use of a graphical tool.

The network descriptor is used to generate a data definition in a format that the lean network services are capable of processing.

The tables in the following subclauses contain a column that is titled "stored". The entries marked with "yes" are contained in the storage format for the network descriptor, while the entries marked with "no" are relevant for the network configuration, which is established during runtime. "Optional" entries, mainly resource handles, are relevant during runtime but may also be pre-configured.

10.2 Network

REQ	7.11 AL – Network configuration properties for the network
The properties of the network configuration that are specified in Table 2 shall be associated with the network.	

Table 2 — Network properties

Property	Data type/values	Stored	Description
`Activation`	`enumeration {on, off}`	no	`Activation` determines the intended state of the network.

10.3 Node

REQ	7.12 AL – Network configuration properties for nodes
The properties of the network configuration that are specified in Table 3 shall be associated with nodes.	

Table 3 — Node properties

Property	Data type/values	Stored	Description
`Availability`	`enumeration {available, not_available}`	no	The availability of a node depends on whether the `Hello/Welcome` sequence is performed successfully.
`Signature`	`complexType {NodeAddress, GroupAddress, MACAddress_47to32, MACAddress_31to16, MACAddress_15to0, DiagID, NumberOfPorts}`	yes	The signature of a node contains the `DiagID` and other relevant information.

10.4 Path

Paths are links between two nodes, consisting of two couplings and a streaming connection.

REQ	7.13 AL – Network configuration properties for paths
Table 4 specifies the properties of the network configuration for paths.	

Table 4 — Path properties

Property	Data type/values	Stored	Description
`Activation`	`enumeration {on, off}`	optional	`Activation` triggers the creation of the path.
`Identifier`	`string`	yes	`Identifier` is the unique identifier of the path.
`In_Coupling`	`string`	yes	`In_Coupling` is the identifier of the coupling that forwards the data from the network socket to the streaming socket.
`Out_Coupling`	`string`	yes	`Out_Coupling` is the identifier of the coupling that forwards the data from the streaming socket to the network socket.

10.5 Coupling

A coupling is a tie between a streaming socket and a network socket.

REQ	7.14 AL – In_Coupling direction
The `Direction` of both sockets for an `In_Coupling` shall be `in`.	

REQ	7.15 AL – Out_Coupling direction
The `Direction` of both sockets for an `Out_Coupling` shall be `out`.	

REQ	7.16 AL – Network configuration properties for couplings
The properties of the network configuration specified in Table 5 shall be associated with couplings.	

Table 5 — Coupling properties

Property	Data type/values	Stored	Description
`Direction`	`enumeration {in, out}`	yes	Determines whether the coupling is an `In_Coupling` or an `Out_Coupling`.
`Identifier`	`string`	yes	`Identifier` is the unique identifier of the coupling.
`Usage_Counter`	`unsignedByte`	no	A coupling can be used in more than one path. The usage counter reflects how many paths include the coupling.
`Network_Socket`	`string`	yes	`Network_Socket` is the identifier of the network socket.
`Node`	`string`	yes	`Node` is the unique identifier of the associated node.
`ResourceHandle`	`unsignedShort`	optional	Handle of the coupling, which is unique within a node. FFFF$_{16}$ indicates an invalid value.
`Streaming_Socket`	`string`	yes	`Streaming_Socket` is the identifier of the streaming socket.

10.6 Streaming socket

REQ	7.17 AL – Network configuration properties for streaming sockets
The properties of the network configuration specified in Table 6 shall be associated with streaming sockets.	

Additional vendor-specific properties may exist.

Table 6 — Streaming socket properties

Property	Data type/values	Stored	Description
`Direction`	`enumeration {in, out}`	yes	Determines whether the streaming socket is used in in- or out-direction.
`Identifier`	`string`	yes	`Identifier` is the unique identifier of the streaming socket.
`ResourceHandle`	`unsignedShort`	optional	Handle of the streaming socket, which is unique within a node. FFFF$_{16}$ indicates an invalid value.

10.7 Network socket

REQ	7.18 AL – Network configuration properties for network sockets
The properties of the network configuration specified in Table 7 shall be associated with network sockets.	

Table 7 — Network socket properties

Property	Data type/values	Stored	Description
`Bandwidth`	`unsignedShort`	yes	It is the bandwidth of the streaming connection.
`Channel`	`enumeration {synchronous, isochronous}`	yes	Determines whether the synchronous channel or the isochronous channel is used.

Table 7 *(continued)*

Property	Data type/values	Stored	Description
Connection_Label	unsignedShort	optional	Connection_Label is the MOST connection label that is associated with the network socket.
Direction	enumeration {in, out}	yes	Determines whether the network socket is used in in- or out-direction.
Identifier	string	yes	Identifier is the unique identifier of the network socket.
ResourceHandle	unsignedShort	optional	Handle of the network socket, which is unique within a node. FFFF_{16} indicates an invalid value.

11 Network supervisor/application interface (NSAI)

11.1 Overview

The network supervisor/application interface (NSAI) provides access to the network supervisor.

REQ	7.19 AL – NSAI requests, responses, and events
The requests, responses, and events that are exchanged between the application and the network supervisor shall be in accordance with Tables 8, 9, and 10.	

Requests are directed from the application to the network supervisor. Each request is answered by a response. Events are sent from the lean network services to the network supervisor. The parameters are specified in Annex A and shall be followed. The status of Annex A is normative.

In general, the network supervisor decides what actions are taken if it receives a response with Status = error.

Table 8 — Messages for the NSAI – Network startup and shutdown

Message	Parameter	Description
Network_Startup_Request	BoundaryDescriptor	Request to start up the MOST network
Network_Startup_Response	Status	Response to the network startup request
Network_Shutdown_Request	---	Request to shut down the MOST network
Network_Shutdown_Response	Status	Response to the shutdown request
Network_Status_Event	Accessibility, AccessibilityInfo, TransitionCause, NodeAddress, VisibleNodes, PacketBandwidth	Indication of a network status change

Table 9 — Messages for the NSAI – Connection management

Message	Parameter	Description
Connection_Management_Start_Request	NetworkDescriptor	Request to start connection management
Connection_Management_Start_Response	Status	Response to connection management start request
Connection_Management_Stop_Request	---	Request to stop connection management

Table 9 *(continued)*

Message	Parameter	Description
Connection_Management_Stop_Response	Status	Response to connection management stop request
Path_Activation_Set_Request	Path, Activation	Request to set the Activation of a path
Path_Activation_Set_Response	Path, Activation, Status	Response to path Activation set request

Table 10 — Messages for the NSAI – Diagnosis

Message	Parameter	Description
Diagnosis_Start_Request	DiagnosisSelector	Request to start diagnosis
Diagnosis_Start_Response	DiagnosisSelector, Status	Response to diagnosis start request
Diagnosis_Result_Event	DiagResult	Result for one link or node during diagnosis
Diagnosis_End_Event	---	Indication of completion of the diagnosis process

11.2 Requests, responses, and events

11.2.1 Network startup and shutdown

The application uses Network_Startup_Request to start up the MOST network. In Network_Startup_Response, if Status indicates success, the network is active. The value error indicates failure during startup.

The application uses Network_Shutdown_Request to shut down the MOST network. In Network_Shutdown_Response, if Status indicates success, the network is shut down. The value error indicates that the request is currently not allowed.

The network supervisor uses the Network_Status_Event to inform the application about a change in either network availability or a related parameter.

11.2.2 Connection management

The application uses Connection_Management_Start_Request to start connection management. In Connection_Management_Start_Response, if Status indicates success, the connection management is processing the network descriptor. The value error indicates a missing resource, which is dealt with by the network supervisor, typically by initialising the lean network services.

The application uses Connection_Management_Stop_Request to terminate connection management. In Connection_Management_Stop_Response, if Status indicates success, the connection management has stopped processing. The value error indicates that resources could not be released. Before starting connection management again, the lean network services should be initialised by the network supervisor.

The application uses Path_Activation_Set_Request to set the Activation of a path. In Path_Activation_Set_Response, if Status indicates success, the connection management processes connections related to the corresponding path. The value error indicates invalid parameters.

11.2.3 Diagnosis

The application uses Diagnosis_Start_Request to start diagnosis. Depending on the parameter, full-duplex network diagnosis or half-duplex network diagnosis is started. In Diagnosis_Start_Response, if

Status indicates success, the diagnostic process is running. The value error indicates that the selected diagnostic process cannot be started.

The network supervisor uses Diagnosis_Result_Event to report the diagnosis result for one link or node during diagnosis.

The network supervisor uses the Diagnosis_End_Event report to indicate that the diagnosis process is completed.

12 Lean network services interface (LNSI)

12.1 Overview

The lean network services interface (LNSI) provides access to the lean network services.

REQ	7.20 AL – LNSI requests, responses, and events
The requests, responses, and events that are exchanged between the network supervisor and the lean network services shall be in accordance with Tables 11 to 15.	

Requests are directed from the network supervisor to the lean network services. Each request is answered by a response. Events are sent from the lean network services to the network supervisor. The parameters are specified in Annex A and shall be followed. The details regarding restrictions for the different types used in the LNSI are contained in the corresponding XML schema definition in Annex B.

In general, the network supervisor decides what actions are taken if it receives a response with Status = error.

Table 11 — Messages for the LNSI – Lean network services

Message	Parameter	Description
Init_LNS_Request	InitDataStructure	Request to initialise the lean network services
Init_LNS_Response	Status	Response to the initialisation request
Stop_LNS_Request	---	Request to stop the lean network services process
Stop_LNS_Response	Status	Response to the stop request

Table 12 — Messages for the LNSI – Network startup and shutdown

Message	Parameter	Description
Network_Startup_Request	BoundaryDescriptor	Request to start up the MOST network
Network_Startup_Response	Status	Response to the network startup request
Network_Shutdown_Request	---	Request to shut down the MOST network
Network_Shutdown_Response	Status	Response to the shutdown request
Network_Status_Event	Accessibility, AccessibilityInfo, TransitionCause, NodeAddress, VisibleNodes, PacketBandwidth	Indication of a network status change
Node_Availability_Set_Request	Signature, Availability	Request to set the Availability property of a node

Table 12 *(continued)*

Message	Parameter	Description
Node_Availability_Set_Response	Signature, Availability, Status	Response to the Availability set request

Table 13 — Messages for the LNSI – Node discovery

Message	Parameter	Description
Node_Discovery_Start_Request	---	Request to start node discovery
Node_Discovery_Start_Response	Status	Response to the node discovery start request
Node_Discovery_Event	Signature	Indication that a node is discovered
Node_Discovery_Stop_Request	---	Request to stop node discovery
Node_Discovery_Stop_Response	Status	Response to the node discovery stop request
Node_Welcome_Request	Signature	Request to send a Welcome_StartResult message to a remote node
Node_Welcome_Response	Signature, Status	Response to the Welcome_StartResult request
Check_Uniqueness_Request	Signature	Used when two remote nodes with identical signatures exist
Check_Uniqueness_Response	Signature, Status	Response to uniqueness check request

Table 14 — Messages for the LNSI – Connection management

Message	Parameter	Description
Connection_Management_Start_Request	NetworkDescriptor	Request to start connection management
Connection_Management_Start_Response	Status	Response to connection management start request
Connection_Management_Stop_Request	---	Request to stop connection management
Connection_Management_Stop_Response	Status	Response to connection management stop request
Path_Activation_Set_Request	Path, Activation	Request to set the Activation of a path
Path_Activation_Set_Response	Path, Activation, Status	Response to path Activation set request

Table 15 — Messages for the LNSI – Diagnosis

Message	Parameter	Description
Diagnosis_Start_Request	DiagnosisSelector	Request to start diagnosis
Diagnosis_Start_Response	DiagnosisSelector, Status	Response to diagnosis start request
Diagnosis_Result_Event	DiagResult	Result for one link or node during diagnosis
Diagnosis_End_Event	---	Indication of completion of the diagnosis process

12.2 Requests, responses, and events

12.2.1 Lean network services

The network supervisor uses Init_LNS_Request to initialise the lean network services. In Init_LNS_Response, if Status indicates success, the initialisation of the lean network services is completed. The value error indicates invalid parameters.

The network supervisor uses Stop_LNS_Request to stop the network management processes (see Clause 13) of the lean network services. In Stop_LNS_Response, if Status indicates success, the lean network services are stopped after completing the requested actions. The value error indicates that the lean network services terminated the management processes after expiration of $t_{\text{LNS\_Termination}}$. The management processes are reset without completing all requested actions.

12.2.2 Network startup and shutdown

The network supervisor uses Network_Startup_Request to start up the MOST network. In Network_Startup_Response, if Status indicates success, the network is active. The value error indicates failure during startup.

The network supervisor uses Network_Shutdown_Request to shut down the MOST network. In Network_Shutdown_Response, if Status indicates success, the network is shut down. The value error indicates that the request is currently not allowed.

The lean network services use the Network_Status_Event to inform the network supervisor about a change in either network availability or a related parameter.

12.2.3 Node discovery

The network supervisor uses Node_Discovery_Start_Request to commence node discovery. In Node_Discovery_Start_Response, if Status indicates success, the node discovery process is running.

The value error indicates missing preconditions, for example, a successful network startup.

The lean network services use Node_Discovery_Event to inform the network supervisor that a node is discovered. The network supervisor compares the signature to the network descriptor and sends Node_Welcome_Request. When the remote node is added to the network, the network supervisor sends Node_Availability_Set_Request (available).

The network supervisor uses Node_Welcome_Request to instruct the lean network services to send Welcome_StartResult to a remote node. In Node_Welcome_Response, if Status indicates success, the node identified by the signature is added to the network and becomes part of the network configuration. The value error indicates a transmission error during the "Welcome" sequence for the corresponding node.

The network supervisor uses Node_Availability_Set_Request to set the Availability property of a node. In Node_Availability_Set_Response, if Status indicates success, the connection management processes connections related to the corresponding node. The value error indicates invalid parameters.

The network supervisor uses Node_Discovery_Stop_Request to terminate node discovery. In Node_Discovery_Stop_Response, if Status indicates success, the node discovery process is stopped. The value error indicates that the process is still running.

The network supervisor uses Check_Uniqueness_Request to determine why two remote nodes report identical signatures. When an already available node responds to node discovery, the network supervisor checks whether a reset occurred or two remote nodes with identical signatures exist. In Check_Uniqueness_Response, if Status indicates success, there is only one node with that signature in the network. The value error indicates the presence of two nodes with identical signatures.

12.2.4 Connection management

The network supervisor uses `Connection_Management_Start_Request` to start the connection management. In `Connection_Management_Start_Response`, if `Status` indicates `success`, the connection management is processing the network descriptor. The value `error` indicates a missing resource and the lean network services should be initialised again.

The network supervisor uses `Connection_Management_Stop_Request` to terminate the connection management. In `Connection_Management_Stop_Response`, if `Status` indicates `success`, the connection management has stopped processing. The value `error` indicates that resources could not be released. Before starting connection management again, the lean network services should be initialised.

The network supervisor uses `Path_Activation_Set_Request` to set the `Activation` of a path. In `Path_Activation_Set_Response`, if `Status` indicates `success`, the connection management processes connections related to the corresponding path. The value `error` indicates invalid parameters.

12.2.5 Diagnosis

The network supervisor uses `Diagnosis_Start_Request` to start the diagnosis. Depending on the parameter, full-duplex network diagnosis or half-duplex network diagnosis is started. In `Diagnosis_Start_Response`, if `Status` indicates `success`, the diagnostic process is running. The value `error` indicates that the selected diagnostic process cannot be started.

The lean network services use `Diagnosis_Result_Event` to report the diagnosis result for one link or node during diagnosis.

The lean network services use the `Diagnosis_End_Event` report to indicate that the diagnosis process is completed.

13 Network management

13.1 Network startup and shutdown

REQ	7.21 AL – Network supervisor starts up the network
After initialising the lean network services, the network supervisor shall send `Network_Startup_Request` to the lean network services to start up the network. NOTE Only the root node is authorized to start up the network.	

REQ	7.22 AL – Lean network services performs startup and shutdown
The lean network services shall perform startup and shutdown by switching on or off the MOST output, respectively.	

13.2 Network events

13.2.1 General

The following network events will cause a node in a MOST network to take corrective action: sudden signal off (SSO, see 13.2.2), unlock/critical unlock (see 13.2.3), network change event (NCE, see 13.2.4), and source drop (see 13.2.5).

REQ	7.23 AL – Root node mutes
If a source drop, unlock, or end of network activity occurs, the root node shall activate the hardware signal for muting the output of affected `In_Couplings`.	

<table>
<tr><th>REQ</th><th>7.24 AL – Remote node mutes</th></tr>
<tr><td colspan="2">If a source drop, unlock, or end of network activity occurs, a remote node shall activate the hardware signal for muting the output of affected In_Couplings.</td></tr>
</table>

13.2.2 End of network activity

<table>
<tr><th>REQ</th><th>7.25 AL – Root node reacts to end of network activity</th></tr>
<tr><td colspan="2">When network activity ends, the root node shall mark all remote nodes as not_available.</td></tr>
</table>

When network activity ends and the shutdown flag is set, a normal shutdown is performed. When network activity ends and the shutdown flag is not set, a sudden signal off (SSO) has occurred.

<table>
<tr><th>REQ</th><th>7.26 AL – Root node reacts on SSO</th></tr>
<tr><td colspan="2">If an SSO occurs and Activation is on for the network, the root node shall restart the network.</td></tr>
</table>

<table>
<tr><th>REQ</th><th>7.27 AL – Connection management invalidates resource handles</th></tr>
<tr><td colspan="2">When network activity ends, the connection management shall set the ResourceHandles that are associated with couplings, streaming sockets, and network sockets to FFFF$_{16}$.
NOTE FFFF$_{16}$ indicates an invalid value.</td></tr>
</table>

13.2.3 Unlock and critical unlock

<table>
<tr><th>REQ</th><th>7.28 AL – Root node reacts on critical unlock</th></tr>
<tr><td colspan="2">When a critical unlock occurs and Activation is on for the network, the root node shall restart the network.</td></tr>
</table>

13.2.4 Network change event

A network change event occurs when a node opens or closes its bypass, that is, enters or leaves the network. When a node leaves the network, any sink that was connected to that node detects a source drop.

An NCE triggers the lean network services to stop $t_{\mathtt{Hello}}$ and send Hello_Get.

<table>
<tr><th>REQ</th><th>7.29 AL – Remote node sends ResourceMonitor_Status</th></tr>
<tr><td colspan="2">If a source drop occurs, that is, a resource becomes unavailable, a remote node shall send ResourceMonitor_Status to the address of the node that called RemoteNodeSync_StartResult.</td></tr>
</table>

13.2.5 Source drop

The availability of a connection is continuously verified by the data link layer. A source malfunction leads to automatic de-allocation of allocated bandwidth.

When a sink detects that one of its connections is no longer available, for example, due to network or source malfunction, this is referred to as source drop.

<table>
<tr><th>REQ</th><th>7.30 AL – MNC provides hardware signal for muting</th></tr>
<tr><td colspan="2">The MNC shall provide a hardware signal for muting the output that is associated with an In_Coupling.</td></tr>
</table>

13.3 Node discovery

13.3.1 General

<table>
<tr><th>REQ</th><th>7.31 AL – Root node performs node discovery</th></tr>
<tr><td colspan="2">The root node shall perform node discovery.</td></tr>
</table>

REQ	7.32 AL – Nodes provide their signature
Nodes shall provide their signature on request.	

13.3.2 Addresses

REQ	7.33 AL – Node resets the logical node address
When a node enters `s_NetInterface_Init` or receives `Init_Start`, it shall reset its logical node address to `0FFE`$_{16}$. NOTE The node becomes ready for receiving `Hello_Get`.	

REQ	7.34 AL – Remote node uses un-initialised logical node address
Before sending `Welcome_Result`, a remote node shall use the un-initialised logical node address.	

REQ	7.35 AL – Remote node uses logical node address from signature
For `Welcome_Result` and all following messages, a remote node shall use the logical node address that is included in the signature.	

13.3.3 Scanning the network

REQ	7.36 AL – Root node scans the network
The root node shall scan the network and mark as `available` those remote nodes that report a signature that matches the network descriptor.	

REQ	7.37 AL – Network supervisor starts node discovery
To start node discovery, the network supervisor shall send `Node_Discovery_Start_Request` to the lean network services.	

REQ	7.38 AL – Lean network services send Init_Start
The lean network services shall start node discovery by sending `Init_Start` to the blocking broadcast address.	

`Init_Start` is necessary because there may be situations in which the remote nodes do not recognize that network activity has ended. This is due to t_{Unlock}, which expires before a remote node recognizes the end of network activity. In the meantime, the root node might go through reset and without `Init_Start` the network would be in an inconsistent state.

REQ	7.39 AL – Lean network services broadcast Hello_Get
To initiate node discovery, the lean network services shall send `Hello_Get` to the blocking broadcast address.	

REQ	7.40 AL – Remote node sends Hello_Status
If a remote node with an un-initialised logical node address receives `Hello_Get`, it shall answer with `Hello_Status` with its current node position address in the corresponding parameter in the signature.	

A node position address consists of an offset plus the node position:

node position address = 0400_{16} + Node_Position

REQ	**7.41 AL – Reported node position address remains unchanged**
Between two consecutive Hello_Status reports, a remote node shall not change the reported node position address.	

REQ	**7.42 AL – Lean network services send signature to network supervisor**
The lean network services shall send the signature that they receive in Hello_Status to the network supervisor.	

13.3.4 Adding nodes to the network

REQ	**7.43 AL – Network supervisor determines addition to the network**
The network supervisor shall determine whether a discovered node is added to the network. NOTE The network supervisor compares the discovered nodes in the network to the expected nodes according to the network descriptor.	

REQ	**7.44 AL – Network supervisor marks remote node not_available**
For a remote node that is not added to the network, the network supervisor shall send Node_Availability_Set_Request to the lean network services, with Availability for that node set to not_available.	

REQ	**7.45 AL – Lean network services send Welcome_StartResult**
To add a remote node, the lean network services shall send Welcome_StartResult to the reported node position address of the remote node.	

REQ	**7.46 AL – Remote node sends Welcome_Result**
A remote node shall answer Welcome_StartResult with Welcome_Result if its signature matches.	

REQ	**7.47 AL – Network supervisor marks remote node as available**
When the network supervisor receives Node_Welcome_Response(success) for a remote node, it shall send Node_Availability_Set_Request to the lean network services, with Availability for that node set to available.	

Hello_Get is sent every time t_{Hello} expires.

REQ	**7.48 AL – Nodes that are already added do not answer**
After receiving Welcome_StartResult, a remote node shall not answer Hello_Get requests.	

An example of a successful Hello/Welcome sequence is shown in Figure 7. The example is based on a MOST network that contains four nodes as shown in Figure 6.

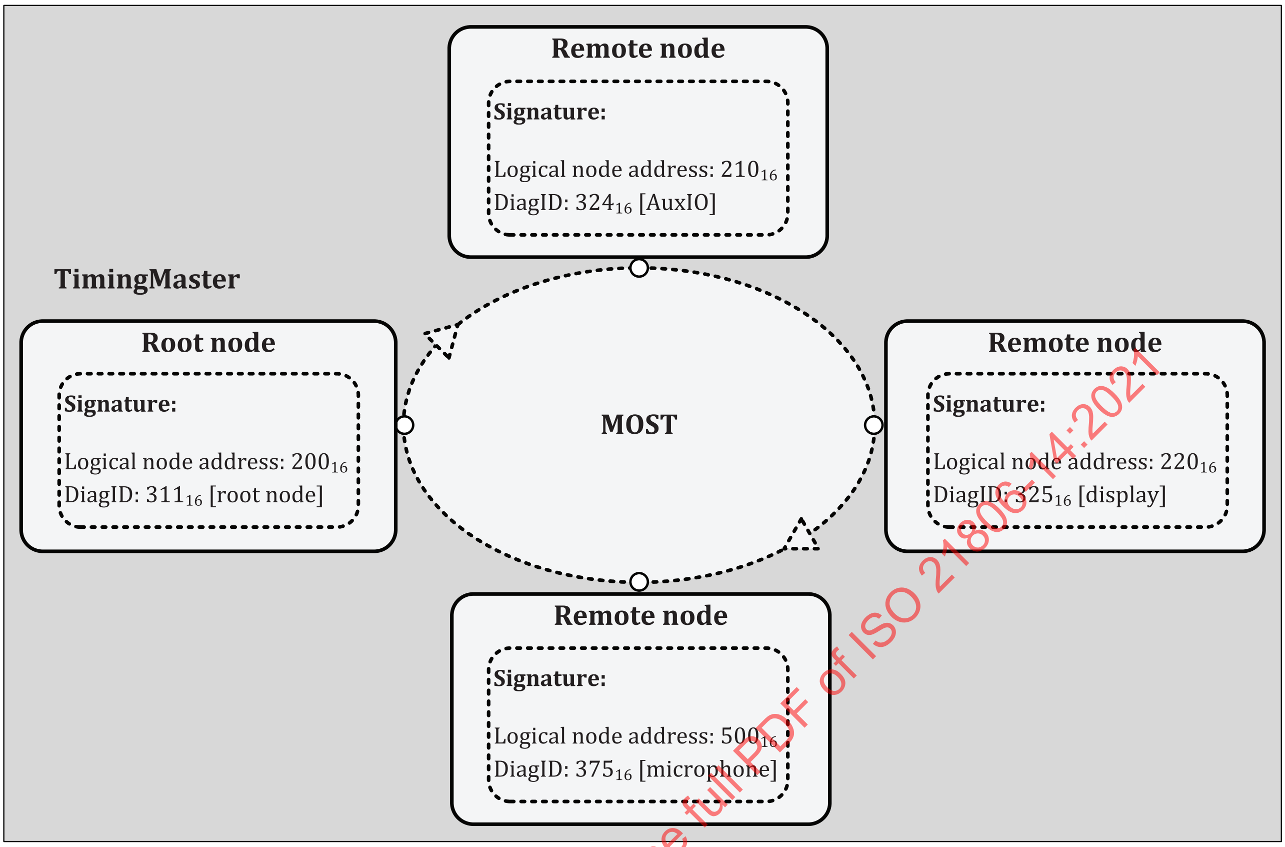
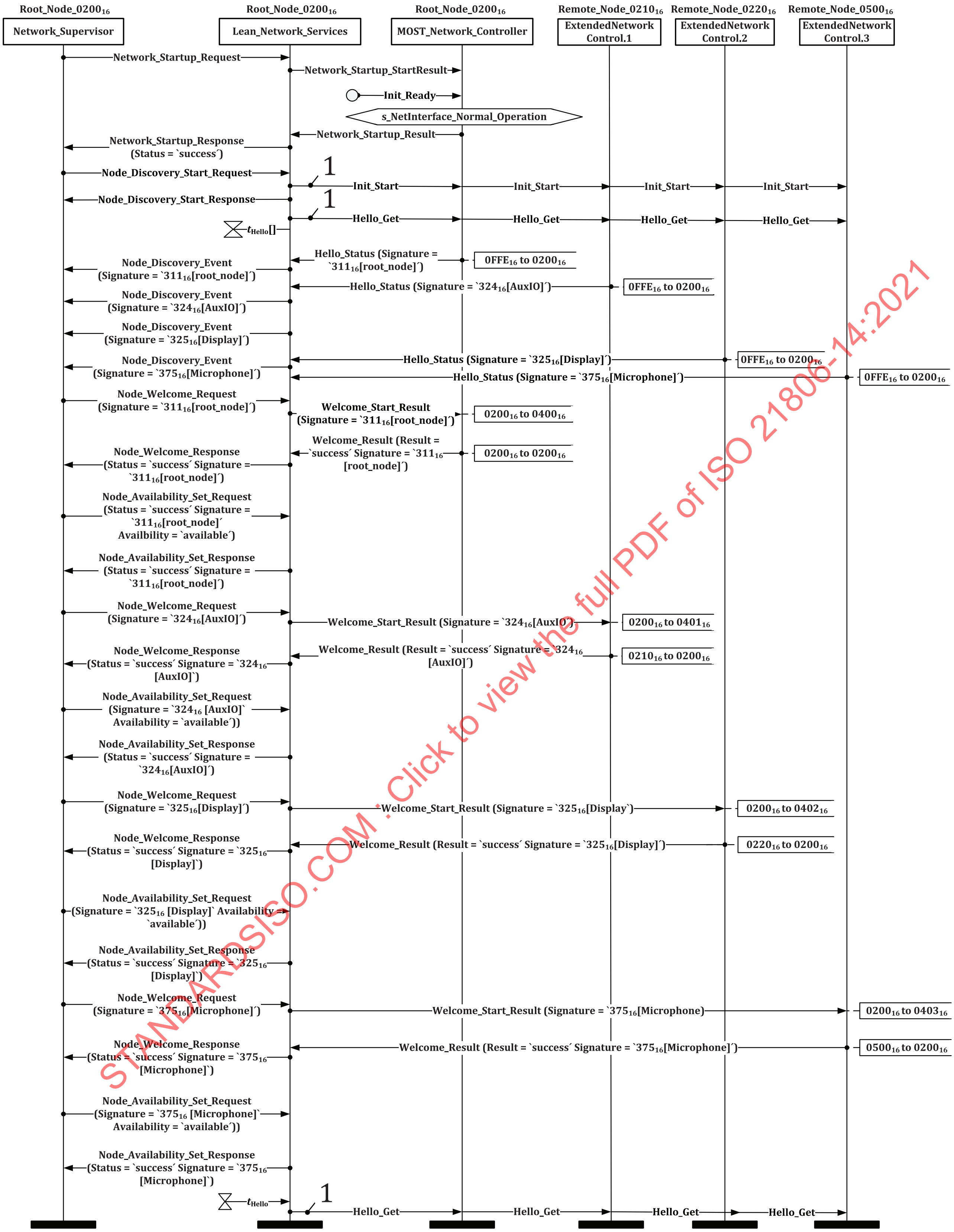

Remote node
Signature:
Logical node address: 210_{16}
DiagID: 324_{16} [AuxIO]
TimingMaster
Root node
Signature:
Logical node address: 200_{16}
DiagID: 311_{16} [root node]
MOST
Remote node
Signature:
Logical node address: 220_{16}
DiagID: 325_{16} [display]
Remote node
Signature:
Logical node address: 500_{16}
DiagID: 375_{16} [microphone]

Figure 6 — Node discovery example configuration


Root_Node_0200_{16}
Network_Supervisor
Root_Node_0200_{16}
Lean_Network_Services
Root_Node_0200_{16}
MOST_Network_Controller
Remote_Node_0210_{16}
ExtendedNetwork Control.1
Remote_Node_0220_{16}
ExtendedNetwork Control.2
Remote_Node_0500_{16}
ExtendedNetwork Control.3
Network_Startup_Request
Network_Startup_StartResult
Init_Ready
s_NetInterface_Normal_Operation
Network_Startup_Result
Network_Startup_Response (Status = `success´)
Node_Discovery_Start_Request
1
Init_Start
Init_Start
Init_Start
Init_Start
Node_Discovery_Start_Response
1
Hello_Get
Hello_Get
Hello_Get
Hello_Get
t_{Hello}[]
Hello_Status (Signature = `311_{16}[root_node]´)
$0FFE_{16}$ to 0200_{16}
Node_Discovery_Event (Signature = `311_{16}[root_node]´)
Hello_Status (Signature = `324_{16}[AuxIO]´)
$0FFE_{16}$ to 0200_{16}
Node_Discovery_Event (Signature = `324_{16}[AuxIO]´)
Node_Discovery_Event (Signature = `325_{16}[Display]´)
Hello_Status (Signature = `325_{16}[Display]´)
$0FFE_{16}$ to 0200_{16}
Node_Discovery_Event (Signature = `375_{16}[Microphone]´)
Hello_Status (Signature = `375_{16}[Microphone]´)
$0FFE_{16}$ to 0200_{16}
Node_Welcome_Request (Signature = `311_{16}[root_node]´)
Welcome_Start_Result (Signature = `311_{16}[root_node]´)
0200_{16} to 0400_{16}
Welcome_Result (Result = `success´ Signature = `311_{16} [root_node]´)
0200_{16} to 0200_{16}
Node_Welcome_Response (Status = `success´ Signature = `311_{16}[root_node]´)
Node_Availability_Set_Request (Status = `success´ Signature = `311_{16}[root_node]´ Availbility = `available´)
Node_Availability_Set_Response (Status = `success´ Signature = `311_{16}[root_node]´)
Node_Welcome_Request (Signature = `324_{16}[AuxIO]´)
Welcome_Start_Result (Signature = `324_{16}[AuxIO]´)
0200_{16} to 0401_{16}
Welcome_Result (Result = `success´ Signature = `324_{16} [AuxIO]´)
0210_{16} to 0200_{16}
Node_Welcome_Response (Status = `success´ Signature = `324_{16} [AuxIO]`)
Node_Availability_Set_Request (Signature = `$324_{16}$ [AuxIO]` Availability = `available´))
Node_Availability_Set_Response (Status = `success´ Signature = `324_{16}[AuxIO]´)
Node_Welcome_Request (Signature = `325_{16}[Display]´)
Welcome_Start_Result (Signature = `$325_{16}$[Display`)
0200_{16} to 0402_{16}
Welcome_Result (Result = `success´ Signature = `325_{16}[Display]´)
0220_{16} to 0200_{16}
Node_Welcome_Response (Status = `success´ Signature = `325_{16} [Display]`)
Node_Availability_Set_Request (Signature = `$325_{16}$ [Display]` Availability = `available´))
Node_Availability_Set_Response (Status = `success´ Signature = `325_{16} [Display]`)
Node_Welcome_Request (Signature = `375_{16}[Microphone]´)
Welcome_Start_Result (Signature = `375_{16}[Microphone]
0200_{16} to 0403_{16}
Welcome_Result (Result = `success´ Signature = `375_{16}[Microphone]´)
0500_{16} to 0200_{16}
Node_Welcome_Response (Status = `success´ Signature = `375_{16} [Microphone]`)
Node_Availability_Set_Request (Signature = `$375_{16}$ [Microphone]` Availability = `available´))
Node_Availability_Set_Response (Status = `success´ Signature = `375_{16} [Microphone]`)
t_{Hello}
1
Hello_Get
Hello_Get
Hello_Get
Hello_Get

Key

1 broadcast

Figure 7 — Node discovery example MSC

13.3.5 Reset detection

When a remote node is reset, it will answer to the next `Hello_Get` broadcast. In this situation, the signature received in `Hello_Status` is identical to a signature already stored in the root node.

REQ	**7.49 AL – Network supervisor resolves duplicates**
The network supervisor shall decide whether a duplicate signature belongs to a different node that erroneously uses an identical signature or if the reset of a remote node requires re-initialisation of all resources that include this node in currently established paths.	

REQ	**7.50 AL – Network supervisor requests uniqueness check**
To distinguish between a double entry and a reset, the network supervisor shall send `Check_Uniqueness_Request` with the currently stored signature to the lean network services.	

REQ	**7.51 AL – Lean network services send Signature_Get**
When the lean network services receive `Check_Uniqueness_Request`, they shall send `Signature_Get` to the logical node address found in the signature.	

If a remote node replies with `Signature_Status`, the node that has answered to `Hello_Get` is a duplicate. The lean network services report `Check_Uniqueness_Request` with `Status=error` to the network supervisor.

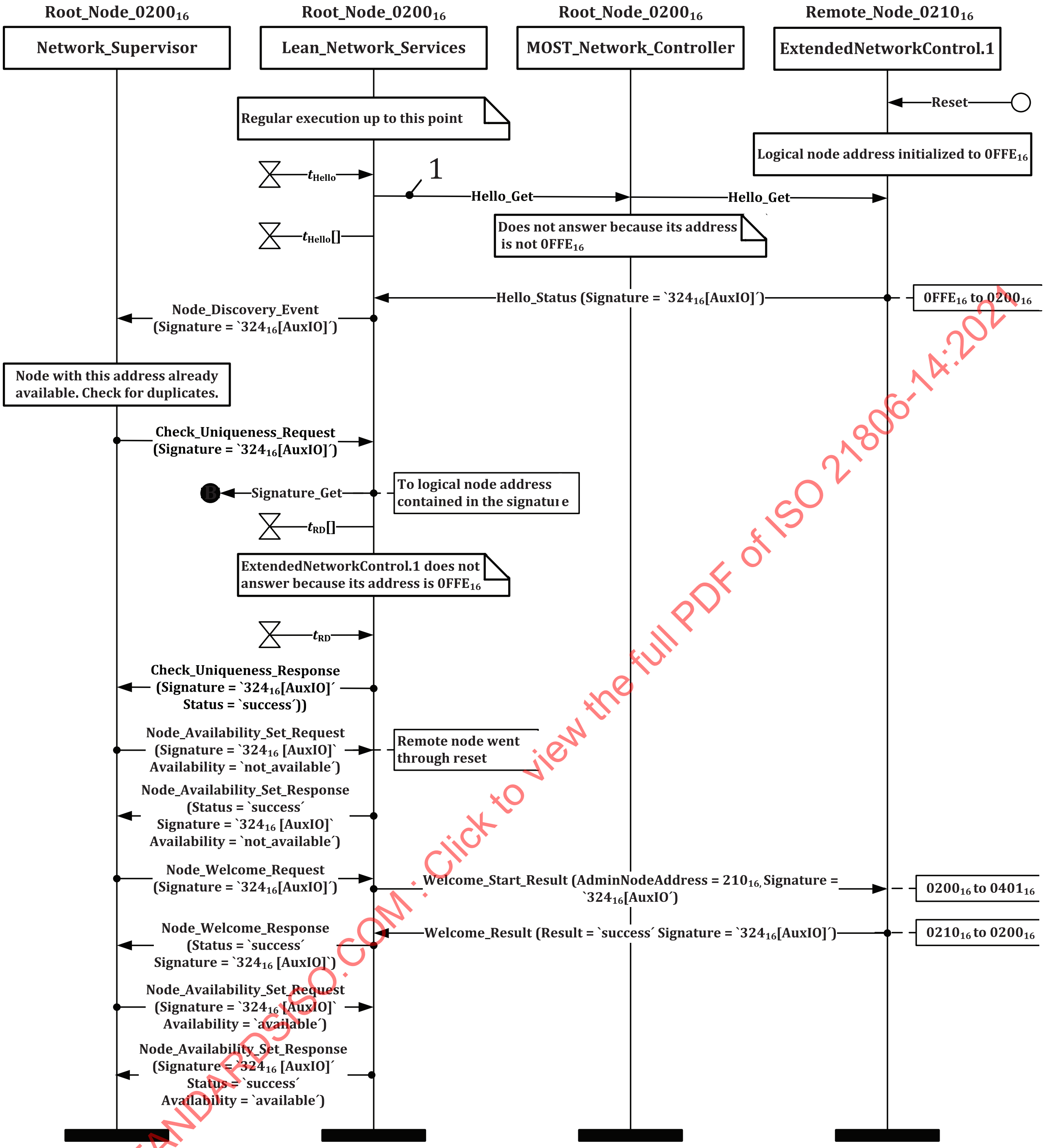
REQ	**7.52 AL – Network supervisor ignores duplicate**
When the network supervisor receives `Check_Uniqueness_Request` with `Status=error` for a node, the network supervisor shall ignore the node that triggered the uniqueness check.	

If the lean network services do not receive an answer within t_{RD} (RD – reset detection, see 15.3), it is assumed that the node that has answered to `Hello_Get`, underwent a reset condition. The lean network services report `Check_Uniqueness_Request` with `Status = success` to the network supervisor.

REQ	**7.53 AL – Network supervisor marks remote node not_available after reset**
When the network supervisor receives `Check_Uniqueness_Request` with `Status = success` for a node, the network supervisor shall send `Node_Availability_Set_Request` to the lean network services, with `Availability` for that node set to `not_available`. NOTE The connection management cleans up the associated paths (see 13.4.6).	

After a `Welcome` sequence, the node is available again and the connection management re-initialises all active paths.

Figure 8 illustrates how the reset of a remote node after node discovery (see Figure 7) is handled by the root node.

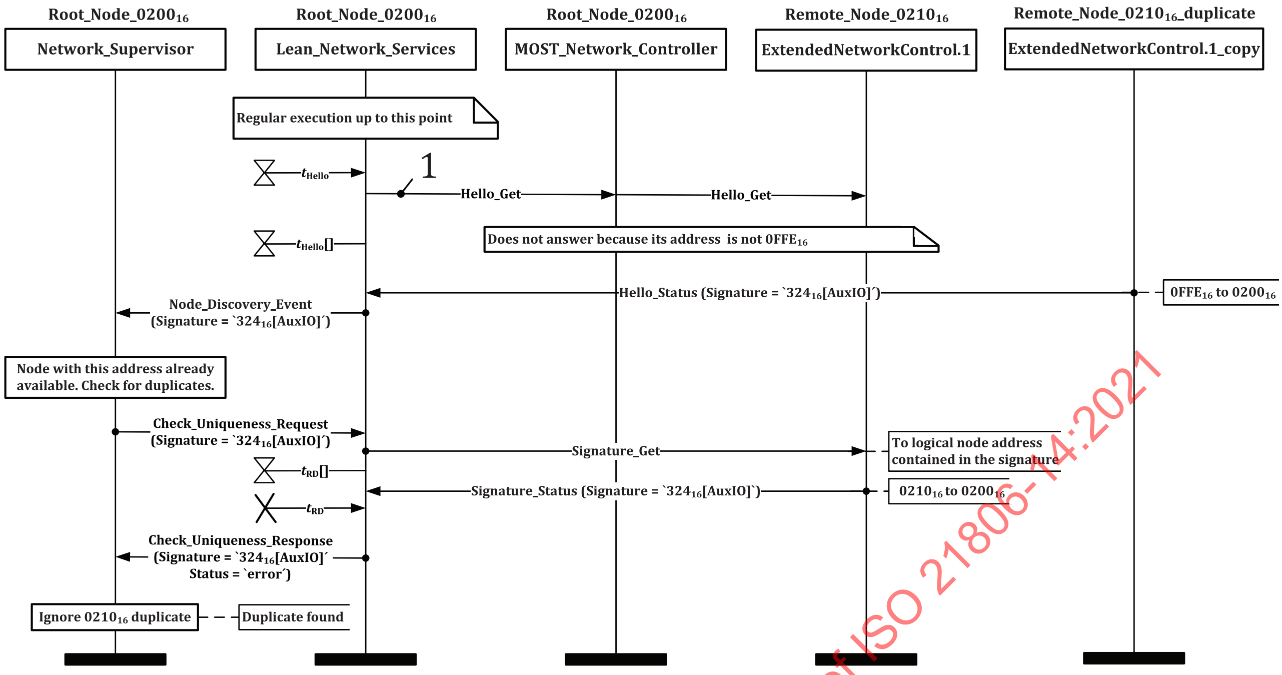

Root_Node_0200$_{16}$
Network_Supervisor
Root_Node_0200$_{16}$
Lean_Network_Services
Root_Node_0200$_{16}$
MOST_Network_Controller
Remote_Node_0210$_{16}$
ExtendedNetworkControl.1
Reset
Regular execution up to this point
Logical node address initialized to 0FFE$_{16}$
t_{Hello}
1
Hello_Get
Hello_Get
t_{Hello}[]
Does not answer because its address is not 0FFE$_{16}$
Hello_Status (Signature = `324$_{16}$[AuxIO]´)
0FFE$_{16}$ to 0200$_{16}$
Node_Discovery_Event (Signature = `324$_{16}$[AuxIO]´)
Node with this address already available. Check for duplicates.
Check_Uniqueness_Request (Signature = `324$_{16}$[AuxIO]´)
B
Signature_Get
To logical node address contained in the signature
t_{RD}[]
ExtendedNetworkControl.1 does not answer because its address is 0FFE$_{16}$
t_{RD}
Check_Uniqueness_Response (Signature = `324$_{16}$[AuxIO]´ Status = `success´))
Node_Availability_Set_Request (Signature = `324$_{16}$ [AuxIO]` Availability = `not_available´)
Remote node went through reset
Node_Availability_Set_Response (Status = `success´ Signature = `324$_{16}$ [AuxIO]` Availability = `not_available´)
Node_Welcome_Request (Signature = `324$_{16}$[AuxIO]´)
Welcome_Start_Result (AdminNodeAddress = 210$_{16}$, Signature = `324$_{16}$[AuxIO´)
0200$_{16}$ to 0401$_{16}$
Node_Welcome_Response (Status = `success´ Signature = `324$_{16}$ [AuxIO]`)
Welcome_Result (Result = `success´ Signature = `324$_{16}$[AuxIO]´)
0210$_{16}$ to 0200$_{16}$
Node_Availability_Set_Request (Signature = `324$_{16}$ [AuxIO]` Availability = `available´)
Node_Availability_Set_Response (Signature = `324$_{16}$ [AuxIO]´ Status = `success´ Availability = `available´)

Key

1 broadcast

Figure 8 — Reset detection example MSC

Figure 9 illustrates how the appearance of a node with duplicate logical node address after node discovery (see Figure 7) is handled by the root node.


Root_Node_0200$_{16}$
Network_Supervisor
Root_Node_0200$_{16}$
Lean_Network_Services
Root_Node_0200$_{16}$
MOST_Network_Controller
Remote_Node_0210$_{16}$
ExtendedNetworkControl.1
Remote_Node_0210$_{16}$_duplicate
ExtendedNetworkControl.1_copy
Regular execution up to this point
t_{Hello}
1
Hello_Get
Hello_Get
t_{Hello}[]
Does not answer because its address is not 0FFE$_{16}$
Hello_Status (Signature = `324$_{16}$[AuxIO]´)
0FFE$_{16}$ to 0200$_{16}$
Node_Discovery_Event
(Signature = `324$_{16}$[AuxIO]´)
Node with this address already available. Check for duplicates.
Check_Uniqueness_Request
(Signature = `324$_{16}$[AuxIO]´)
Signature_Get
To logical node address contained in the signature
t_{RD}[]
Signature_Status (Signature = `324$_{16}$[AuxIO]`)
0210$_{16}$ to 0200$_{16}$
t_{RD}
Check_Uniqueness_Response
(Signature = `324$_{16}$[AuxIO]´
Status = `error´)
Ignore 0210$_{16}$ duplicate
Duplicate found

Key

1 broadcast

Figure 9 — Duplicate address detection example MSC

13.4 Connection management

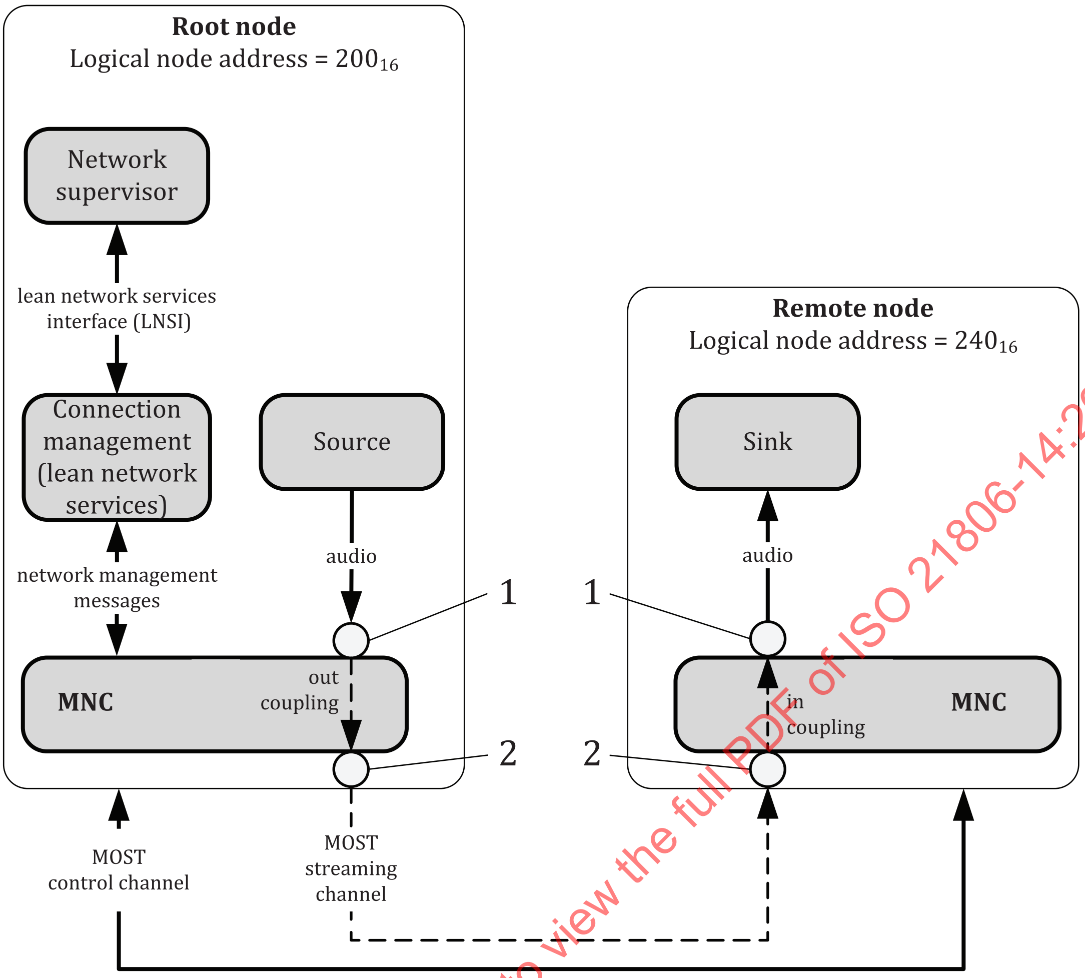
13.4.1 General

For transfer of synchronous or isochronous data, the MNC offers interfaces that are internal to the node, which are called streaming sockets. Towards the MOST network, other interfaces exist, which are called network sockets.

The network descriptor contains information for setting up streaming sockets and network sockets. Streaming sockets depend on additional vendor-specific data provided in the network descriptor.

REQ	7.54 AL – Connection management creates connections, couplings, and paths
The connection management shall rely on the network descriptor to: — create streaming sockets and network sockets; — establish connections; — create couplings and paths; — activate and deactivate paths; — monitor paths; and — recreate lost paths.	

Figure 10 shows a path configuration example for the transport of audio data.


Root node
Logical node address = 200$_{16}$
Network supervisor
lean network services interface (LNSI)
Connection management (lean network services)
Source
audio
network management messages
1
MNC
out coupling
2
MOST control channel
MOST streaming channel
Remote node
Logical node address = 240$_{16}$
Sink
audio
1
in coupling
MNC
2

Key

1	streaming socket
2	network socket

Figure 10 — Path configuration example

13.4.2 Sockets

The MNC is ordered to create sockets. When it reports `success`, the sockets exist and have a corresponding `ResourceHandle`. The connection management destroys sockets when network activity ends or after they are reported by `ResourceMonitor_Status` and `ResourceInvalidList_Status`.

REQ	7.55 AL – Resources are destroyed when network activity ends
When network activity ends, the root node or a remote node shall destroy its existing sockets and couplings.	

13.4.3 Couplings

Before creating a coupling, the network socket and the streaming socket are created. Couplings are characterized as `Out_Coupling` and `In_Coupling`, and exist in sources and sinks, respectively.

REQ	7.56 AL – In_Coupling structure
A sink shall create an `In_Coupling` between a network socket and a streaming socket.	

REQ	7.57 AL – Out_Coupling structure
A source shall create an `Out_Coupling` between a streaming socket and a network socket.	

REQ	7.58 AL – Connection management creates sockets
When a coupling is created, the connection management shall create the corresponding sockets.	

REQ	7.59 AL – Connection management allocates
When an `Out_Coupling` is created, the connection management shall allocate the bandwidth that is associated with the network socket.	

REQ	7.60 AL – Connection management destroys sockets
When the usage counter of a coupling is decreased to zero, the connection management: — shall remove the coupling; — shall destroy the corresponding sockets; and — shall set the `ResourceHandles` to `FFFF`$_{16}$.	

REQ	7.61 AL – Connection management deallocates
When the usage counter of an `Out_Coupling` is decreased to zero, the connection management shall deallocate the associated connection.	

REQ	7.62 AL – Connection management stores the Connection_Label
The connection management shall store the `Connection_Label` for a coupling in the corresponding network socket property.	

13.4.4 Paths

REQ	7.63 AL – Path structure
A path shall consist of an `In_Coupling` in one node and an `Out_Coupling` in another node, attached to a streaming connection.	

A coupling may be used in more than one path.

The network configuration contains `Activation` properties for paths, which can have the value `on` or `off`. The connection management switches between paths by activating and deactivating paths.

13.4.5 Activating paths

REQ	7.64 AL – Connection management creates paths
The connection management shall create configured paths automatically based on the network descriptor. NOTE This depends on the availability of the associated nodes. That means that they are in `s_NetInterface_Normal_Operation` state, are found during node discovery, and the network supervisor added them to the network.	

REQ	7.65 AL – Lean network services send RemoteNodeSync_StartResult
After the `Availability` property of a remote node is set to `available`, the lean network services shall send `RemoteNodeSync_StartResult` to the remote node.	

REQ	7.66 AL – Network supervisor activates paths
The network supervisor shall send `Path_Activation_Set_Request` with `Activation = on` to the lean network services to trigger the creation of the path.	

Consequently, connection management sets the `Activation` property of a path to `on`.

REQ	7.67 AL – Connection management creates Out_Coupling
After setting the `Activation` property of a path to `on`, the connection management shall create the `Out_Coupling`.	

REQ	7.68 AL – Connection management creates In_Coupling
When a path is created, if the `Out_Coupling` exists and the `Node` that is associated with the `In_Coupling` is available, the connection management shall create the `In_Coupling`.	

REQ	7.69 AL – Connection management increases coupling usage counter on activation
The connection management shall increase the usage counter of a coupling when a corresponding path is activated.	

REQ	7.70 AL – Path depends on valid ResourceHandles
A path shall be considered established when each coupling contains a valid `ResourceHandle`.	

13.4.6 Deactivating paths

REQ	7.71 AL – Connection management reacts to end of network activity
When network activity ends, connection management shall deactivate all paths.	

REQ	7.72 AL – Connection management deactivates paths
The network supervisor shall send `Path_Activation_Set_Request` with `Activation = off` to the lean network services to deactivate a path.	

REQ	7.73 AL – Connection management decreases coupling usage counter on deactivation
The connection management shall decrease the usage counter of a coupling by 1, when a corresponding path is deactivated.	

REQ	7.74 AL – Connection management handles paths for not_available nodes
When the `Availability` property of a node is set to `not_available`, the connection management shall clean up the paths that are associated with the node.	

13.4.7 Re-creation of lost paths

REQ	7.75 AL – Connection management re-creates paths
If a network event causes streaming connections and, consequently, paths to be removed, the connection management shall re-create them, when the necessary conditions are met. NOTE Depending on the error type, paths are automatically re-created.	

13.4.8 Monitoring

REQ	7.76 AL – Remote node indicates invalid resources
A remote node shall indicate invalid resources through `ResourceMonitor_Status`.	

REQ	7.77 AL – Connection management obtains list of invalid resources
If the lean network services receive `ResourceMonitor_Status(ActionRequired)`, the connection management shall obtain the list of invalid resources with `ResourceInvalidList_Get`.	

REQ	7.78 AL – Root node processes invalid list
If the lean network services receive `ResourceInvalidList_Status`, the connection management shall process the content of the invalid list.	

REQ	7.79 AL – Connection management removes invalid couplings and paths
The connection management shall remove couplings with invalid resources and paths that include remote nodes that are `not_available`.	

13.5 Network management message format

13.5.1 General

Network management communication relies on `N_CONTROL_DATA.SEND` and `N_CONTROL_DATA.RECEIVE`.

REQ	7.80 AL – TelID is zero
For `N_CONTROL_DATA.SEND` and `N_CONTROL_DATA.RECEIVE`, the value of the `TelID` shall be 0000_2. NOTE Setting the TelID to zero indicates that single transfer is used.	

13.5.2 Command and report reference

The commands and reports described in this subclause are used for network management, including connection management.

REQ	7.81 AL – Commands and reports
For commands and reports, the nodes in the network shall use the messages that are specified in Table 16 to Table 18.	

The parameters are specified in Annex A. The status of Annex A is normative.

NOTE The listed parameters serve the purpose of providing the minimum information that is required to perform an operation. In an actual implementation, it is possible that the parameter list differs in extent and parameter position.

Table 16 — Messages for commands and reports – Initialisation and network monitoring

Command/Report	Parameter	Description
`Init_Start`	---	This command resets the logical node address of a remote node. If the node is already added to the network, after `Init_Start`, it will respond to `Hello_Get` requests again.
`Init_Error`	`ErrorCode, ErrorInfo`	This report is used by the remote node to indicate an error.
`Hello_Get`	---	This command is used by the lean network services to request the `Signature` from a remote node during node discovery.
`Hello_Status`	`Signature`	This report is used by the remote node to provide its `Signature` to the root node.
`Hello_Error`	`ErrorCode, ErrorInfo`	This report is used by the remote node to indicate an error.
`Welcome_StartResult`	`AdminNodeAddress, Signature`	This command is used by the lean network services to notify the remote node that it was added to the MOST network.
`Welcome_Result`	`Result, Signature`	This report is used by the remote node to acknowledge its addition to the MOST network.
`Welcome_Error`	`ErrorCode, ErrorInfo`	This report is used by the remote node to indicate an error.
`Signature_Get`	---	The lean network services use this command to obtain the `Signature` of a remote node.
`Signature_Status`	`Signature`	This report is the response of a remote node to a `Signature_Get` request from the root node.
`Signature_Error`	`ErrorCode, ErrorInfo`	This report is used by the remote node to indicate an error.

Table 17 — Messages for commands and reports – Creation of sockets and couplings

Command/Report	Parameter	Description
`NetworkSocketCreate_StartResult`	`Direction, DataType, Bandwidth, ConnectionLabel`	The lean network services use this command to create a network socket.
`NetworkSocketCreate_Result`	`NetworkSocketHandle, ConnectionLabel`	This report is used by the remote node to confirm the creation of a network socket. `NetworkSocketHandle` is of the same data type as `ResourceHandle`.
`NetworkSocket_Error`	`ErrorCode, ErrorInfo`	This report is used by the remote node to indicate an error.
`StreamingSocketCreate_StartResult`	---	The lean network services use this command to create a streaming socket.
`StreamingSocketCreate_Result`	`StreamingSocketHandle`	This report is used by the remote node to confirm the creation of a streaming socket. `StreamingSocketHandle` is of the same data type as `ResourceHandle`.

Table 17 *(continued)*

Command/Report	Parameter	Description
StreamingSocketCreate_Error	ErrorCode, ErrorInfo	This report is used by the remote node to indicate an error.
CouplingCreate_StartResult	NetworkSocket, StreamingSocket	The lean network services use this command to create a coupling, based on the resource handle of a network socket and a streaming socket.
CouplingCreate_Result	CouplingHandle	This report is used by the remote node to confirm the creation of a network socket. CouplingHandle is of the same data type as ResourceHandle.
CouplingCreate_Error	ErrorCode, ErrorInfo	This report is used by the remote node to indicate an error.

Table 18 — Messages for commands and reports – Synchronization and resource invalidation

Command/Report	Parameter	Description
RemoteNodeSync_StartResult	Synchronization	This command is used by the lean network services to synchronize or unsynchronize a remote node. This command is used, for example, when connection management is started. It is sent to every node that is available.
RemoteNodeSync_Result	Synchronization	This report that is sent by a remote node contains the outcome of a request to synchronize or unsynchronize.
RemoteNodeSync_Error	ErrorCode, ErrorInfo	This report is used by the remote node to indicate an error.
ResourceInvalidList_Get	---	This command is used to request resources that were marked as invalid.
ResourceInvalidList_Status	ResourceHandleList	This report is used by the remote node to provide a list of invalid resources.
ResourceInvalidList_Error	ErrorCode, ErrorInfo	This report is used by the remote node to indicate an error.
ResourceMonitor_Set	Control	This command is used to reset the resource monitor to its default state and deactivate the hardware signal for muting.
ResourceMonitor_Status	State	This report is used by the remote node to indicate that resources are invalid. The lean network services use the ResourceInvalidList_Get command to obtain a list of unique resources handles for the invalid resources.
ResourceMonitor_Error	ErrorCode, ErrorInfo	This report is used by the remote node to indicate an error.
ResourceDestroy_StartResult	ResourceHandleList	The lean network services use this command to free resources. For this command, the ResourceHandleList shall not contain more than 10 entries.

Table 18 *(continued)*

Command/Report	Parameter	Description
ResourceDestroy_Result	---	This report is used by the remote node to confirm the execution of the ResourceDestroy_StartResult command.
ResourceDestroy_Error	ErrorCode, ErrorInfo	This report is used by the remote node to indicate an error.

13.5.3 Message IDs for commands and reports

REQ	7.82 AL – Message IDs for commands and reports
The message IDs for commands and reports for network management and connection management shall be in accordance with Table 19. NOTE For reports, the value of the byte 2 of the message ID is not relevant in a static configuration and, therefore, replaced by "--" in Table 19.	

Table 19 — Network management messages – Commands and reports

Operation	Message ID	Classification
RemoteNodeSync_StartResult	0000 2242$_{16}$	Command
RemoteNodeSync_Result	00-- 224C$_{16}$	Report
RemoteNodeSync_Error	00-- 224F$_{16}$	Report
NetworkSocketCreate_StartResult	0000 6112$_{16}$	Command
NetworkSocketCreate_Result	00-- 611C$_{16}$	Report
NetworkSocket_Error	00-- 611F$_{16}$	Report
StreamingSocketCreate_StartResult	0000 6912$_{16}$	Command
StreamingSocketCreate_Result	00-- 691C$_{16}$	Report
StreamingSocketCreate_Error	00-- 691F$_{16}$	Report
ResourceDestroy_StartResult	0000 8002$_{16}$	Command
ResourceDestroy_Result	00-- 800C$_{16}$	Report
ResourceDestroy_Error	00-- 800F$_{16}$	Report
ResourceInvalidList_Get	0000 8011$_{16}$	Command
ResourceInvalidList_Status	00-- 801C$_{16}$	Report
ResourceInvalidList_Error	00-- 801F$_{16}$	Report
ResourceMonitor_Set	0000 8020$_{16}$	Command
ResourceMonitor_Status	00-- 802C$_{16}$	Report
ResourceMonitor_Error	00-- 802F$_{16}$	Report
CouplingCreate_StartResult	0000 8712$_{16}$	Command
CouplingCreate_Result	00-- 871C$_{16}$	Report
CouplingCreate_Error	00-- 871F$_{16}$	Report
Hello_Get	0A00 2001$_{16}$	Command
Hello_Status	0A-- 200C$_{16}$	Report
Hello_Error	0A-- 200F$_{16}$	Report
Welcome_StartResult	0A00 2012$_{16}$	Command
Welcome_Result	0A-- 201C$_{16}$	Report
Welcome_Error	0A-- 201F$_{16}$	Report
Signature_Get	0A00 2021$_{16}$	Command
Signature_Status	0A-- 202C$_{16}$	Report

Table 19 *(continued)*

Operation	Message ID	Classification
`Signature_Error`	`0A-- 202F`$_{16}$	Report
`Init_Start`	`0A00 2030`$_{16}$	Command
`Init_Error`	`0A-- 203F`$_{16}$	Report

14 Diagnosis

14.1 General

The diagnosis capabilities depend on the physical layer.

REQ	7.83 AL – Root node controls diagnostic features
Control of diagnostic features shall be implemented in the root node.	

REQ	7.84 AL – No network and connection management during diagnosis
During diagnosis, the network supervisor shall stop network management and connection management.	

14.2 Full-duplex network diagnosis

Full-duplex network diagnosis is based on the coaxial physical layer. It is specified in ISO 21806-10, which shall be followed.

The network supervisor corresponds to the diagnosis evaluator in the MOST150 cPHY Duplex Network Diagnosis.[5] The lean network services correspond to the diagnostic worker.

Table 20 defines the message IDs that correspond the to the functions that are listed in the MOST150 cPHY Duplex Network Diagnosis.

NOTE For reports, the value of the second byte of the message ID is not relevant in a static configuration and, therefore, replaced by "--" in Table 20.

Table 20 — Full-duplex network diagnosis – Commands and reports

Operation	Message ID	Classification
`Hello_Get`	`0A00 2001`$_{16}$	Command
`Hello_Status`	`0A-- 200C`$_{16}$	Report
`Hello_Error`	`0A-- 200F`$_{16}$	Report
`Welcome_StartResult`	`0A00 2012`$_{16}$	Command
`Welcome_Result`	`0A-- 201C`$_{16}$	Report
`Welcome_Error`	`0A-- 201F`$_{16}$	Report
`EnablePort_StartResult`	`0A00 2102`$_{16}$	Command
`EnablePort_Result`	`0A-- 210C`$_{16}$	Report
`EnablePort_Error`	`0A-- 210F`$_{16}$	Report
`CableLinkDiagnosis_StartResult`	`0A00 2112`$_{16}$	Command
`CableLinkDiagnosis_Result`	`0A-- 211C`$_{16}$	Report
`CableLinkDiagnosis_Error`	`0A-- 211F`$_{16}$	Report

14.3 Half-duplex network diagnosis

Half-duplex network diagnosis is based on the balanced media physical layer. It is specified in ISO 21806-12, which shall be followed.

The network supervisor corresponds to the diagnosis evaluator in MOST50 bPHY half-duplex network diagnosis[6]. The lean network services correspond to the diagnostic worker.

Table 21 defines the message IDs that correspond to the functions that are listed in the half-duplex network diagnosis.

NOTE For reports, the value of the second byte of the message ID is not relevant in a static configuration and, therefore, replaced by "--" in Table 21.

Table 21 — Half-duplex network diagnosis – Commands and reports

Operation	Message ID	Classification
NetworkDiagnosisHalfDuplex_StartResult	0000 52E2$_{16}$	Command
NetworkDiagnosisHalfDuplex_Result	00-- 52EC$_{16}$	Report
NetworkDiagnosisHalfDuplex_Error	00-- 52EF$_{16}$	Report
EnableTX_StartResult	0A00 2232$_{16}$	Command
EnableTX_Result	0A-- 223C$_{16}$	Report
EnableTX_Error	0A-- 223F$_{16}$	Report
ReverseRequest_StartResult	0A00 2222$_{16}$	Command
ReverseRequest_Result	0A-- 222C$_{16}$	Report
ReverseRequest_Error	0A-- 222F$_{16}$	Report
NetworkDiagnosisHalfDuplexEnd_StartResult	00-- 52FC$_{16}$	Report
NetworkDiagnosisHalfDuplexEnd_Result	00-- 52FC$_{16}$	Report
NetworkDiagnosisHalfDuplexEnd_Error	00-- 52FF$_{16}$	Report

14.4 Network diagnosis using diagnosis line

Network diagnosis as described in this subclause is an optional feature.

Network diagnosis provides information about the status of the network.

For network diagnosis, all MOST devices in the network are connected by a diagnosis line.

REQ	**7.85 AL – Network owner designates diagnosis master**
The network owner shall designate one node in the MOST network as the diagnosis master.	

The diagnosis master collects and evaluates network diagnosis reports.

In network diagnosis, the following checks are performed:

- network activity detection,
- lock detection.

In Figure 11, the event “latch on” has two different meanings, depending on the use case:

- for network activity detection, it indicates network activity;
- for lock detection, it indicates locking onto the TimingMaster’s transmitted bit stream.

“Latch off” means the opposite:

- for network activity detection, it means that the network activity disappeared;

— for lock detection, it indicates losing lock on the TimingMaster's transmitted bit stream.

Figure 11 illustrates the network diagnosis states, which are provided by the NetInterface and which are valid for network activity detection as well as lock detection. "Off" is the initial state.

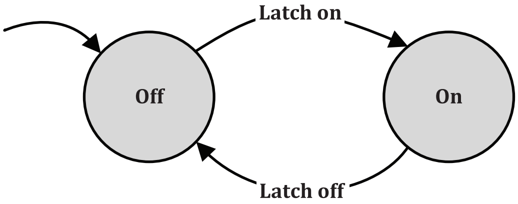


Figure 11 — Generic network diagnosis states

Network diagnosis is triggered by an external event (e.g. over electrical wake-up, signal on diagnosis line, or switch to power).

REQ	7.86 AL – Network supervisor performs network startup
If the MOST network is not running already when network diagnosis is triggered, the network supervisor shall request `cmd_Network_Startup`.	

The network owner is responsible for specifying the timing definitions of the test so that the test result is obtained after the network supervisor has initiated network startup by requesting `cmd_Network_Startup`.

REQ	7.87 AL – Network supervisor performs shutdown after completion
If the network was started up to perform network diagnosis and no qualified local wake-up event or network wake-up event occurs, the network supervisor shall shut down the network when reporting the results on the diagnosis line has started.	

REQ	7.88 AL – Nodes query the network diagnosis state
In the time frame that is specified by the network owner, each node shall query the current network diagnosis state and report "Off" or "On" over the diagnosis line.	

If network activity or lock—depending on the test— is identified, "On" is reported. If a ring break is present, "Off" is reported.

15 Timing definitions

15.1 Overview

Table 22 provides an overview of the relevant timing definitions.

Table 22 — Timers

Name	Minimum value	Typical value	Maximum value	Unit	Purpose
$t_{\texttt{Hello}}$	Network owner specific	---	Network owner specific	s	Cycle time for `Hello_Get` broadcasts by the lean network services
t_{RD}	Network owner specific	---	Network owner specific	ms	Time until the lean network services determine that a remote node went through reset

Table 22 *(continued)*

Name	Minimum value	Typical value	Maximum value	Unit	Purpose
$t_{\mathtt{LNS\_Termination}}$	Network owner specific	200	Network owner specific	ms	Termination of the lean network services regardless of running processes

15.2 Timer $t_{\mathbf{Hello}}$

15.2.1 Purpose

$t_{\mathtt{Hello}}$ controls the pause between cyclic transmissions of `Hello_Get` of the lean network services.

15.2.2 Start and stop conditions

REQ	7.89 AL – Lean network services start $t_{\mathbf{Hello}}$ with Hello_Get
The lean network services shall start $t_{\mathtt{Hello}}$ when they send `Hello_Get` to the blocking broadcast address.	

REQ	7.90 AL – Lean network services stop $\mathrm{t}_{\mathbf{Hello}}$ when node discovery is terminated
The lean network services shall stop $t_{\mathtt{Hello}}$ when node discovery is terminated. NOTE This is the case, for example, during diagnosis.	

REQ	7.91 AL – Lean network services stop $t_{\mathbf{Hello}}$ on NCE
When a network change event occurs, the lean network services shall stop $t_{\mathtt{Hello}}$ and send `Hello_Get` to the blocking broadcast address.	

15.2.3 Timer expiration

REQ	7.92 AL – Lean network services send Hello_Get when $t_{\mathbf{Hello}}$ expires
When $t_{\mathtt{Hello}}$ expires, the lean network services shall send `Hello_Get` to the blocking broadcast address.	

15.3 Timer $t_{\mathbf{RD}}$

15.3.1 Purpose

t_{RD} controls when the lean network services determine that a remote node went through reset.

15.3.2 Start and stop conditions

REQ	7.93 AL – Lean network services start $t_{\mathbf{RD}}$ with Signature_Get
The lean network services shall start t_{RD} when they send `Signature_Get` to a remote node.	

REQ	7.94 AL – Lean network services stop $t_{\mathbf{RD}}$ on Signature_Status
When the lean network services receive `Signature_Status` from the corresponding remote node, they shall: — stop t_{RD}, and — send `Check_Uniqueness_Response` with `Status = error` to the network supervisor.	

15.3.3 Timer expiration

REQ	7.95 AL – Lean network services report uniqueness success after t_{RD}
When t_{RD} expires, the lean network services shall send `Check_Uniqueness_Response` with `Status = success` to the network supervisor.	

15.4 Timer $t_{\text{LNS\_Termination}}$

15.4.1 Purpose

$t_{\text{LNS\_Termination}}$ enforces termination of the lean network services regardless of running processes.

15.4.2 Start and stop conditions

REQ	7.96 AL – Lean network services start $t_{\text{LNS\_Termination}}$
The lean network services shall start $t_{\text{LNS\_Termination}}$ when the lean network services receive `Stop_LNS_Request`.	

REQ	7.97 AL – Lean network services stop $t_{\text{LNS\_Termination}}$
The lean network services shall stop $t_{\text{LNS\_Termination}}$ when management processes of the lean network services are terminated and the associated network configuration data is cleared.	

15.4.3 Timer expiration

REQ	7.98 AL – Lean network services react to $t_{\text{LNS\_Termination}}$ expiration
When $t_{\text{LNS\_Termination}}$ expires, the lean network services shall reset running management processes and the associated network configuration data.	

16 Service interface definition to transport layer and network layer

The service interface to the transport layer and network layer is specified in ISO 21806-4 and shall be followed.

Annex A
(normative)

Parameters for LNSI and network management messages

A.1 General

The parameters described in this subclause are used in:

- requests, responses, or events for LNSI, or
- commands and reports for network management messages.

The details regarding restrictions for the different types used in the LNSI are contained in the corresponding XML schema definition in Annex B. The status of Annex B is informative.

A.2 Activation

Table A.1 specifies the `Activation`, which determines whether a path is activated or deactivated.

Table A.1 — `Activation` parameter definition

Data type	Mnemonic	Description
`enumeration`	`on`	Path active or activation is requested.
	`off`	Path not active or deactivation is requested.

A.3 AdminNodeAddress

Table A.2 specifies the `AdminNodeAddress` values of a remote node, which sets its logical node address to the provided value unless the value is $\texttt{FFFF}_{16}$. In this case, the logical node address that is contained in the `Signature` is used.

Table A.2 — `AdminNodeAddress` parameter definition

Data type	Valid values
`unsignedShort`	$\texttt{0F00}_{16}$ to $\texttt{0FEF}_{16}$, $\texttt{FFFF}_{16}$

A.4 Accessibility

Table A.3 specifies the `Accessibility`, which determines whether the network is accessible.

Table A.3 — `Accessibility` parameter definition

Data type	Mnemonic	Description
`enumeration`	`accessible`	Network is accessible.
	`not_accessible`	Network is not accessible.

A.5 AccessibilityInfo

Table A.4 specifies the AccessibilityInfo, which provides additional information about the parameter Accessibility.

Table A.4 — AccessibilityInfo parameter definition

Data type	Mnemonic	Description
enumeration	Regular	The network is in s_NetInterface_Off or s_NetInterface_Init state. It is pending to become available again. If TransitionCause is ErrorSystem, the network cannot be started before the error condition is cleared.
	Diagnosis	Diagnosis state entered.
	Unstable	Unlocks detected.
	Stable	Network is in stable lock.

A.6 Availability

Table A.5 specifies the Availability, which determines whether a node is available or not available.

Table A.5 — Availability parameter definition

Data type	Mnemonic	Description
enumeration	available	Node is available.
	not_available	Node is not available.

A.7 Bandwidth

Table A.6 specifies the required socket bandwidth in bytes. The maximum value depends on the currently available network resources.

Table A.6 — Bandwidth parameter definition

Data type	Valid values
unsignedShort	0001_{16} to $FFFF_{16}$

A.8 BoundaryDescriptor

Table A.7 specifies the BoundaryDescriptor, which determines the bandwidth available for streaming data transmission in number of quadlets per MOST frame.

Table A.7 — BoundaryDescriptor parameter definition

Data type	Bit rate	Valid values
unsignedByte	MOST50	0 to 29
	MOST150	0 to 93

A.9 Control

Table A.8 specifies the Control, that the resource monitor is reset.

Table A.8 — `Control` parameter definition

Data type	Valid values	Mnemonic	Description
`unsignedByte`	00_{16}	`Reset`	Resets the resource monitor.

A.10 DataType

Table A.9 specifies the data type for creating a network socket.

Table A.9 — `DataType` parameter definition

Data type	Valid values	Mnemonic	Description
`unsignedByte`	00_{16}	`Sync`	Synchronous streaming data type
	03_{16}	`AVPacketized`	A/V packetized isochronous streaming data type
	04_{16}	`QoSPacket`	Quality of service packet data type

A.11 DiagResult

The structure of `DiagResult` depends on the kind of diagnosis that is performed and should be adopted from the corresponding specification.

A.12 DiagnosisSelector

Table A.10 specifies the `DiagnosisSelector`, which determines the diagnostic process started.

Table A.10 — `DiagnosisSelector` parameter definition

Data type	Mnemonic	Description
`enumeration`	`half_duplex`	Half-duplex network diagnosis
	`full_duplex`	Full-duplex network diagnosis

A.13 Direction

Table A.11 specifies `Direction`, which indicates the direction of the data stream.

Table A.11 — `Direction` parameter definition

Data type	Valid values	Mnemonic	Description
`unsignedByte`	00_{16}	`In`	Socket receives data from the MOST network.
	00_{16}	`Out`	Socket transfers data to the MOST network.

A.14 ErrorCode

Table A.12 specifies the `ErrorCode`, which is the first parameter in an error message in network management message format.

Table A.12 — `ErrorCode` parameter definition

Data type	Valid values
`unsignedByte`	00_{16} to FF_{16}

A.15 ErrorInfo

Table A.13 specifies the ErrorInfo, which is the second parameter in an error message in network management message format. Whether ErrorInfo and, potentially, additional error information details are present depends on the ErrorCode.

Table A.13 — ErrorInfo parameter definition

Data type	Valid values
unsignedByte	00_{16} to FF_{16}

A.16 InitDataStructure

InitDataStructure contains manufacturer-specific information that is necessary to initialise the lean network services.

A.17 NodeAddress

Table A.14 specifies the NodeAddress, which is the logical node address of a node.

Table A.14 — NodeAddress parameter definition

Data type	Valid values
unsignedShort	0010_{16} to $00FF_{16}$, 0100_{16} to 013F, 0140_{16} to $02FF_{16}$,0500_{16} to $0FEF_{16}$, $FFFE_{16}$, $FFFF_{16}$

A.18 NetworkDescriptor

NetworkDescriptor is a parameter that is handed to the LNSI. It contains the network descriptor in a suitable format.

A.19 PacketBandwidth

Table A.15 specifies the current size of the bandwidth for packet data while the network is available. The maximum value depends on the currently available network resources. $FFFF_{16}$ is reported if the network is not available.

Table A.15 — PacketBandwidth parameter definition

Data type	Valid values
unsignedByte	0000_{16} to $FFFF_{16}$

A.20 Path

Table A.16 specifies the Path, which is the unique identifier of a path, when a request for activation or deactivation is handed to the LNSI.

Table A.16 — Path parameter definition

Data type	Valid values
string	No restrictions

A.21 ResourceHandleList

Table A.17 specifies the unique invalid ResourceHandleList.

Table A.17 — ResourceHandleList parameter definition

Data type	Elements
sequence[1 to 22]	ResourceHandle

A.22 ResourceHandle

Table A.18 specifies the unique ResourceHandle. FFFF_{16} is the END identifier. Receiving only the END identifier in the result indicates that there are no more invalid resources.

Table A.18 — ResourceHandle parameter definition

Data type	Valid values
unsignedShort	0000_{16} to FFFF_{16}

A.23 Result

Table A.19 specifies the Result of the comparison between the Signature provided in Welcome_StartResult and the Signature of the remote node.

Table A.19 — Result parameter definition

Data type	Valid values	Mnemonic	Description
unsignedByte	00_{16}	Success	The Signature matches.
	01_{16}	NoSuccess	The Signature does not match.

A.24 Signature

Table A.20 specifies the unique Signature, which corresponds to a node in the MOST network.

Table A.20 — Signature parameter definition

Data type	Elements
complexType	NodeAddress, GroupAddress, MACAddress_47to32, MACAddress_31to16, MACAddress_15to0, NodePositionAddress, DiagID, NumberOfPorts

Table A.21 specifies the elements for the Signature parameter.

Table A.21 — Signature elements of parameter definition

Name	Data type	Valid values	Description
NodeAddress	unsignedShort	0010_{16} to 00FF_{16}, 0100_{16} to 013F_{16}, 0140_{16} to 02FF_{16}, 0500_{16} to 0EFF_{16}, FFFF_{16}	The logical node address can be customized.
GroupAddress	unsignedShort	0300_{16} to 03C7_{16}, 03C9_{16} to 03FE_{16}	The group address can be customized.

Table A.21 *(continued)*

Name	Data type	Valid values	Description
MACAddress_47to32	unsignedShort	0000$_{16}$ to FFFF$_{16}$	Packet EUI-48[4] bits 47 to 32 of the MAC address, according to IEEE Guidelines for 48-Bit Global Identifier (EUI-48).[4] This parameter can be customized.
MACAddress_31to16	unsignedShort	0000$_{16}$ to FFFF$_{16}$	Packet EUI-48[4] bits 31 to 16 of the MAC address. This parameter can be customized.
MACAddress_15to0	unsignedShort	0000$_{16}$ to FFFF$_{16}$	Packet EUI-48[4] bits 15 to 0 of the MAC address. This parameter can be customized.
NodePositionAddress	unsignedShort	0400$_{16}$ to 043F$_{16}$	This is the NodePositionAddress obtained when sending Hello_Status.
DiagID	unsignedShort	0000$_{16}$ to FFFF$_{16}$	The DiagID parameter can be customized.
NumberOfPorts	unsignedByte	01$_{16}$ to FF$_{16}$	This is the number of MOST ports available. The maximum number of ports is restricted by the physical layer.

A.25 State

Table A.22 specifies the current state of the resource monitor, which can be either OK for the normal state when it requires no attention or ActionRequired when an action by the lean network services is necessary, for example, destroying invalid resources and finally resetting the resource monitor.

Table A.22 — State parameter definition

Data type	Valid values	Mnemonic	Description
unsignedByte	00$_{16}$	OK	Default state, no action is required.
	01$_{16}$	ActionRequired	Action is required.

A.26 Status

Table A.23 specifies the Status, which is contained in responses of the LNSI. Additional data may be provided to describe the status. For such data, no particular structure is mandated by the specification.

Table A.23 — Status parameter definition

Data type	Mnemonic	Description
enumeration	success	Requested operation was completed successfully.
	error	The request returned an error.

A.27 Synchronization

Table A.24 specifies the Synchronization, which unsynchronizes or synchronizes a remote node.

Table A.24 — `Synchronization` parameter definition

Data type	Valid values	Mnemonic	Description
`unsignedByte`	00_{16}	`UnSync`	Unsynchronize
	01_{16}	`Sync`	Synchronize

A.28 TransitionCause

Table A.25 specifies the `TransitionCause`, which indicates the cause of the network state transition from `Accessible` to `NotAccessible`. This parameter behaves like an event. Once reported, it is cleared to `NoTransition`. The default value is `NoTransition`.

Table A.25 — `TransitionCause` parameter definition

Data type	Mnemonic	Description
`enumeration`	`Normal`	Normal shutdown
	`ErrorSuddenSignalOff`	The network is shut down due to an error. In this case, the shutdown reason was a sudden signal off. No shutdown flag is present.
	`ErrorCriticalUnlock`	The network is shut down due to an error. In this case, the shutdown reason was a critical unlock. No shutdown flag is present.
	`ErrorMNC`	The network is shut down due to an MNC error.
	`NoTransition`	No transition

A.29 VisibleNodes

Table A.26 specifies the content of the visible nodes field in the network frame if the network is accessible. This value is updated when an NCE occurs. $\texttt{FF}_{16}$ is reported if the network is not accessible.

Table A.26 — `VisibleNodes` parameter definition

Data type	Valid values
`unsignedByte`	`0 to 63,` $\texttt{FF}_{16}$

Annex B
(informative)

XSD for validation of XML

B.1 Graphical representation of network descriptor

This subclause provides a graphical overview of the representation of the network descriptor XML structure. The structure is based on XML Schema Part 2: Datatypes Second Edition[7].

Figure B.1 shows an overview of the structure, with focus on nodes.

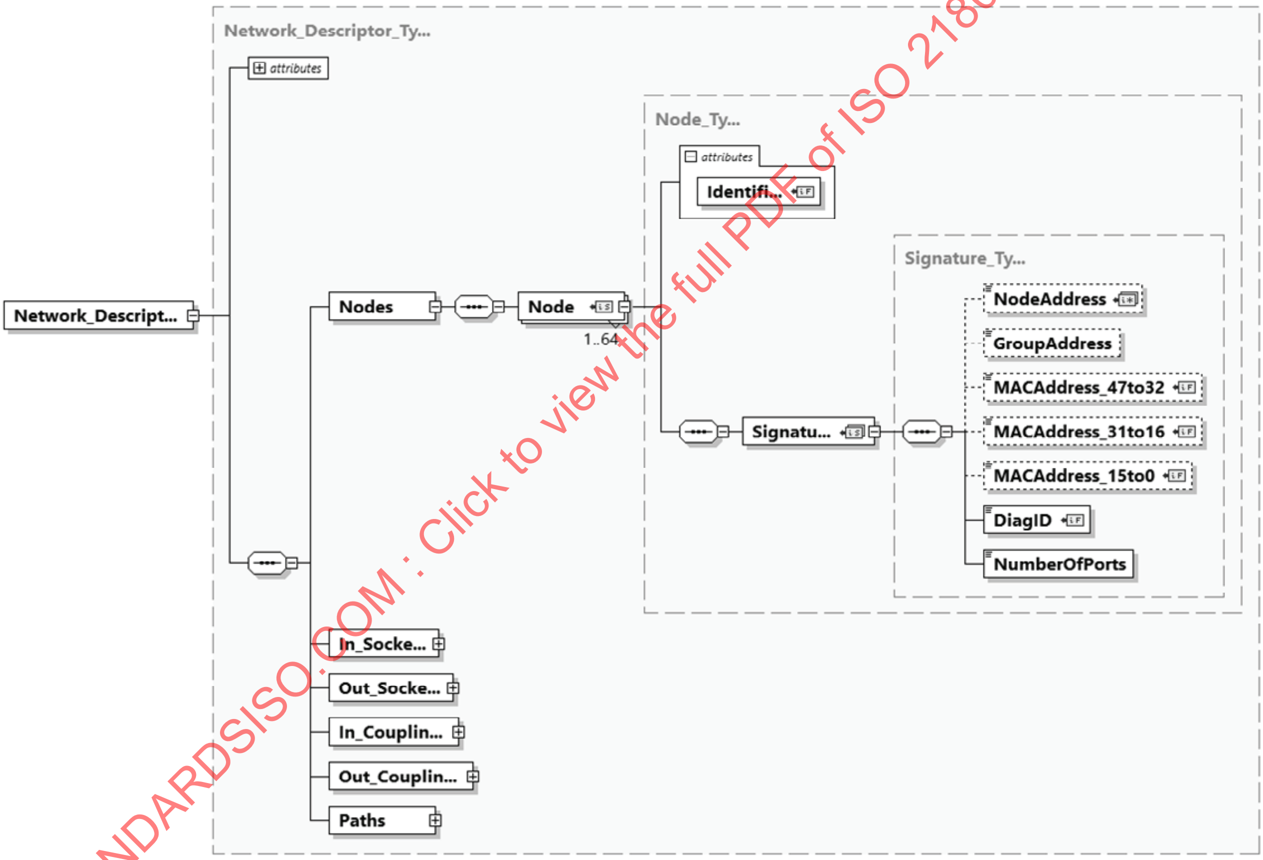


Figure B.1 — Network descriptor XML schema — Nodes

Figure B.2 shows an overview of the structure, with focus on sockets.