International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

G.9956

(11/2011)

SERIES G: TRANSMISSION SYSTEMS AND MEDIA, DIGITAL SYSTEMS AND NETWORKS

Access networks – In premises networks

Narrowband orthogonal frequency division multiplexing power line communication transceivers – Data link layer specification

Recommendation ITU-T G.9956

ITU-T G-SERIES RECOMMENDATIONS

TRANSMISSION SYSTEMS AND MEDIA, DIGITAL SYSTEMS AND NETWORKS

INTERNATIONAL TELEPHONE CONNECTIONS AND CIRCUITS	G.100–G.199
GENERAL CHARACTERISTICS COMMON TO ALL ANALOGUE CARRIER-TRANSMISSION SYSTEMS	G.200–G.299
INDIVIDUAL CHARACTERISTICS OF INTERNATIONAL CARRIER TELEPHONE SYSTEMS ON METALLIC LINES	G.300–G.399
GENERAL CHARACTERISTICS OF INTERNATIONAL CARRIER TELEPHONE SYSTEMS ON RADIO-RELAY OR SATELLITE LINKS AND INTERCONNECTION WITH METALLIC LINES	G.400–G.449
COORDINATION OF RADIOTELEPHONY AND LINE TELEPHONY	G.450–G.499
TRANSMISSION MEDIA AND OPTICAL SYSTEMS CHARACTERISTICS	G.600–G.699
DIGITAL TERMINAL EQUIPMENTS	G.700–G.799
DIGITAL NETWORKS	G.800–G.899
DIGITAL SECTIONS AND DIGITAL LINE SYSTEM	G.900–G.999
MULTIMEDIA QUALITY OF SERVICE AND PERFORMANCE – GENERIC AND USER-RELATED ASPECTS	G.1000–G.1999
TRANSMISSION MEDIA CHARACTERISTICS	G.6000–G.6999
DATA OVER TRANSPORT – GENERIC ASPECTS	G.7000–G.7999
PACKET OVER TRANSPORT ASPECTS	G.8000–G.8999
ACCESS NETWORKS	G.9000–G.9999
Metallic access networks	G.9700–G.9799
Optical line systems for local and access networks	G.9800–G.9899
In premises networks	**G.9900–G.9999**

For further details, please refer to the list of ITU-T Recommendations.

Recommendation ITU-T G.9956

Narrowband orthogonal frequency division multiplexing power line communication transceivers –
Data link layer specification

Summary

Recommendation ITU-T G.9956 contains the data link layer specification for narrowband OFDM power line communications transceivers for communications via alternating current and direct current electric power lines over frequencies below 500 kHz. This Recommendation supports indoor and outdoor communications over low voltage lines, medium voltage lines, through transformer low-voltage to medium-voltage and through transformer medium-voltage to low-voltage power lines in both urban and in long distance rural communications. This Recommendation addresses grid to utility meter applications, advanced metering infrastructure (AMI), and other Smart Grid applications such as the charging of electric vehicles, home automation and home area networking (HAN) communications scenarios.

This version integrates Recommendation ITU-T G.9956 (11/2011), its Corrigendum 1 (09/2012) and its Amendment 1 (10/2012).

History

Edition	Recommendation	Approval	Study Group	Unique ID[*]
1.0	ITU-T G.9956	2011-11-06	15	11.1002/1000/11147-en
1.1	ITU-T G.9956 (2011) Cor. 1	2012-09-21	15	11.1002/1000/11528-en
1.2	ITU-T G.9956 (2011) Amd. 1	2012-10-29	15	11.1002/1000/11816-en

[*] To access the Recommendation, type the URL http://handle.itu.int/ in the address field of your web browser, followed by the Recommendation's unique ID. For example, http://handle.itu.int/11.1002/1000/11830-en.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at http://www.itu.int/ITU-T/ipr/.

Table of Contents

Recommendation ITU-T G.9956

Narrowband orthogonal frequency division multiplexing power line communication transceivers – Data link layer specification

1 Scope

This Recommendation contains the data link layer specification for narrowband orthogonal frequency division multiplexing (OFDM) power line communications transceivers for communications via alternating current and direct current electric power lines over frequencies below 500 kHz. This Recommendation supports indoor and outdoor communications over low voltage lines, medium voltage lines, through transformer low-voltage to medium-voltage and through transformer medium-voltage to low-voltage power lines in both urban and in long distance rural communications. This Recommendation addresses grid to utility meter applications, advanced metering infrastructure (AMI), and other Smart Grid applications such as the charging of electric vehicles, home automation and home area networking (HAN) communications scenarios.

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

[ITU-T G.9955] Recommendation ITU-T G.9955 (2011), *Narrowband orthogonal frequency division multiplexing power line communication transceivers - Physical layer specification.*

[IEEE 802.3] IEEE 802.3 (2005), *Local and metropolitan area networks – Specific requirements. Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) access method and Physical Layer specifications*.

[IETF RFC 791] IETF RFC 791 (1981), *Internet Protocol, DARPA Internet Program, Protocol Specification.*

[IETF RFC 2460] IETF RFC 2460 (1998), *Internet Protocol, Version 6 (IPv6) Specification.*

[IETF RFC 2464] IETF RFC 2464 (1998), *Transmission of IPv6 Packets over Ethernet Networks*.

[IETF RFC 4291] IETF RFC 4291(2006), *IP Version 6 Addressing Architecture*.

[IETF RFC 4861] IETF RFC 4861 (2007), *Neighbor Discovery for IP version 6 (IPv6).*

[IETF RFC 4862] IETF RFC 4862 (2007), *IPv6 Stateless Address Autoconfiguration*.

[IETF RFC 4944] IETF RFC 4944 (2007), *Transmission of IPv6 Packets over IEEE 802.15.4 Networks.*

[IETF RFC 6282] IETF RFC 6282 (2011), *Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks.*

[NIST PUB 197] NIST FIPS PUB 197 (2001), *Advanced Encryption Standard (AES).*

[NIST SP 800-38C] NIST SP 800-38C (2004), *Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality*.

[EUI-64] EUI-64™ (in force), *Guidelines for 64-bit Global Identifier, IEEE.* <http://standards.ieee.org/develop/regauth/tut/eui64.pdf>

3 Definitions

This Recommendation defines the following terms:

3.1 alien domain: Any group of non-ITU-T G.9955/9956 nodes connected to the same or a different medium (wired or wireless) operating in a close proximity. These domains can be used as backbones to the ITU-T G.9955/9956 network or as separate networks. The L3 bridging function to an alien domain, as well as coordination with an alien domain to avoid mutual interference is beyond the scope of ITU-T G.9955/9956.

3.2 advanced metering infrastructure (AMI): Primary means for utilities to interact with meters on customer sites. In addition to basic meter reading, AMI provides two-way communications allowing to collect and analyse energy usage, and interact with advanced devices such as electricity meters, gas meters, heat meters, and water meters, through various communication media.

3.3 bandplan: A specific range of the frequency spectrum that is defined by a lower frequency and upper frequency.

3.4 bridge to alien domain/network: An application device implementing an L2- or L3-bridging function to interconnect a ITU-T G.9955/9956 domain to an alien domain (or alien network). Bridging to alien domains/networks is beyond the scope of ITU-T G.9955/9956.

3.5 broadcast: A type of communication where a node sends the same frame simultaneously to all other nodes in the home network or in the domain.

3.6 carrier sense (CRS): Generated by the receiver, CRS indicates that the medium is busy, i.e., a PHY frame, or sequence of PHY frames, or a special signal (e.g., INUSE, PR) is currently transmitted on the medium by another node. CRS may be either a physical carrier sense signal or a virtual carrier sense indicator.

– Physical carrier sense is generated by analysing physical signals present on the medium.

– Virtual carrier sense is generated based on the information on the PHY frame duration or PHY frame sequence duration derived from the frame header or communicated to a node by other means (e.g., in another frame).

3.7 ceiling(x): A function that returns the minimum integer value bigger or equal to x.

3.8 channel: A transmission path between nodes. One channel is considered to be one transmission path. Logically a channel is an instance of communications medium used for the purpose of passing data between two or more nodes.

3.9 data: Bits or bytes transported over the medium or via a reference point that individually convey information. Data includes both user (application) data and any other auxiliary information (overhead, including control, management, etc.). Data does not include bits or bytes that, by themselves, do not convey any information, such as preamble.

3.10 data rate: The average number of data elements (bits, bytes, or frames) communicated (transmitted) in a unit of time. Depending on the data element, data bit rate, data byte rate, and symbol frame rate may be used. The usual unit of time for data rate is 1 second.

3.11 domain: A part of a ITU-T G.9955/9956 home network comprising a domain master and all those nodes that are registered with this same domain master. In the context of this

Recommendation, use of the term 'domain' without a qualifier means ' ITU-T G.9955/9956 domain', and use of the term 'alien domain' means 'non- ITU-T G.9955/9956 domain'.

3.12 **domain access point (DAP)**: The unique node in centralized mode (CM) that supports relay functionality through which all nodes communicate.

3.13 **domain ID**: A unique identifier of a domain.

3.14 **domain master (DM)**: A node that manages (coordinates) all other nodes of the same domain. Domain master is a node with extended management capabilities that enables to form, control, and maintain the nodes associated with its domain.

3.15 **end-node**: A node that is not a domain master; all nodes in the domain except the domain master are end-nodes.

3.16 **global master (GM)**: A function that provides coordination between different domains of the same network (such as communication resources, priority setting, policies of domain masters, and interference mitigation). A GM may also convey management functions initiated by the remote management system. Detailed specification and use of this function is for further study.

3.17 **hidden node**: A node that cannot communicate directly with some other nodes within a domain. A hidden node may be able to communicate with another node using a relay node.

3.18 **network**: Two or more nodes that can communicate with each other either directly or through a relay node at the physical layer, or through an inter-domain bridge above the physical layer.

3.19 **home area network (HAN)**: A network at customer premises that interconnects customer-owned devices for energy management and communications with the utility.

3.20 **inter-domain bridge (IDB)**: A bridging function to interconnect nodes of two different domains.

3.21 **inter-network bridge (INB)**: A bridging function to interconnect nodes of two different ITU-T G.9955/9956 networks.

3.22 **latency**: A measure of the delay from the instant when the last bit of a frame has been transmitted through the assigned reference point of the transmitter protocol stack to the instant when a whole frame reaches the assigned reference point of receiver protocol stack. Mean and maximum latency estimations are assumed to be calculated on the 99th percentile of all latency measurements. If retransmission is required for a frame, the retransmission time is a part of latency for the protocol reference points above the MAC.

3.23 **logical (functional) interface**: An interface in which the semantic, syntactic, and symbolic attributes of information flows are defined. Logical interfaces do not define the physical properties of signals used to represent the information. It is defined by a set of primitives.

3.24 **management overhead**: A part of the traffic used for management purposes (any used bandwidth that does not contain application data).

3.25 **medium**: A wire-line facility allowing physical connection between nodes. Nodes connected to the same medium may communicate on the physical layer, and may interfere with each other unless they use orthogonal signals (e.g., different frequency bands, different time periods).

3.26 **multicast**: A type of communication when a node sends the same frame simultaneously to more than one node in the network.

3.27 **net data rate**: The data rate at the A-interface of the transceiver reference model.

3.28 **node**: Any network device that contains an ITU-T G.9955/9956 transceiver. In the context of this Recommendation, use of the term 'node' without a qualifier means ' ITU-T G.9955/9956 node', and use of the term 'alien node' means 'non- ITU-T G.9955/9956 node'. Additional qualifiers (e.g., 'relay') may be added to either 'node' or 'alien node'.

3.29 node ID: A unique identifier allocated to a node within the domain.

3.30 peer-to-peer communication: A type of communication within a domain in which direct signal traffic is established between nodes with no relay nodes.

3.31 physical interface: An interface defined in terms of physical properties of the signals used to represent the information transfer. A physical interface is defined by signal parameters like power (power spectrum density), timing, and connector type.

3.32 primitives: Variables and functions used to define logical interfaces and reference points.

3.33 priority: Medium access (MA) priority is a value assigned to a particular frame(s) that determines the relative importance of transmitting the frame(s) during the upcoming opportunity to use the medium. The protocol data unit (PDU) priority is a value assigned to a particular protocol data unit that determines the relative order of queuing of this PDU for transmission over the medium.

3.34 quality of service (QoS): A set of quality requirements on the communications in the network.

3.35 reference point: A location in a signal flow, either logical or physical, that provides a common point for observation and/or measurement of the signal flow.

3.36 registration: The process used by a node to join the domain.

3.37 relayed communication: A type of communication within a domain, in which a node can communicate with another node through a relay node. A relay node receives a signal from a node and forwards it to the addressee node.

3.38 relay node: A node supporting relay functionality that acts as an intermediary node, through which other nodes of the same domain can pass their signal traffic (data, control, or management).

3.39 subcarrier (OFDM subcarrier): The center frequency of each OFDM sub-channel onto which bits may be modulated for transmission over the sub-channel.

3.40 symbol (OFDM symbol): A fixed time-unit of an OFDM signal. An OFDM symbol consists of multiple sine-wave signals or subcarriers. Each subcarrier can be modulated by certain number of data bits and transmitted during the fixed time called symbol period.

3.41 symbol rate: The rate, in symbols per second, at which OFDM symbols are transmitted by a node onto a medium. Symbol rate is calculated only for time periods of continuous transmission.

3.42 throughput: The amount of data transferred from the A-interface of a source node to the A-interface of a destination node over some time interval, expressed as the number of bits per second.

3.43 transmission overhead: A part of the overhead used to support transmission over the line (e.g., samples of cyclic prefix, inter-frame gaps, and silent periods).

3.44 unicast: A type of communication when a node sends the frame to another single node.

3.45 utility access network (UAN): A power line communications network that operates under control of the electric utility over the utility-owned electricity distribution lines, and provides communications between the utility and the utility-controlled devices and network infrastructure at the customer premises.

4 Abbreviations

This Recommendation uses the following abbreviations:

AC	Alternating Current
ACK	Acknowledge
ADP	Application Data Primitives
AE	Application Entity
AES	Advanced Encryption Standard
AMI	Advanced Metering Infrastructure
APC	Application Protocol Convergence
ARQ	Automatic Repeat Request
BAT	Bit Allocation Table
BPSK	Binary Phase Shift Keying
CM	Centralized Mode
CP	Customer Premise
CSMA-CA	Carrier Sense Multiple Access-Collision Avoidance
CRC	Cyclic Redundancy Check
DAP	Domain Access Point
DLL	Data Link Layer
DM	Domain Master
DSL	Digital Subscriber Line
EMS	Energy Management System
ESI	Energy Service Interface
FCS	Frame Check Sequence
FEC	Forward Error Correction
GM	Global Master
HAN	Home Area Network
HCS	Header Check Sum
IDB	Inter-Domain Bridge
INB	Inter-Network Bridge
IPv4	Internet Protocol, Version 4
IPv6	Internet Protocol, Version 6
LLC	Logical Link Control
LPDU	LLC Protocol Data Using
LSB	Least Significant Bit
MAC	Medium Access Control
MDI	Medium-Dependent Interface
MIB	Management Information Base

MPDU	MAC Protocol Data Unit
MSB	Most Significant Bit
NACK	Negative Acknowledge
OFDM	Orthogonal Frequency Division Multiplexing
P2P	Peer-To-Peer Communication
PFH	PHY Frame Header
PHY	Physical Layer
PLC	Power Line Communications
PMI	Physical Medium-Independent Interface
PMA	Physical Medium Attachment
PMD	Physical Medium Dependent
PPDU	PHY Protocol Data Unit
PPM	Parts Per Million
PSD	Power Spectral Density
QoS	Quality of Service
RCM	Robust Communication Mode
RMS	Root Mean Square
RS	Reed-Solomon
SNR	Signal to Noise Ratio
UAN	Utility Access Network
UM	Unified Mode

5 Network architecture and reference models

5.1 Network architecture and topology

5.1.1 Basic principles of ITU-T G.9956 networking

See clause 5.1.1 of [ITU-T G.9955].

5.1.2 Energy management network architecture and topology

The energy management network architecture and topology, including a description of a generic UAN architecture and a generic HAN architecture are specified in clause 5.1.2, clause 5.1.2.1 and clause 5.1.2.2 of [ITU-T G.9955], respectively.

The rules of UAN and HAN domain operation are defined in clause 5.1.3.

5.1.3 Domain

5.1.3.1 General rules of operation

Both HAN and UAN domains, as depicted in Figure 5-1 of [ITU-T G.9955], comprise of a domain master and one or more end-nodes. Nodes may be of basic capabilities (usually, majority of end nodes) and of extended capabilities such as a relay or a domain master, or a domain access point (DAP).

The following rules apply for any domain:

1. The function of the domain master is to manage and coordinate operation of all nodes in its domain. The specific functions of the domain master and details of the particular management procedures are for further study.
2. There shall be one and only one active domain master per domain at a given time.
3. Nodes of the same domain communicate with each other using general rules defined in clause 5.1.1.

   NOTE – Nodes of different domains can communicate if applications employing these nodes have compatible security requirements. Details are for further study.
4. Domains of independent HANs or UANs may interfere with each other. Methods of coordination between domains of independent HANs or UANs are for further study.
5. Nodes are not required to be domain master capable or relay-capable or DAP-capable. That is, some nodes may not support the functionality necessary to become a domain master or a relay or a DAP.
6. All nodes within the same domain shall indicate their domain ID in all communications. Each node of a domain shall keep track of the domain IDs, and shall discard frames received from all domains other than its own, except those assigned for inter-domain communication. The details are for further study.
7. Broadcast transmissions shall be supported in any domain.

5.1.3.2 Unicast communication types

Nodes of a domain can use 3 types of communication: peer-to-peer communications (P2P), centralized mode communications (CM), or unified mode communications (UM).

With P2P communications, direct signal traffic is established between two communicating nodes. Figure 5-1 shows the use of P2P between nodes A and B. Frames addressed to nodes outside the domain are sent to the node associated with the corresponding IDB (node C in Figure 5-1).

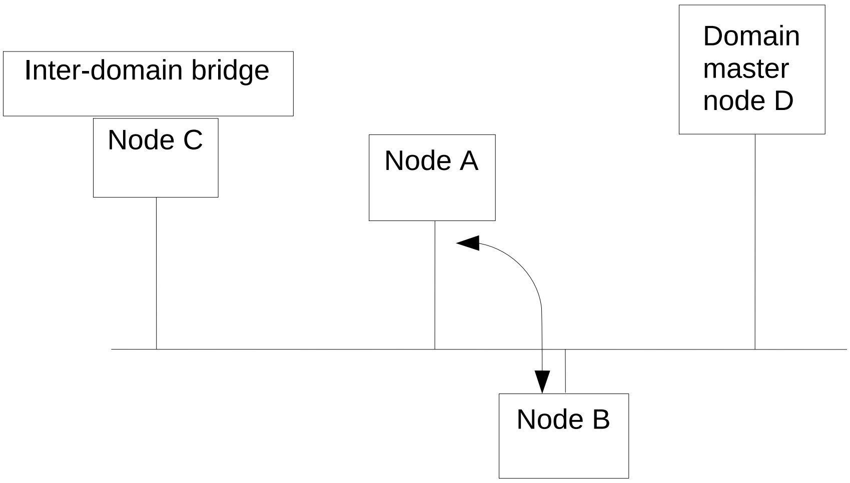


Figure 5-1 – Example of a P2P communication

For CM, only relayed communications are used. Thus, any node of the domain can communicate with another node only through the DAP node. The DAP receives frames from all nodes of the domain and further forwards them to the corresponding addressee nodes. Frames addressed to nodes outside of the domain are forwarded by the DAP to the node associated with the corresponding IDB (node C in Figure 5-2). Usually, but not necessarily, the DAP also serves as a domain master (Figure 5-2). If a particular node has no direct connection to the DAP (is hidden from DAP), this node may use other nodes as relays to reach the DAP (if direct connection between node C and D shown in Figure 5-2 is impossible, node A may relay the frame from C to D).

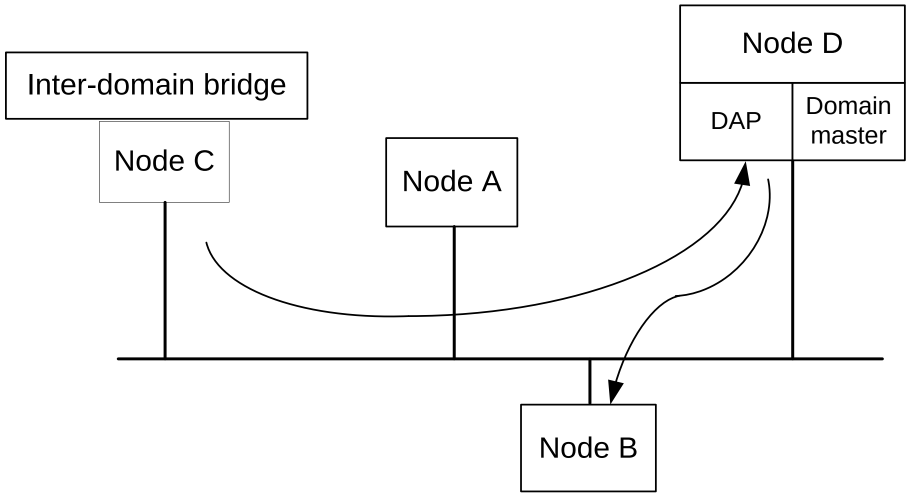


Figure 5-2 − Example of a CM communication

In case of DAP failure, no communication between nodes in the domain is allowed in CM.

For a UM communications, a node can communicate with another node either directly or through one or more relay nodes as shown in Figure 5-3. In the example, two nodes within the same domain (node C and node H) that are hidden from each other, communicate with each other via the relay node (node A). Both nodes are managed by the domain master (node D) and can communicate directly with all other nodes. Frames addressed to nodes outside the domain are sent to the node associated with the IDB (node C in Figure 5-3).

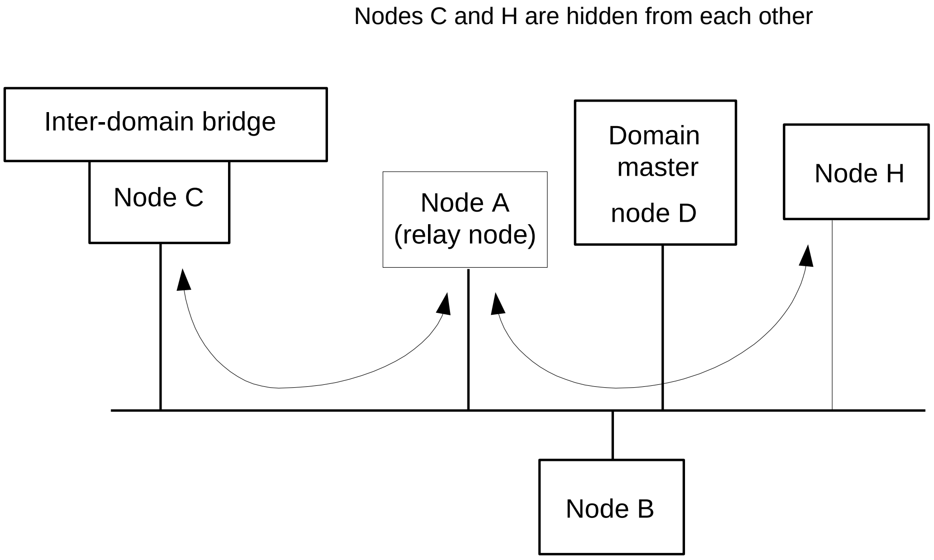


Figure 5-3 − Example of a UM communication

5.1.3.3 Management of the domain

Domain master manages the domain by transmission beacon frames and a set of control/management messages. Beacon frames carry information to be distributed to all nodes of the domain, thus describing for all nodes of the domain the specifics of the domain operation, such as bandplan, domain name, domain ID, the security mode to be used, spectral compatibility requirements (frequency band and maximum transmit power, if applicable), mode of operation, and others (see clause 8.4.2 and clause 8.5.4). Nodes that join the domain shall detect the beacon, and setup the required parameters to comply with the domain operation requirements prior starting the transmission. Beacons also inform all operating nodes on changing the status of the domain or operation rules of the domain.

The domain master also manages domain topology: it may request topology reports from the nodes of the domain, store, process, and distributes relevant topology information back to the nodes. In case L2 routing is set for the domain, the domain master assigns one or more nodes as relays. The domain master shall also assign beacon proxies (BPR). The BPR re-generates the beacon to extend coverage of the network.

Domain master is also responsible to collect statistics on the performance of the domain and provide domain management information to the nodes of the domain.

Main management functions provided by the domain master are summarized in Table 5-1; the detailed description of these functions is in clause 8.5.

Table 5-1 – Domain master management functions

Function	Description
Indication of presence	– Periodically communicate beacon to all nodes in the domain
Determine the domain operation rules	– Inform all nodes on current operation modes and settings in the domain and provide their modification if necessary
Admission control	– Register new nodes into the domain – Facilitate resignation of nodes from the domain – Expel nodes from the domain in case of domain policy violation
Topology management	– Collect and maintain topology information from nodes of the domain – Process and distribute the topology information – Facilitate routing in the domain (assigns relay nodes and BPRs)
Statistics collection	– Collect statistics on performance of the nodes – Store, process, and communicate statistics to the management centers

5.1.4 Quality of service (QoS)

Quality of service (QoS) defines the quality of services delivery to the clients in the network that places requirements on the transmission and queuing of the traffic. All ITU-T G.9956 transceivers shall support priority-based QoS. Support of parameter-based QoS is for further study.

The QoS requirements are supported between nodes inside the same domain and between nodes connected to different domains (if services communicated between nodes that belong to different domains). In the latter case, inter-domain bridges are expected to facilitate provisioning of QoS between nodes connected to different domains and not to compromise the QoS requirements (such as latency). This provisioning is beyond the scope of this Recommendation.

With priority-based QoS, the ITU-T G.9956 transceiver assigns each protocol data unit, incoming from the application entity or management entity, to a certain priority queue, based on the QoS-related parameters indicated by the primitives of this incoming data unit or by QoS-related management primitives. Further, the defined priority-based access method determines the order in which frames from each queue shall be sent to the medium and in which order frames will be processed (and possibly dropped), based on the medium access (MA) priorities assigned to these queues (see clause 8.2.1). The number of supported MA priority queues may be 4, 3, or 2, depending on the profile (see clause 6): standard profile supports 4 queues. Three priority levels associated with the regular incoming application data (at A-interface) are defined, denoted by 0 (lowest level), 1, and 2 (highest level). The priority level 3 is defined as an "over-ruling" priority, for pre-defined emergency data transfers and asynchronous beacons only, and is not intended for regular application data. Management communications use priority level 2 (highest level of application data priority).

5.1.5 Security

ITU-T G.9956 security is designed to address operation over shared media, such as power line. It uses advanced encryption mechanism based on AES-128 [NIST PUB 197] and Counter Mode with Cipher Block Chaining Message Authentication Code algorithm (CCM), according to [NIST SP 800-38C]. The defined node authentication and key management procedure (AKM) provides authentication of each node registering into the domain and can establish point-to-point security, with a unique encryption keys assigned for each pair of communicating nodes. AKM also includes encryption key update and periodical re-authentication of all nodes in the domain.

ITU-T G.9956 security settings are configurable: the domain may be set to a secure mode with different levels of message authentication or to a non-secure mode, without encryption and message authentication. All nodes of the domain are set to the same security level, indicated by the domain master. Security of a network containing more than a single domain is provided by setting all the domains of the network in secure mode. If a domain operates in non-secure mode or has insufficient level of security or message authentication, its communication with secure domains shall be restricted; communications between authenticated nodes of a secure domain with nodes of insecure domains shall not be authorized.

This Recommendation defines security on layer 2 (DLL). If security is provided by layers above L2, fully or partially, a non-secure mode or a mode with reduced security for L2 may be appropriate.

Security procedures are user-friendly, but may require the user to establish a password for each node prior to installation. The rest of the procedures necessary to establish and maintain security do not require involvement of the user. Nodes with no appropriate user interface may use a manufacturer-set password (blind admission).

Confidentiality between clients associated with the same node is supposed to be resolved at the higher layers of the client protocol stack and is beyond the scope of this Recommendation. Details of security are defined in clause 8.6.

5.2 Reference models

5.2.1 Protocol reference model of transceiver

The protocol reference model of a transceiver is shown in Figure 5-4 and described in clause 5.2.1 of [ITU-T G.9955].

The layers above the data link layer (above the A-interface) are beyond the scope of this Recommendation. The shaded part of the reference model is defined in this Recommendation; the non-shaded part is defined in Recommendation [ITU-T G.9955].

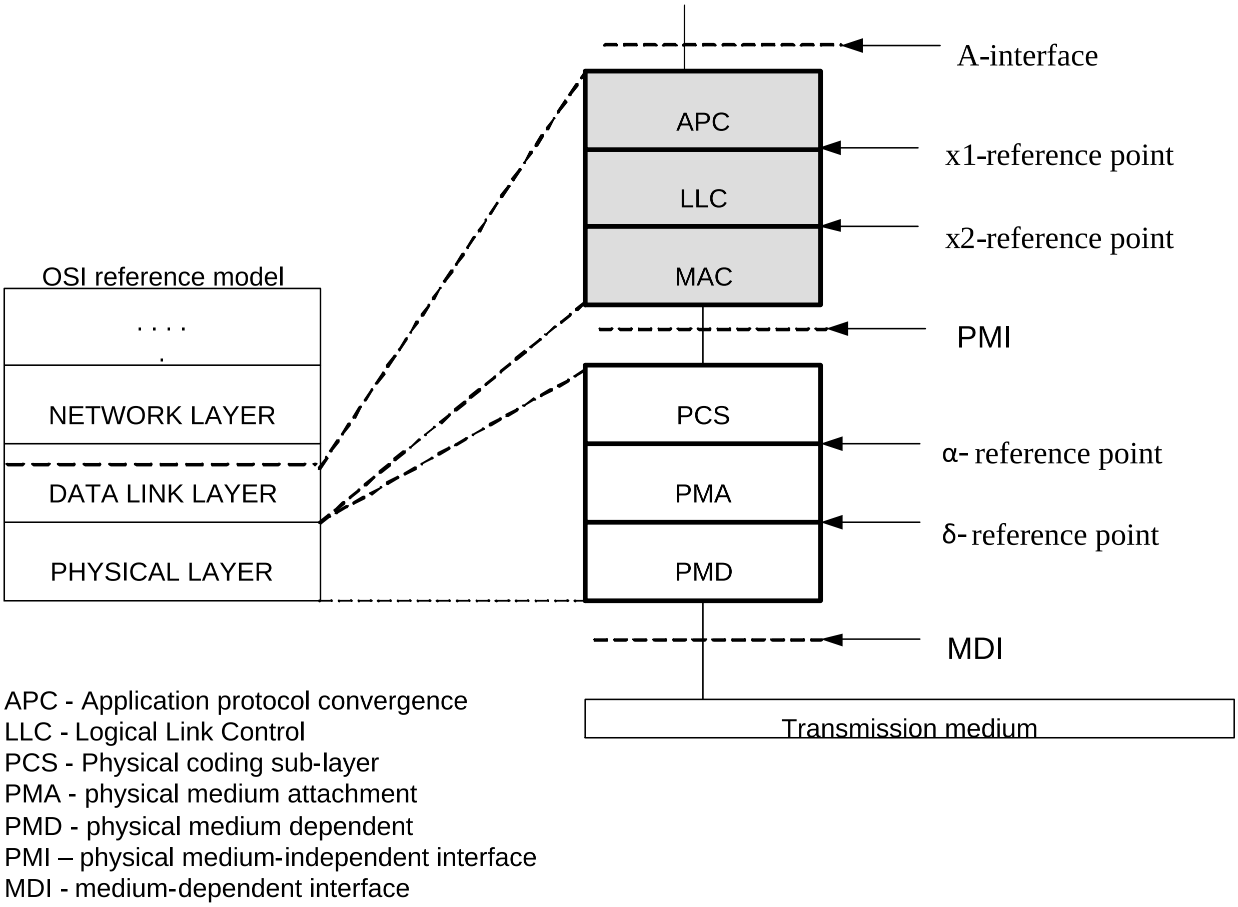

A-interface
APC
x1-reference point
LLC
x2-reference point
OSI reference model
MAC
. . . .
.
PMI
NETWORK LAYER
PCS
α- reference point
DATA LINK LAYER
PMA
δ- reference point
PHYSICAL LAYER
PMD
MDI
Transmission medium
APC - Application protocol convergence
LLC - Logical Link Control
PCS - Physical coding sub-layer
PMA - physical medium attachment
PMD - physical medium dependent
PMI – physical medium-independent interface
MDI - medium-dependent interface

Figure 5-4 – Protocol reference model of ITU-T G.9955/9956 transceiver

5.2.2 Functional description of interfaces

This clause contains the functional description of the ITU-T G.9956 transceiver interfaces (reference points) based on the protocol reference model presented in Figure 5-5. The interfaces shown in Figure 5-5 are defined in this Recommendation.

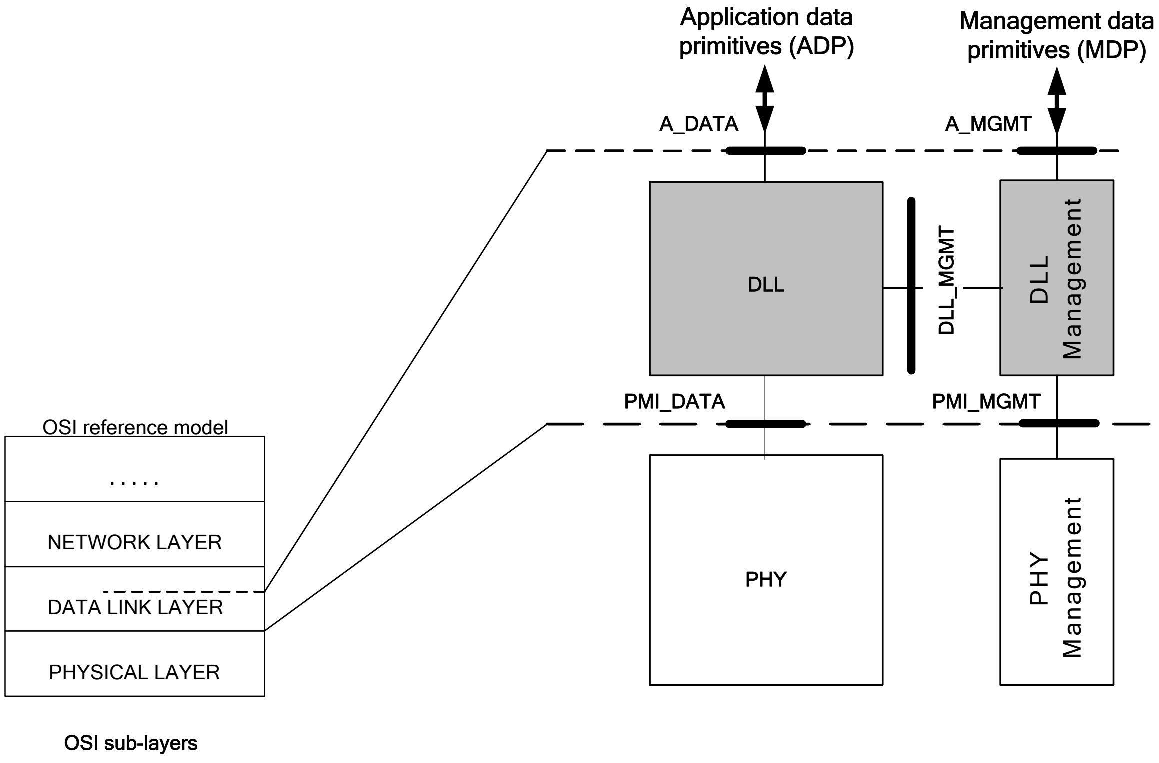


Figure 5-5 − Transceiver reference points related to DLL

The model in Figure 5-5 shows interfaces related to the application data path (A_DATA, PMI_DATA), the management data path (A_MGMT, PMI_MGMT), and management interfaces between data and management plains of the DLL (DLL_MGMT). All interfaces are specified as reference points in terms of primitive flows exchanged between the corresponding entities. The description does not imply any specific implementation of the transceiver interfaces.

5.2.2.1 A-interface

The A-interface is described in terms of primitives exchanged between the AE and DLL. There are six general types of A-interface primitives as shown in Table 5-2. Each primitive type may consist of one or more primitives, related to control or data, respectively. Data primitives (A_DATA) represent the data path of the A-interface, while management primitives (A_MGMT) represent the management and control path. The format of the application data primitives (ADPs) and management data primitives (MDP) is application specific, determined by the AE. The definition of the A-interface management/control and data primitives of each type of the AE are defined in Annex I. The A-interface management primitives that are not AE-specific are defined in clause 8.4.3.4.

Table 5-2 – A-interface primitive type summary

Primitive type	Direction	Description
A-interface data primitives		
A_DATA.REQ	AE → DLL	Data from AE to DLL
A_DATA.CNF	DLL →AE	Data confirmation from DLL to AE
A_DATA.IND	DLL → AE	Data from DLL to AE
A-interface management & control primitives		
A_MGMT.REQ	AE → DLL	Control from AE to DLL
A_MGMT.CNF	DLL → AE	Control confirmation from DLL to AE
A_MGMT.IND	DLL → AE	Control from DLL to AE
NOTE – Primitives presented in this table are exclusively for descriptive purposes and do not imply any specific implementation.		

5.2.2.2 Physical medium-independent interface (PMI)

See clause 5.2.2.1 of [ITU-T G.9955].

5.2.2.3 Peer interfaces between data and management paths

5.2.2.3.1 DLL_MGMT reference point

This reference point defines management and control primitives related to all sub-layers of the DLL (APC, LLC, MAC), as defined in Figure 5-4. These primitives (APC_MGMT, LLC_MGMT, and MAC_MGMT), are shown in DLL functional model, clause 8.1, and defined in clause 8.4.3.

5.2.3 Functional model of a transceiver

The functional model of a transceiver is presented in Figure 5-6. It addresses nodes without extended capabilities as well as nodes with extended capabilities such as domain master, and relaying (including DAP), which differ by their MAC, LLC and upper layer management functionalities. This Recommendation addresses only the shaded part of the functional model; the non-shaded part is addressed in [ITU-T G.9955].

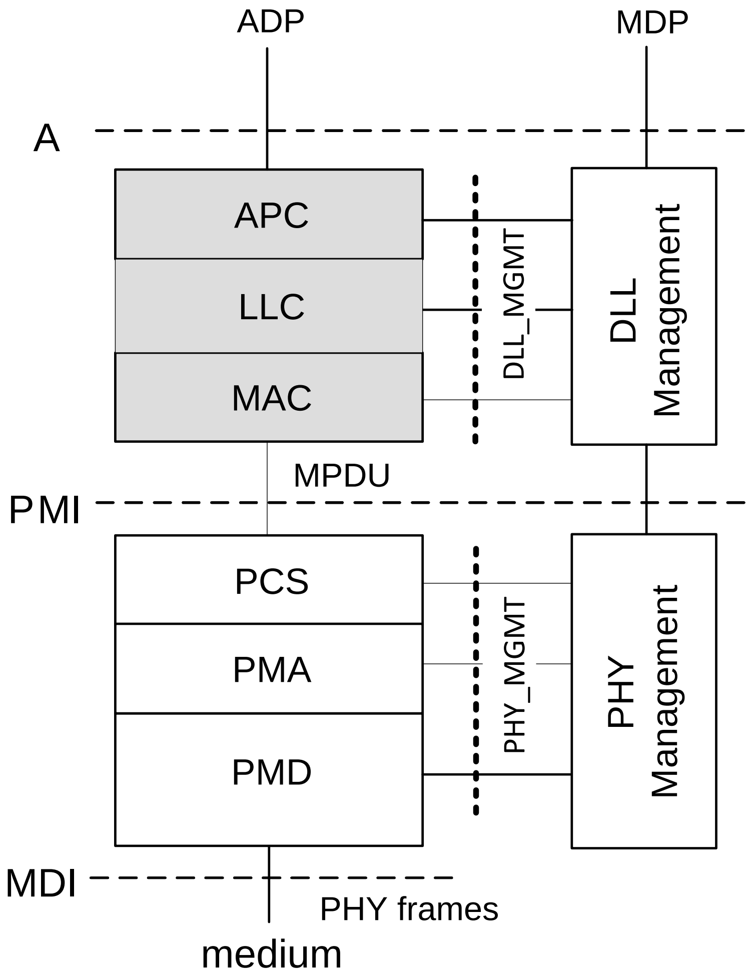


Figure 5-6 – Functional model of ITU-T G.9956 transceiver

The detailed description of the functional model of the DLL is presented in clause 8.1.

6 Convention

6.1 Bit ordering convention

See clause 6.1 in [ITU-T G.9955].

6.2 Message nomenclature convention

The nomenclature for management messages defined in this Recommendation follows the convention follows the conventions described in this clause. The generic nomenclature format of the management message contains the name and the type: *Name.Type*.

Four message types are defined: Request (*req*), Confirm (*cnf*), Indication (*ind*) and Response (*rsp*). The *.req* and *.ind* messages are intended to initialize the message exchange. The *.cnf* message shall be sent only in response to the received *.req* message. The *.rsp* message shall be sent only in response to the received *.req* message. An *.ind* message may be sent in response to *.req* message. A generic description of the message sequence chart is presented in Figure 6-1.

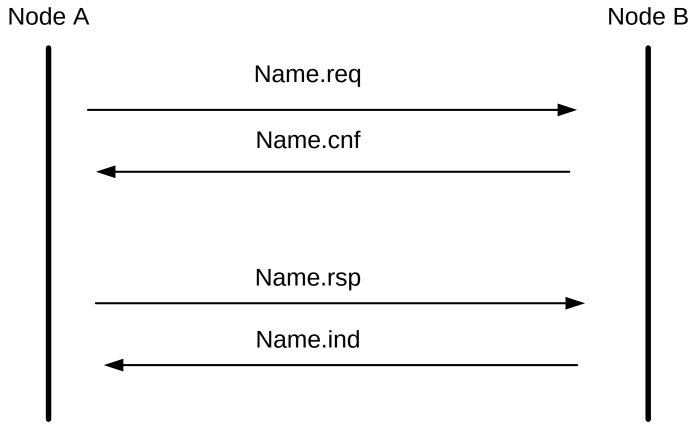


Figure 6-1 – Generic message sequence chart

The *Name* has the following structure: [*Category*][*Source*][*Function*] with the following meaning:

Category: The message application category, such as Admission, Beacon, Chanel Estimation;

Source: The source of the message, such as end-node or domain-master;

Function: Generic function of the message, such as parameter request, parameter report, specific command, and similar.

Source and *Function* parts are optional, although at least one of them shall be present. This Recommendation uses management messages presented in Table 8-19.

7 Profiles

Profiles are intended to specify nodes with significantly different levels of complexity and functionality. A more complex profile is a superset of less complex profile and shall interoperate with that profile. Less complex profiles target specific deployment scenarios or regional variations and are identified by a reduced list of supported features and parameter values comparing to a Standard profile, which is a superset of all less complex profiles.

A node shall be classified into a particular profile according to its degree of complexity and functionality. For compliance with ITU-T G.9955/G.9956, a node is required to support at least one profile. Profiles are summarized in Table 7-1.

Table 7-1 – Profiles

Profiles	Description	Note
Standard Profile	Implements all requirements of the Recommendation	
Low Complexity Profile	Reduced functionality, as per Table 7-2	

7.1 Low complexity profile (LCP)

Table 7-2 describes the valid values of parameters for the LCP that differentiate it from Standard profile.

Table 7-2 – Reduced features and valid values of parameters

Parameter/feature	Description	Notes
DLL-related		
Domain master functionality	Not required	
DAP functionality	Not required	
Data relaying	Not required	
Beacon relaying	Not required	
Selective acknowledgement	Not required	
Multi segment MPDU transmission	Not required	
Supported MA priorities	0, 2	
PHY-related		
Modulation	2 bits per subcarrier	
Code rate	½	
EVM	12 dB	Corresponds to QPSK
Maximum FEC block size	128 bytes	

8 Data link layer (DLL) specification

8.1 Functional model of DLL and frame formats

The functional model of the DLL is presented in Figure 8-1. The A-interface and PMI are, respectively, two demarcation reference points between AE and DLL and between DLL and PHY. Internal reference points x1 and x2, respectively, show logical separation between the APC and LLC and between the LLC and MAC.

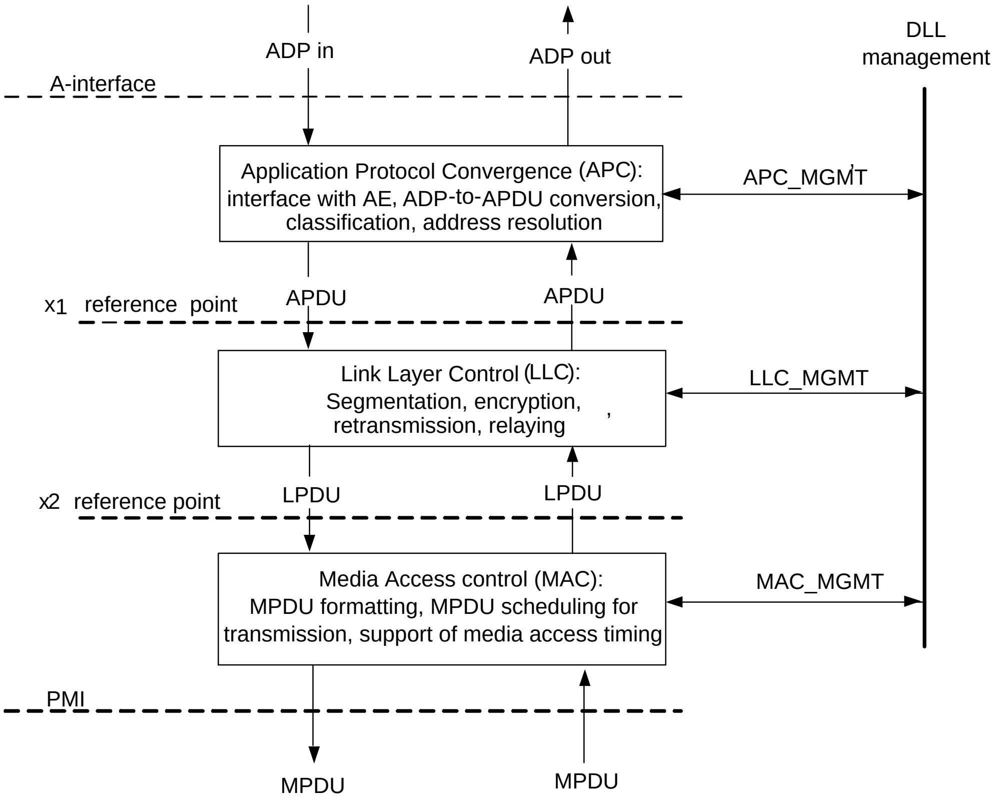

ADP in
ADP out
DLL management
A-interface
Application Protocol Convergence (APC): interface with AE, ADP-to-APDU conversion, classification, address resolution
APC_MGMT
x1 reference point
APDU
APDU
Link Layer Control (LLC): Segmentation, encryption, retransmission, relaying
LLC_MGMT
x2 reference point
LPDU
LPDU
Media Access control (MAC): MPDU formatting, MPDU scheduling for transmission, support of media access timing
MAC_MGMT
PMI
MPDU
MPDU

Figure 8-1 – Functional model of DLL

In the transmit direction, application data primitives (ADP) enter the DLL from the AE via the A-interface. Every incoming ADP set meets the format defined by the particular application protocol; for an AE of IPv6 a particular type the ADP set is defined in Annex I, clause I.1 (IPv6 APC). Each incoming ADP set is converted by the APC into an Application Protocol Data Units (APDUs), which includes all parts of ADP intended for communication to the destination node(s). The APC is responsible to forward APDUs to peers APCs.

The APC also identifies ADP classification primitives (e.g., class of service, priority tags, etc.) to support QoS requirements assigned for the service delivered by the ADP and shall maintain one or more priority queues associated with APDUs it transmits; it assigns each APDU to the corresponding queue. The number of queues may depend on the application protocol and the profile of the device.

NOTE – The bridging function between the clients associated with the ITU-T G.9955/9956 node (if more than one) is considered to be a part of the AE and is beyond the scope of this document.

The APC transfers APDUs to the LLC via the x1 reference point, which is application independent. The LLC also receives from the DLL management entity sets of management data primitives for LLC control frames, which are mapped into Link Control Data Units (LCDU). The LLC is responsible to establish exchange of LCDUs (management frames) between peer LLCs.

In the LLC, the incoming APDU and LCDU are mapped into LLC frames and may be encrypted using assigned encryption keys (see clause 8.6). Long LLC frames are segmented as described in clause 8.1.2.2. Segments are transformed into LLC Protocol Data Units (LPDUs) and then passed to the MAC via the x2 reference point. The LLC is also responsible for relay operation (if enabled) and for retransmission of improperly received segments (if required).

The MAC is responsible to concatenate LPDUs into MAC Protocol Data Units (MPDUs) and then convey these MPDUs to the PHY via the PMI, in the order determined by their MA priorities, and according to the medium access rules established in the domain (e.g., see clause 8.2).

In the receive direction, MPDUs from the PHY enter via the PMI. The MAC disassembles the received MPDUs into LPDUs, which are passed over the x2 reference point to the LLC. The LLC recovers the original APDUs and LCDUs from the received LPDUs, decrypts them if required, and conveys them to the APC and the LLC management entity, respectively. In the APC, the ADPs are recovered from the received APDUs and conveyed to the AE.

The LLC is responsible for detection of erroneous LPDUs and generation of acknowledgement. It discards the erroneous LPDUs and, if the source node requested retransmission of erroneous LPDUs, LLC generates an ACK response to trigger retransmission.

NOTE – No assumptions should be made on partitioning of APC, LLC, and MAC in particular implementations; x1 and x2 are reference points and serve exclusively for convenience of system definition.

8.1.1 Application protocol convergence sub-layer (APC)

The functional model of the APC is presented in Figure 8-2. It is intended to describe in more detail the APC functional block presented in Figure 8-1.

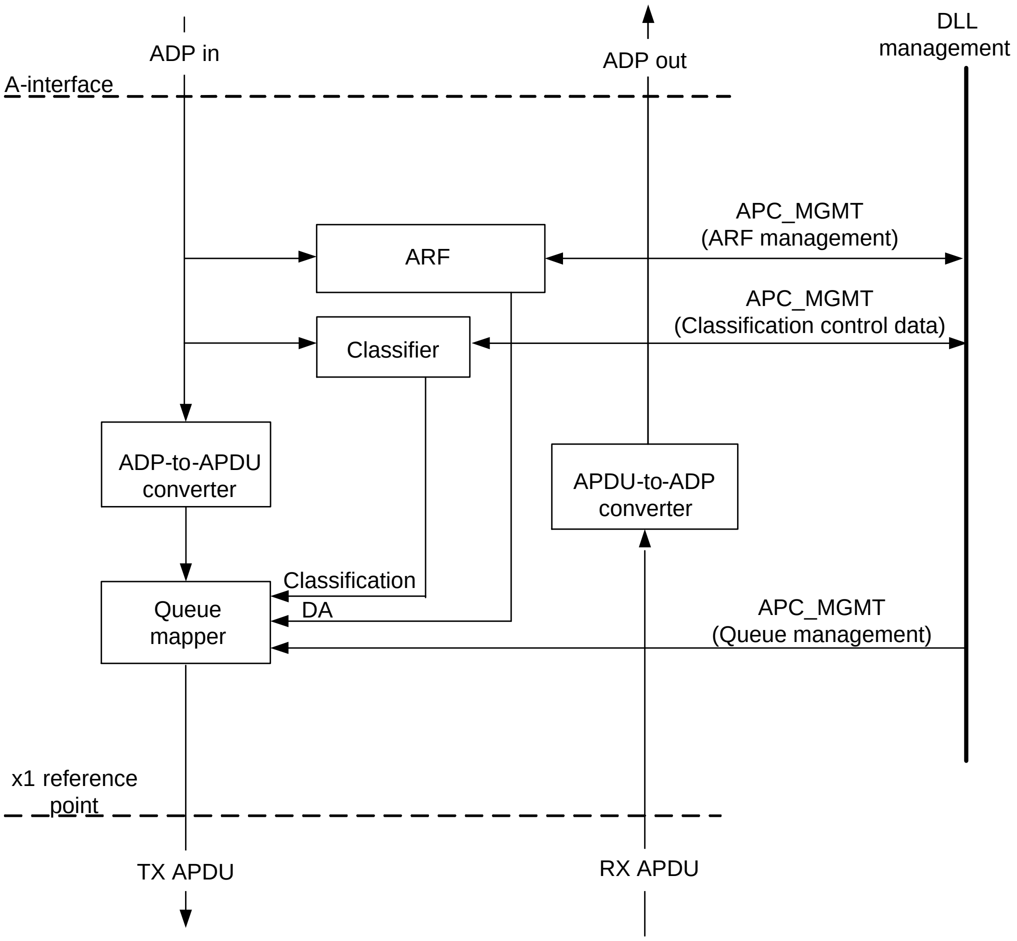


Figure 8-2 – Functional model of APC

In the transmit direction, the incoming ADP data unit is converted into an APDU as defined in Annex I. The queue mapper maps APDUs into queues, depending on their destination address (DA), class of service (priority), and control parameters provided by DLL management entity, as described in Annex I. After mapping, each APDU, tagged with its priority, is sent to LLC via the x1 reference point. The size of the ADP (in both transmit and receive directions) can be up to 1522 bytes.

In the receive direction, the APDUs incoming via the x1 reference point are converted back into the ADP set of the corresponding application protocol.

The classification information embedded in the ADP is extracted from the incoming data units and may be used to map the APDU into an appropriate priority queue. Relevant classification parameters depend on the application protocol and are defined in Annex I.

The Address Resolution Function (ARF) associates the destination address of the incoming ADP with the physical address of the node this ADP has to be sent. The ARF may store addresses of the clients associated with the node collected from the incoming ADP, and addresses of the clients associated with other nodes in the network (advertised by these nodes). The address resolution procedure performed by ARF depends on the application protocol and is defined in Annex I.

The APC sublayer allows serving of one or more application protocols. The functional model in Figure 8-2 shows an APC serving a particular AE protocol. In case more than one application protocol is needed, multiple APC shall be used, as shown in Figure 8-3.

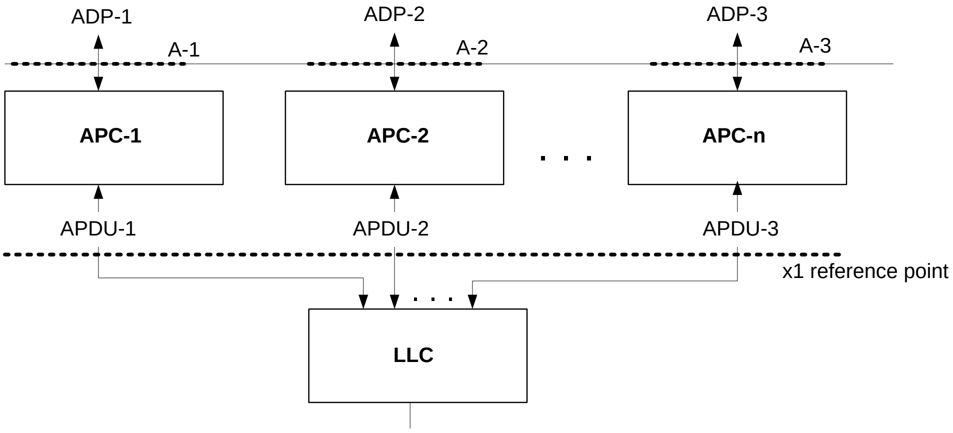


Figure 8-3 – Block diagram of APC sublayer serving multiple application protocols

Each APC in Figure 8-3, same as in Figure 8-2, serves a particular AE protocol. All ADPUs from all APCs are passed to the LLC via the x1 reference point, each with a tag indicating its application protocol. The tag is further communicated to the receiving node in the LLCFT field of the LFH (see clause 8.1.2.1.2); based on this tag, the received APDU is passed to the corresponding APC. The format of each APDU is per its application protocol, and shall be as defined in Annex I.

Simultaneous support of multiple application protocols is optional. The maximum number of APCs (not more than 8) and arbitration of the order in which APDUs of different APCs are passed to LLC (internal priority of APC) are for further study.

8.1.2 Logical link control sub-layer (LLC)

The functional model of the LLC is presented in Figure 8-4. It is intended to describe in more detail the LLC functional block presented in Figure 8-1.

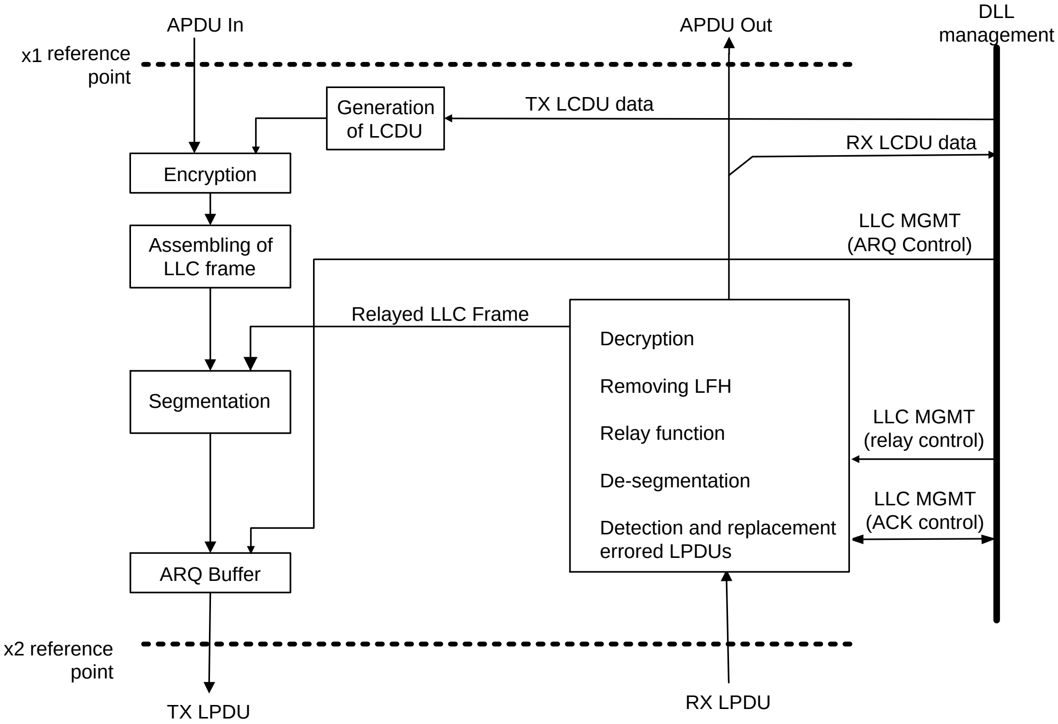


Figure 8-4 – Functional model of LLC

In the transmit direction, an LLC frame is formed from each APDU entering via the x1 reference point and from each LCDU entering via the LLC_MGMT reference point. The APDU has a format depending on the application protocol, as defined in Annex I. The LCDU carry management data and has a format defined in 8.1.2.1.3. The LLC frame may be encrypted using encryption rules defined in clause 8.6.1.

Further, LLC frame is divided into segments of equal size, as defined in clause 8.1.2.2; the size of the segment is controlled by the LLC. Each segment is pre-pended by a header and appended with a CRC, thus forming an LPDU. LPDUs are passed to the MAC via the x2 reference point and also stored in the ARQ buffer, if the transmitted LLC frame has to be acknowledged.

LPDUs that need to be retransmitted are extracted from the ARQ buffer and passed to the MAC to be assembled into the outgoing MPDU. To assist retransmission, the receive part of the LLC generates ACKs, which are passed to the MAC to be transmitted using Imm-ACK frames, as defined in clause 8.3.3.1.

In the receive direction, the LPDUs disassembled from the incoming MPDU in the MAC enter the LLC via the x2 reference point. The LLC verifies all received LPDUs, generates an acknowledgement, if so instructed by the source of the received frame, and discards erroneous LPDUs. Further, after all the LPDUs associated with the sent LLC frame are cleared, the LLC recovers the LLC frames from the received LPDUs. The recovered LLC frames are decrypted (if encryption is enabled) and their payloads are passed to APC via x1-reference point, if they carried an APDU or to the DLL management via the LLC_MGMT reference point, if they carried an LCDU.

The relay function, if enabled, extracts LLC frames that are subject to relaying using information in the LFH (see clause 8.1.2.1.2) and passes them back to the MAC for transmission to the next destination. The relayed LLC frames are processed as regular LLC frames, see Figure 8-4. The DLL management entity controls priority settings for the relayed LLC frames as defined in clause 8.1.2.4. Relayed LLC frames shall not be decrypted.

8.1.2.1 Assembling of LLC frame

The LLC frame is formed from either an APDU or an LCDU, with a format as described in Figure 8-5.

8.1.2.1.1 LLC frame format

If encryption or other security measures (see Table 8-42) is required, the incoming APDU or LCDU shall be encrypted using CCMP, as described in clause 8.6.1. A CCMP header and a message integrity check (MIC) shall be added as described in Figure 8-5 (case – CCM encrypted). The format and content of the CCMP header and the MIC shall be as specified in clause 8.6.1.2.3.

If no security measures are required, the CCMP header and the MIC shall not be added, as described in Figure 8-5 (Case – CCM unencrypted). The presence of a CCMP header and a MIC shall be indicated in the LLC frame header (LFH), as defined in clause 8.1.2.1.2.

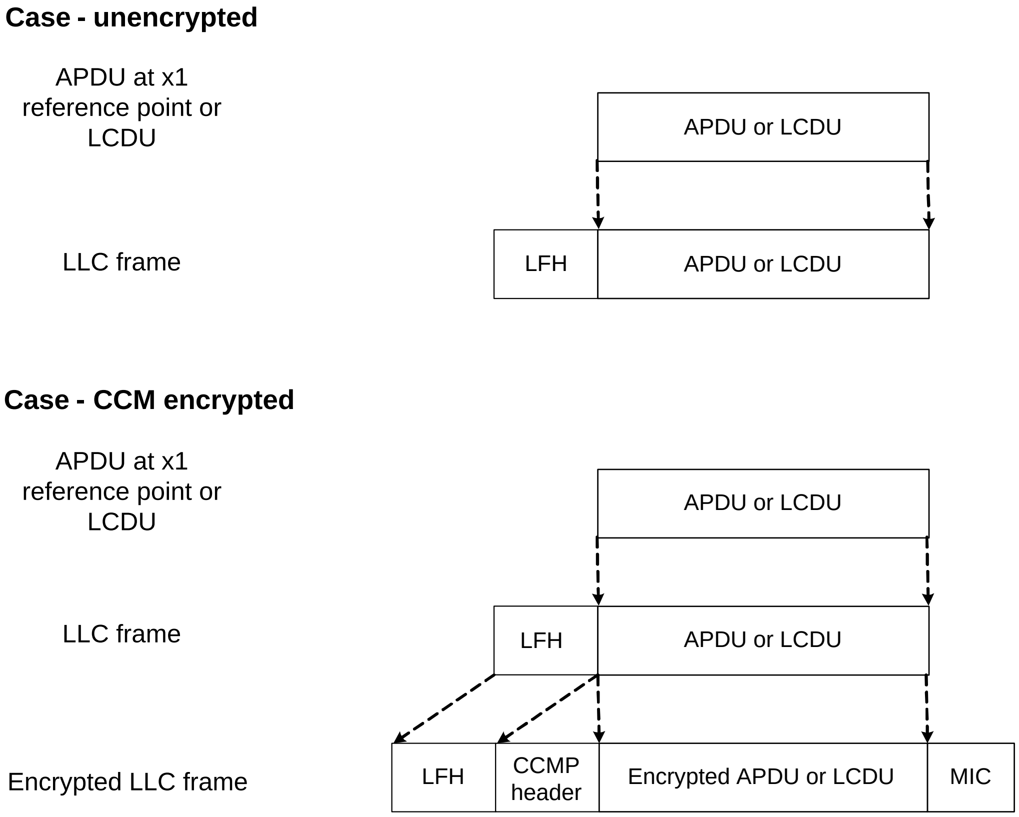


Figure 8-5 – LLC frame format (unencrypted and encrypted)

8.1.2.1.2 LLC frame header (LFH)

The LLC frame header (LFH) shall be composed of the fields described in Table 8-1. The LFH is variable in length. Octet 0 shall be passed to the MAC first.

Table 8-1 – LFH fields format

Field	Octet	Bits	Definition
LLCFT	0	[3:0]	LLC frame type
CCMPI	0	[4]	CCMP header presence indication
MHI	0	[5]	Mesh header presence indication
Reserved by ITU-T	0	[7:6]	Reserved by ITU-T
MESH	Variable	[7:0]	Mesh header (see Table 8-4)
NOTE – All bits reserved by ITU-T shall be set to the default value of zero by the transmitter and ignored by the receiver.			

8.1.2.1.2.1 LLC frame type (LLCFT)

The LLCFT field is a 4-bit field that indicates the type of the APDU that makes up the LLC frame. Table 8-2 lists the valid LLC frame types:

Table 8-2 – LLC frame types

LLC frame type	Value
Reserved	0
Management frame (LCDU)	1
Data frame (APDU) – See Table 8-3	2-15

Table 8-3 – Data frame (APDU) types

APDU type	Value	Description
IPv6 APDU (non-compressed)	2	IPv6 [IETF RFC 2460]
Compressed IPv6 APDU, unicast	3	IPv6, default IPv6 header compression as per Annex J
Compressed IPv6 APDU, multicast	4	
Compressed IPv6 APDU (6LoWPAN)	5	IPv6, compressed as per clause 10 of [IETF RFC 4944], Annex J
	6	IPv6, compressed as per [IETF RFC 6282], Annex J
IPv4 APDU (non-compressed)	9	IPv4 [IETF RFC 791]
Ethernet APDU	12	Ethernet
Vendor APDU	15	Vendor discretionary L3 protocol and APC (Note 2)
NOTE 1 – Other values are reserved by ITU-T. NOTE 2 – In case of Vendor APDU, interoperability can be established for proprietary network solutions only.		

8.1.2.1.2.2 CCMP header presence indicator (CCMPI)

CCMPI is a 1-bit field that is used to indicate whether security is applied to the LLC frame or not. If set to one, the LLC frame shall be secured using one of the settings defined in Table 8-42 and the CCMP header shall follow the LFH and MIC shall be added. If set to zero, the LLC frame shall not be secured and shall not include CCPM header and MIC.

8.1.2.1.2.3 Mesh header presence indicator (MHI)

The MHI is a 1-bit field that is used to indicate whether the mesh header field is present or not. The MHI shall be set to 0 if the mesh header is not present and set to 1 if it is present.

8.1.2.1.2.4 Mesh header (MESH)

The MESH field supports L2-relaying operation (see clause 8.3.2.1.2) and extended addressing options.

The MESH field shall be used when at least one the following conditions apply:

1. The extended addressing mode (see clause 8.3.1) is required to identify the source node or the destination node or both.
2. The LLC frame is transmitted using L2-relaying.

In case neither of conditions above applies this field shall not be sent (MHI set to 0) and shall be ignored by the receiver.

The MESH field shall be formatted as defined in Table 8-4; the length of the field depends of whether short (16-bit) or extended (64-bit) addressing mode is used and whether source routing (SR) extension field is added to the mesh header. The used addressing mode is indicated in the ADM field. The content of the MESH field (except for the HopsLft field) shall not be changed when an LLC frame is relayed by another node.

Table 8-4 – Description of the MESH field

Field	Octet	Bits	Description
RELI	0	[0]	Indicates whether L2-relaying extension headers are present

ADM	0	[3:1]	Bit [2] indicates the SA format (V-bit);
			Bit [3] indicates the DA format (F-bit);

LPRI	0	[5:4]	LLC Frame priority
SRI	0	[6]	Source routing extension presence indication
Reserved by ITU-T	0	[7]	Reserved by ITU-T
SA	3-4 or 3-10	[15:0] or [63:0]	Originator address (source)
DA	5-6 or 5-12 or 11-12 or 11-18	[15:0] or [63:0]	Final address (destination)
HopsLft	1	[7:0]	Maximum number of hops allowed
SN	2	[7:0]	Sequence number (flooding control)
SR	Variable	[15:0]	Source routing header
NOTE – All bits reserved by ITU-T shall be set to the default value of zero by the transmitter and ignored by the receiver.			

8.1.2.1.2.4.1 Relaying presence indicator (RELI)

This filed shall be set to 1 to indicate the presence of additional L2-relaying headers, and shall be set to 0 otherwise.

If the RELI field is set to 1, the HopsLft and SN fields shall follow the DA field, and the LPRI and SRI fields shall be transmitted and shall be processed by the receiver. If, in addition, the SRI field is also set to 1, the SR header shall follow the SN field.

If the RELI field is set to 0, the LPRI and SRI shall be set to 0 by the transmitter and ignored by the receiver.

8.1.2.1.2.4.2 Addressing mode (ADM)

This field defines the addressing mode of the SA and DA as follows:

- SA: a short 16-bit format if bit [2] is set to 1 and EUI-64 format if bit [2] is set to 0;
- DA: a short 16-bit format if bit [3] is set to 1 and EUI-64 format if bit [3] is set to 0.

For intra-domain communications, both for APDU and LCDU, bit [2] and bit [3] shall be both set to 1. When the 16-bit format for source or destination node is not available, bit [2] or bit [3] shall be set to 0, respectively. For inter-domain communications, if allowed, bits [2] and [3] shall be both set to 0.

8.1.2.1.2.4.3 LLC frame priority (LPRI)

The LPRI field is a 2-bit field with valid values from 0 to 3 used to indicate the priority of L2-relayed LLC frames.

For LLC frames that are not L2-relayed, this field shall be set to 0 by the transmitter and ignored by the receiver.

For L2-relayed LLC frames carrying APDUs, this field shall be set to the priority assigned by the classifier (see clause 8.2.1) of the node that originated the LLC frame; otherwise, it shall be set to 0. For L2-relayed LLC frames carrying LCDUs, this field shall be set to 2.

8.1.2.1.2.4.4 Source routing header presence indicator (SRI)

This bit shall be set to 1 to indicate the presence of the SR field, and shall be set to 0 otherwise. In case the RELI field is set to 0, this field shall be set to 0 by the transmitter and ignored by the receiver.

8.1.2.1.2.4.5 Originator address (SA)

Address of the originating node expressed in a 16-bit format if bit [2] of ADM field is set to 1 or in EUI-64 format if bit [2] of ADM field is set to 0.

8.1.2.1.2.4.6 Final destination address (DA)

Address of the final destination node expressed in a 16-bit format if bit [3] of ADM field is set to 1 or in EUI-64 format if bit [3] of ADM field is set to 0.

8.1.2.1.2.4.7 Number of hops left (HopsLft)

The HopsLft is an 8-bit field that shall indicate the number of times the LLC frame is allowed to be relayed expressed as an unsigned integer. If a node receives an LLC frame to be relayed with HopsLft field not equal to zero, it shall relay the LLC frame and decrement HopsLft by one in the relayed LLC frame. If a node receives an LLC frame with HopsLft field equal to zero, such LLC frame shall not be relayed.

The initial value of the HopsLft field shall be set by the node originating the frame. For the case the source routing extension is not present (SRI = 0), the initial value of the HopsLft field shall be equal or higher than the number of times that the LLC frame is expected to be relayed before reaching its destination.

In case the source routing extension is present (SRI=1), the initial value of the HopsLft field shall be set to the exact number of relays on the path from the source node to the destination node which equals the number of entries in the SR field (see clause 8.1.2.1.2.4.9).

8.1.2.1.2.4.8 Sequence number (SN)

The SN field is to assist L2 broadcast relaying of the frame inside the domain using the procedure defined in clause 8.3.2.2.2.

The field shall be formatted as an 8-bit unsigned integer that defines the sequential number of the LLC frame in the range from 0 to 255. The value shall wrap around, i.e., the SN=0 shall be set for the LLC frame which is next to one with SN = 255.

8.1.2.1.2.4.9 Source routing extension header (SR)

The SR field supports L2-relaying by using source routing (see clause 8.3.2.2.1); this field shall only be present if the SRI field is set to 1.

The SR field shall be formatted as defined in Table 8-5; the length of the field is variable.

Table 8-5 – Description of the SR field

Field	Octet	Bits	Description
RNAL_n	0-1	[15:0]	Short address of the last relay node (relay *n*, closest to the final destination node)
RNAL_n-1	2-3	[15:0]	Short address of the relay node preceding the last relay node (relay *n*-1)
. . .	. . .	. . .	. . .
RNAL_1	2(n-1) – (2(n-1)+1)	[15:0]	Short address of the first relay node (relay 1, closest to the source node)

The SR field shall consist of a list of short addresses of relay nodes participating in the path from the source node to the destination node, as described in Table 8-5.

The last relay node of the path (closest to the destination) shall be set as the first entry in the list and the first relay node of the path (closest to the source) shall be set as the last entry in the list.

For any given node, the next relay node address shall be indicated in the entry of the list that precedes the entry containing the address of the given node; the number of this entry equals to the value of HopsLft set by the given node.

8.1.2.1.3 LCDU frame format

The LCDU frame format, including size of the fields, shall be as presented in Figure 8-6.

	LSB MSB
≤ 1000 octets	LCDU Payload
4 octets	FCS

Figure 8-6 – LCDU format

The LCDU payload shall be used for carrying management messages only, with the format defined in clause 8.5.1.

The LLC frame carrying an LCDU shall have a format as defined in clause 8.1.2.1.1 and LLCFT = 1 in LPH. The DA and SA of the LCDU are 16/64-bit originating node ID and 16/64-bit final destination node ID indicated in the MESH field of the LFH, if MESH field is present, or 16-bit SNID and DNID indicated in the MPH, if MESH field is not present. The LCDUs communicated between nodes of the same domain shall use 16-bit Node ID both for the originating and the final addresses.

For FCS computation, the DA field followed by the SA field shall be pre-pended to the LCDU such that LSB of the DA is transmitted first. The FCS shall be then computed over all fields, from the first bit of the DA field to the last bit of the LCDU payload using the standard IEEE 802.3 Ethernet 32-bit FCS computation algorithm [7]. The FCS field shall not be included when MIC is used in the LLC frame encapsulating the LCDU.

NOTE – The MIC protects both the LPDU payload and its source/destination address that is a part of Associated Data of the encrypted message, see clause 8.6.1.1.3.

An LCDU shall be transmitted starting from the first octet of the LCDU payload, LSB first.

Relaying of LLC frames carrying LPDUs shall follow the same rules as relaying of LLC frames carrying APDUs, as described in clause 8.1.2.3 and clause 8.3.2.2. If L3 relaying is set in the domain (L2 relaying is disabled), LCDUs that may require L2 relaying shall not be used and their functions, if required, shall be provided by corresponding L3 management messages or in-band management messages.

8.1.2.1.4 APDU frame format

APDU frame format depends on the used application protocol and may be different for different types of APC. APDU formats per APC are presented in Annex I.

8.1.2.2 Segmentation

8.1.2.2.1 Generation of LPDUs

The process of generating LPDUs from LLC frames is presented in Figure 8-7.

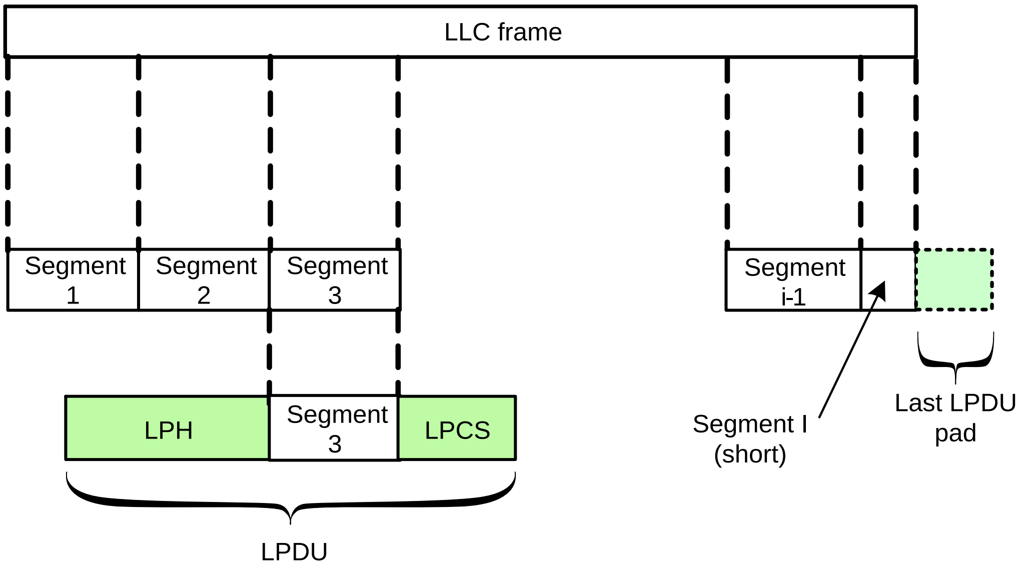


Figure 8-7 – Generation of LPDUs from LLC frames

An LLC frame shall be formed by pre-pending an LFH to an APDU or to an LCDU, and can either be unencrypted or encrypted, as described in clause 8.1.2.1. If the size of the LLC frame exceeds the assigned size of a single LLC segment, which is S_LLC bytes, the LLC frame shall be segmented as described in Figure 8-7:

- The first bit of the first segment shall be aligned with the first bit of the LLC frame.
- All the segments of the same LLC frame shall be of the same length, except the last one. which may be shorter.
- The length of the segment shall be equal to or less than S_LLC bytes.
- The number of segments for an LLC frame shall not exceed 64.

The value of S_LLC shall be as defined in clause 8.4.

An LPDU is formed by pre-pending an LPDU header (LPH) to the segment and by appending an LPDU check sequence (LPCS), as shown in Figure 8-7. The LPH contains the information necessary to re-assemble the LLC frame from the received segments and the LPCS allows detection of corrupted LPDUs. The LPCS shall be calculated as described in clause 8.1.2.2.3. The last LPDU, if shorter than others, shall be padded accordingly. The content of the padding is vendor discretionary; the size of the pad is indicated in MPH and shall not exceed 63 bytes. The format of the LPH is defined in clause 8.1.2.2.2. If the LLC frame is shorter than S_LLC bytes, this frame forms one segment and one LPDU, respectively, which is also considered as the "last LPDU" carrying the segments of the LLC frame.

NOTES:

The size of the segment (where $N_{SEG} \leq$ S_LLC) for the given LLC frame of N_{LLC} bytes long can be computed as:

1. Minimum overhead:

   $N_{SEG} = ceiling(N_{LLC}/NOS)$, where *NOS* is the desired number of segments. The last segment may be incomplete, and the corresponding LPDU will be padded by $P = N_{SEG}\times NOS - N_{LLC}$.

2. For multi-segment MPDU transmission, it is often beneficial if the size of the LPDU is same or close to the size of the FEC codeword to be assigned by the PHY. This can be achieved by setting the size of the LPDU as:

   $N_{LPDU} = N_{LLC}/[ceiling(N_{LLC}/(K-N_{LPH}-N_{LPCS}))]$, where K is the value of the maximum FEC codeword size, as indicated in the PFH (valid values of K are 128 and 239), N_{LPH} is the size of the LPH field, and N_{LPCS} is the size of the LPCS field.

3. No MPDU padding for MPDUs carrying a single LPDU:

   a. Compute $N_1= ceiling(N_{LLC}/\text{NOS})$;

   b. Compute the smallest value of *a* that brings $N_1 + N_{LPH} + N_{LPCS} + N_{MPH} + a$ to be a valid MPDU size (where N_{LPCS} equals the LPCS field size for MS-MPDU, and equals 0 for SS-MPDU, as defined in clause 8.1.3.1);

   c. Compute $N_{SEG} = N_1 + a$.

   d. The pad of the last LPDU, *P*, can be computed as for Case 1 above.

The LPDUs of the same LLC frame shall be passed to the MAC via x2-reference point in the numerical order of the segments they carry, starting from the LPDU carrying the first segment.

8.1.2.2.2 LPDU header (LPH) format

Table 8-6 shows the format of the LPH. Octet 0 shall be passed to the MAC first.

Table 8-6 – LPDU header format

Field	Octet	# of Bits	Definition
SSN	0	6	Segment sequence number
Reserved	0	2	Reserved by ITU-T (Note)
NOTE – Bits reserved by ITU-T shall be set to 0 by the transmitter, and shall be ignored by the receiver.			

8.1.2.2.2.1 Segment sequence number (SSN)

This 6-bit field identifies the relative order of the segment within the stream of segments corresponding to the transmitted LLC frame. The value shall be formatted as unsigned integer in the range from 0 to 63.

The SSN shall be initialized to 0 for the first segment of the LLC frame and shall be incremented by 1 for each subsequent segment that is associated with this LLC frame.

8.1.2.2.3 LPDU check sequence (LPCS)

The LPCS is a 32-bit cyclic redundancy check (CRC) and shall be computed over all the fields of the LPDU in the order they are transmitted, starting with the LSB of the SSN field of the LPDU header (clause 8.1.2.2.2.1) and ending with the MSB of the last octet of the LPDU segment.

The LPCS shall be computed using the following generator polynomial of degree 32:

$$G(x) = x^{32} + x^{28} + x^{27} + x^{26} + x^{25} + x^{23} + x^{22} + x^{20} + x^{19} + x^{18} + x^{14} + x^{13} + x^{11} + x^{10} + x^{9} + x^{8} + x^{6} + 1.$$

The LPCS shall be constructed as follows:

1. The n bits of the LPDU subject to LPCS are considered to be the coefficients of a polynomial of degree n-1, such that the LSB of the first octet of the LPDU header is the coefficient of the x^{n-1} term, and the MSB of the last octet of the LPDU segment is the coefficient of the x^0 term. This polynomial is referred to as $N(x)$.
2. Replace the 16 highest-order coefficients of $N(x)$ with their ones'-complement values.
3. Multiply the result of step 2 by x^{32}. This result is referred to as $N_1(x)$.
4. Compute the LPCS as the ones-complement of the remainder of $N_1(x)$ divided by $G(x)$.

The bits of the LPCS shall be transmitted in sequential order, starting from the coefficient of the highest order term (x^{31}), referred as the MSB, and continuing to the x^0 term.

8.1.2.3 Retransmission of LPDUs

LPDUs assigned for retransmission (see clause 8.1.2) shall be either assembled into an outgoing MPDU carrying other LPDUs associated with the same LLC frame (to be transmitted for the first time or retransmitted) or sent in a separate MPDU. This separate MPDU shall be assigned MA priority as defined in clause 8.2.1.

LPDUs shall be retransmitted with no changes in their segments and in their LPH.

8.1.2.4 Transmission of LCC frames

The LLC frames shall be assembled and further processed in the order APDUs or LCDUs carried by these LLC frames arrive, except the APDUs of priority 3. If an APDU and an LCDU arrive simultaneously, one that has higher priority shall be assembled and processed first and, in case of same priority, LCDU shall be assembled and processed first.

In case an incoming APDU is of priority 3, even if arrives during the processing of another LLC frame of lower priority, it shall be processed first, i.e., the LPDUs sourced from an LLC frame carrying APDU of priority 3 shall be submitted to x2-reference point with precedence over the remaining LPDUs comprising the LLC frame in transmission, see clause 8.3.3.3.1. Except LLC frame carrying APDUs of priority 3, no LPDU of an LLC frame is allowed to be transmitted until all LPDUs of previously transmitted LLC frame are cleared (i.e., transmitted entirely, if no acknowledgement is required, or acknowledged as received error free or timed out, if acknowledgement is required).

The LLC frames transmitted by any particular device shall be enumerated, thus allowing the receiver to distinguish between the different LLC frames in transmission; the sequence number of the LLC frame is indicated in the LFSN field of the MPH (see clause 8.1.3.1.1.5). In case transmission of LLC frame is interrupted by a frame of priority 3, the receiver detects the interruption by the value of priority and the new sequence number; further the receiver may resume reception of interrupted frame as described in clause 8.3.3.3.2.

A node shall not relay an LLC frame if not all of its comprising segments are received correctly; relaying segments that belong to an incompletely received LLC frame is not allowed. The received LLC frame shall be relayed using the usual LLC frame transmission path, as presented in Figure 8-4.

The next hop destination address (DDID, DNID) of the relayed frame shall be determined as defined in clause 8.3.2.2.

The priority of a relayed LLC frame shall be as indicated in the LPRI field of the LFH. The corresponding MA priority of the MPDUs carrying relayed frame shall be assigned based on the priority of the relayed LLC frame by using the rules defined in 8.4.3.2. The LLC frames with same value of LPRI shall be relayed in the order they have been received.

The content of a relayed LLC frame shall not be modified, except the HopsLft field of the LFH, which has to be decreased by 1 after each hop passed, as defined in clause 8.1.2.1.2. The LLC frames to be relayed shall not be decrypted.

The relayed LLC frame shall use the same settings for acknowledgement (parameters ACK_REQ in the PFH and ACK_TYPE in the MPH, see clause 8.3.3.1) as used in the received LLC frame.

8.1.3 Medium access control sub-layer (MAC)

The functional model of the MAC is presented in Figure 8-8. It is intended to describe in more detail the MAC functional block presented in Figure 8-1.

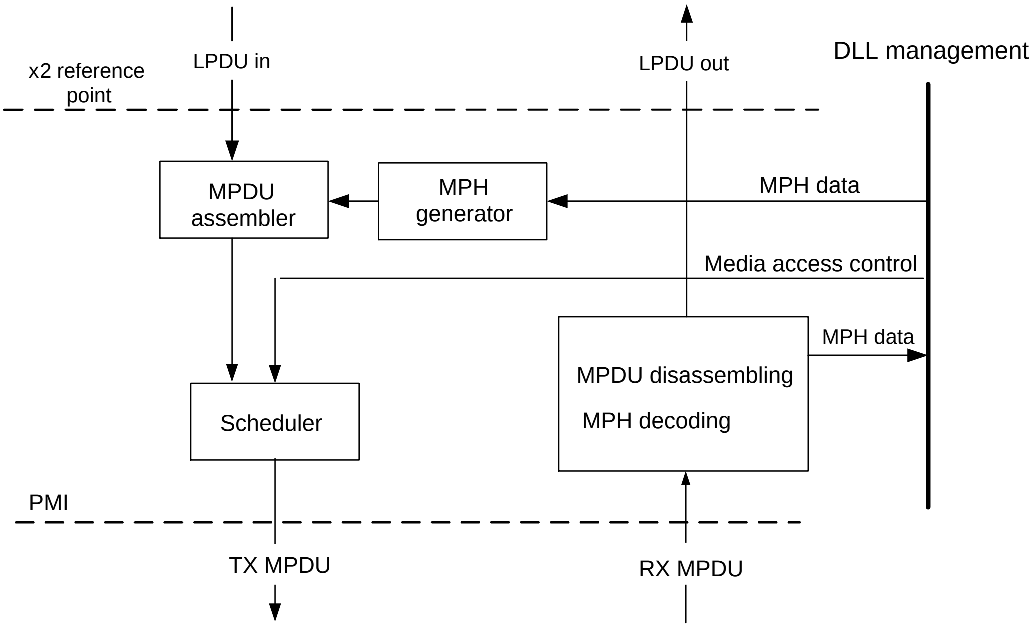


Figure 8-8 – Functional model of MAC

In the transmit direction, MPDUs are assembled from LPDUs that are passed over the x2 reference point and the MPH that is generated by the MAC. The MPDU assembler also adds the pad to meet the closest valid size of MPDU, as defined in clause 7.2.3.2.1 of [ITU-T G.9955].

The assembled MPDUs are scheduled for transmission according to the medium access procedure described in clause 8.2, using one or more established priorities queues. The medium access control primitives include the carrier sense (CRS) that indicates whether medium is busy or not. After being scheduled for transmission, the MPDU is passed to the PHY via PMI (see PMI_DATA.REQ primitives definition in clause 7.8 of [ITU-T G.9955]).

The MAC is also responsible for scheduling an Imm-ACK frame transmission by the PHY if acknowledgement is required. The data of the Imm-ACK frame (see clause 8.3.3.1.1) is passed to the PHY via the PMI interface (see [ITU-T G.9955], PMI_MGMT.REQ primitive).

In the receive direction, the MPDU enters via the PMI (PMI_DATA.IND primitive). The MPH is decoded, the MPDU is disassembled, the relevant MPDU parameters communicated in the MPH are passed to the DLL management via MAC_MGMT reference point, and the recovered LPDUs are passed to the LLC via the x2 reference point. The received acknowledgement data is passed via PMI interface (see [ITU-T G.9955], PMI_MGMT.IND primitive). Duplicates of the received MPDUs shall be dropped based on the LFSN number and the sequential order of the MPDU in the LLC frame.

8.1.3.1 Assembling of MPDUs

The LPDUs shall be assembled into an MPDU in the order they cross the x2 reference point. The LLC shall support the following MPDU assembling modes:

- Single segment MPDU mode (SS-MPDU)
- Multi segment MPDU mode (MS-MPDU)
- The MPDU assembling mode shall be identified by the MPDU_TYPE field of the MPH (see clause 8.1.3.1.1.1).

  NOTE – The term "MPDU" is general through the Recommendation and applies to both SS-MPDU and MS-MPDU.

The process of generation of LPDUs is described in clause 8.1.2.2.1. The assembling process is described in Figure 8-9.

An SS-MPDU shall include a single LPDU (excluding its LPCS field). An MS-MPDU shall include at least one LPDU, but no more than 16 LPDUs, starting with an LPDU that carries any segment of the LLC frame; this LPDU is numbered as LPDU#1 in Figure 8-9. All LPDUs in the MS-MPDU shall be sourced from the same LLC frame. The order of LPDUs in the MS-MPDU is vendor discretionary.

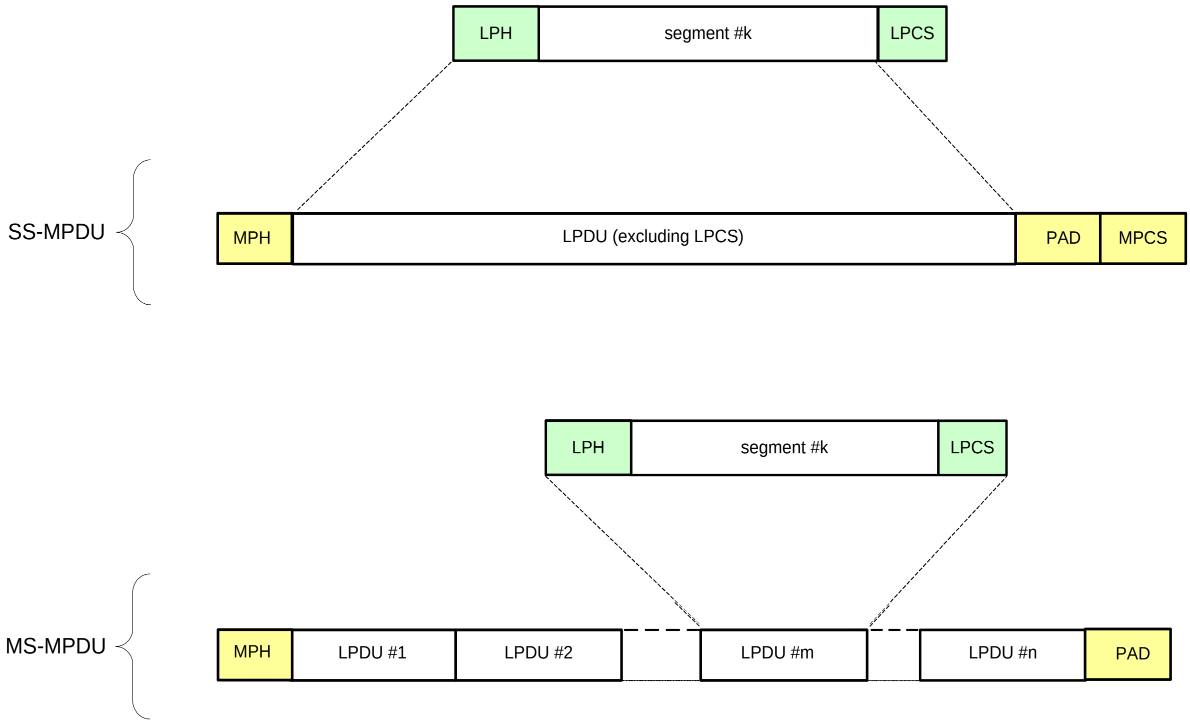


Figure 8-9 – Assembling of an SS-MPDU (top) and an MS-MPDU (bottom) from LPDUs

The size of the LPDU and the number of LPDUs in MS-MPDU shall be such that the time of MPDU transmission (time-on-the-wire) does not exceed the maximum allowed time on the medium, *MaxMediumTime* (see clause 8.1.4).

The position of the segment carried by the LPDU in the LLC frame is indicated in the LPH. The MPH shall be pre-pended to the first LPDU; MPH bit 0 of octet 0 shall be transmitted first. Octet 0 of the LPH of LPDU#1 of the MPDU shall be passed to the PHY first (see Figure 8-9). The MPH is defined in clause 8.1.3.1.1.

The pad field shall be appended to the last LPDU; the content of the pad is vendor discretionary; the valid size of the pad is up to 15 bytes; the receiver derives the size of the pad using the size of the LPDU indicated in the MPH.

The MPCS shall be attached after the PAD field in SS-MPDU and shall be calculated as specified in clause 8.1.3.1.4.

8.1.3.1.1 MPH format

The format of the MPH shall be as defined in Table 8-7. Octet 0 shall be passed to the PHY first.

Table 8-7 – MPH format

Field	Octet	Bits	Definition
MPDU Type	0	[2:0]	000 – SS-MPDU 001 – MS-MPDU 010-111 – Reserved by ITU-T
ACK_TYPE	0	[3]	Shall be set to 0 if ACK per frame is required and set to 1 if ACK per LPDU is required (see clause 8.3.3.1)
NOS	0-1	[9:4]	Number of segments comprising the LLC frame
LSPL	1	[15:10]	Last segment pad length, in bytes
LFSN	2	[7:0]	LLC frame sequence number
SNID	3-4	[15:0]	A 16-bit NODE_ID of the node that transmits the frame
DNID	5-6	[15:0]	A 16-bit NODE_ID of the node that shall receive the frame
SDID	7-8	[15:0]	A 16-bit DOMAIN_ID of the node that transmits the frame
DDID	9-10	[15:0]	A 16-bit DOMAIN_ID of the node that shall receive the frame
NEST	11	[7:0]	Indicates the value of $N_{est}(x)$ used for transmission
LPDU-L	12-13	[10:0]	Length of the LPDU
TP-REQ	13	[12:11]	Request for transmission parameters recommendation
PRI	13	[14:13]	MA priority of the MPDU
Reserved	13	[15]	Reserved by ITU-T
Reserved	14	[7:0]	Reserved by ITU-T
MHCS	15	[7:0]	HCS (CRC-8)
NOTE – Bits reserved by ITU-T shall be set to 0 by the transmitter, and shall be ignored by the receiver.			

8.1.3.1.1.1 MPDU type

This 3-bit field identifies the type of the MPDU. The value shall be formatted as unsigned integer. When set to 000, SS-MPDU mode is used, when set to 001, MS-MPDU mode is used. Other values are reserved by ITU-T.

8.1.3.1.1.2 ACK_TYPE

This 1-bit field identifies the type of acknowledgement as defined in clause 8.3.3.1.

8.1.3.1.1.3 Number of segments (NOS)

This 6-bit field indicates the total number of segments comprising the LLC frame minus 1, represented as an unsigned integer in the range from 1 to 64. The value of NOS shall be computed as:

NOS = *ceiling*(N_{LLC}/N_{SEG}), where N_{LLC}, N_{SEG} are defined in clause 8.1.2.2.1.

8.1.3.1.1.4 Last segment pad length (LSPL)

This 6-bit field indicates the length of the pad, in bytes, used in the last segment of the LLC frame, represented as an unsigned integer in the range from 0 (no padding) to 63.

8.1.3.1.1.5 LLC frame sequence number (LFSN)

This 8-bit field identifies to which LLC frame belong LPDUs comprising the MPDU. The value shall be formatted as unsigned integer in the range from 0 to 255. The value of LFSN shall be incremented by 1 (with wrap around) for each subsequent transmitted LLC frame (either originated by the node or relayed).

8.1.3.1.1.6 SNID, DNID

A 16-bit value of source node NODE_ID and destination node NODE_ID represented as unsigned integer with constraints determined by the ITU-T G.9955/9956 addressing scheme defined in clause 8.1.2.4.

8.1.3.1.1.7 SDID, DDID

A 16-bit value of source node DOMAIN_ID and destination node DOMAIN_ID represented as unsigned integer with constraints determined by the ITU-T G.9955/9956 addressing scheme defined in clause 8.1.2.4.

8.1.3.1.1.8 NEST

The NEST field indicates the value of $N_{est}(x)$ that the transmitter calculated as defined in clause 8.2.2.3 divided by 4 and rounded to the closest integer.

8.1.3.1.1.9 Length of the LPDUs (LPDU_L)

This 11-bit field indicates the length of the LPDUs carried by the MPDU in bytes, expressed as an unsigned integer. The valid range is from 5 to 1680 bytes. The length of the LPDU shall always exceed the size of the pad (if non-zero) applied at the end of the MPDU (see Figure 8-9).

8.1.3.1.1.10 Transmission parameters request (TP-REQ)

This 2-bit field indicates that the receiver of the frame shall provide to the requesting node a TP report with the content depending on the content of the TP-REQ field as follows:

00 – no TP report required;

01 – a full TP report using TP.ind message (see clause 8.5.4) is requested;

10 – a full TP report using Extended Imm-ACK (see clause 8.3.3.1) is requested;

11 – a partial TP report is requested.

This field shall be set to 00 for all broadcast and multicast transmissions (if DNID = 0xFFFF).

The content of full TP report and partial TP report is defined in clause 8.5.4 and in clause 8.3.3.1.1, respectively. If the transmitter requesting a TP report (full or partial) have not received it or received a report that is incomplete, it may repeat the same type of request in the following frame.

NOTE – A receiver that gets a TP request of a particular type, but not capable to provide a response in the Imm-ACK frame associated with the frame requesting the TP report, is expected to be ready for the same report in Imm-ACK associated with the next frame from the same source.

8.1.3.1.1.11 Priority (PRI)

This 2-bit field indicates the MA priority used for the current transmission, represented as unsigned integer.

8.1.3.1.1.12 MPH check sequence (MHCS)

The MHCS is an 8-bit cyclic redundancy check (CRC) and shall be computed over all the fields of the MPH in the order they are transmitted, starting with the LSB of the MPH byte 0 (clause 8.1.3.1.1) and ending with the MSB of the last byte of the MPH.

The MHCS shall be computed using the following generator polynomial of degree 8:

$$G(x) = x^8 + x^5 + x^4 + x^3 + x^2 + x^1 + 1.$$

The MHCS shall be constructed as follows:

1. The n bits of the MPH subject to MHCS are considered to be the coefficients of a polynomial of degree n-1, such that the LSB of the MPH byte 0 is the coefficient of the x^{n-1} term, and the MSB of the MPH byte 12 is the coefficient of the x^0 term. This polynomial is referred to as $N(x)$.
2. Replace the 8 highest-order coefficients of $N(x)$ with their ones'-complement values.
3. Multiply the result of step 2 by x^8. This result is referred to as $N_1(x)$.
4. Compute the MHCS as the ones-complement of the remainder of $N_1(x)$ divided by G(x).

The bits of the MHCS shall be transmitted in sequential order, starting from the coefficient of the highest order term (x^7), referred as the MSB, and continuing to the x^0 term.

8.1.3.1.2 MPDU check sequence (MPCS)

The MPCS is a 32-bit cyclic redundancy check (CRC) and shall be computed over all the fields of the SS-MPDU in the order they are transmitted, starting with the LSB of the MPH and ending with the MSB of the last octet of the PAD field.

The MPCS shall be computed using the following generator polynomial of degree 32:

$$G(x) = x^{32} + x^{28} + x^{27} + x^{26} + x^{25} + x^{23} + x^{22} + x^{20} + x^{19} + x^{18} + x^{14} + x^{13} + x^{11} + x^{10} + x^9 + x^8 + x^6 + 1.$$

The MPCS shall be constructed as follows:

1. The n bits of the SS-MPDU subject to MPCS are considered to be the coefficients of a polynomial of degree n-1, such that the LSB of the first octet of the MPH is the coefficient of the x^{n-1} term, and the MSB of the last octet of the PAD field is the coefficient of the x^0 term. This polynomial is referred to as $N(x)$.
2. Replace the 16 highest-order coefficients of $N(x)$ with their ones'-complement values.
3. Multiply the result of step 2 by x^{32}. This result is referred to as $N_1(x)$.
4. Compute the MPCS as the ones-complement of the remainder of $N_1(x)$ divided by $G(x)$.

The bits of the MPCS shall be transmitted in sequential order, starting from the coefficient of the highest order term (x^{31}), referred as the MSB, and continuing to the x^0 term.

8.1.3.2 Scheduler

The scheduler assigns MPDUs for transmission over the medium; it determines which MPDU to be passed for transmission to the PHY via PMI and the time this MPDU shall be passed to the PHY based on the medium access rules defined in clause 8.2. The scheduler also maintains MA priority queues necessary to support the medium access. MPDUs are assigned to the particular MA priority queue based on their priority and some additional criteria, as defined in clause 8.2.1.

8.1.4 DLL parameters

Bandplan dependent parameters are specified in Table 8-8.

Table 8-8 – DLL bandplan dependent parameters

Parameter	CENELEC	FCC
T_{IFG_MIN}	2.1 ms	2.7 ms
T_{AIFG}	2.1-2.6 ms	2.7-3.2 ms

DLL parameters independent of the band plan are specified in Table 8-9.

Table 8-9 – DLL parameters independent of the band plan

Parameter	Value
$T_{CP\text{-}MAX}$	250 ms
T_{TX_ON}	10 µs
T_{TS}	Same duration as 2 preamble symbols
MaxMediumTime	250 ms
MaxRetransmissions	Depends on the number of LPDUs (N_{LPDU}) comprising the LLC frame as specified in Table 8-10
MaxAckRxTime	The value is determined by the A_MGMT.REQ management primitive, see clause 8.4.3.4.1.
T_{CBIFG}	250 ms

Table 8-10 – Value of *MaxRetransmissions* for different number of LPDUs comprising an LLC frame

Number of LPDUs in an LLC Frame	Number of retransmissions per LLC frame
Up to 5	$N_{LPDU} \times 3$
6-10	$15 + N_{LPDU} \times 2$
11-20	$25 + N_{LPDU} - 10$
21-31	$35 + \text{floor}[(N_{LPDU} - 20) \times 1/2]$
32-46	$41 + \text{floor}[(N_{LPDU} - 32) \times 1/3]$
47-64	$46+\text{floor}[(N_{LPDU} - 47) \times 1/4]$

8.2 Medium access procedures

An ITU-T G.9955/9956 node of a standard profile shall supports prioritized contention-based medium access with four MA priorities denoted from 0 (lowest priority) to 3 (highest priority); nodes of low-complexity profile shall support two MA priorities, 0 and 2. The number of MA priorities supported by other profiles is for further study.

The prioritized contention-based medium access rules are defined in the following clauses. Other types of medium access are for further study.

8.2.1 Assignment of MA priorities

The classifier, in APC, assigns a certain priority to each APDU based on the QoS-related information carried by the primitives of the corresponding ADP or based on the relevant management primitives (see Annex I). This assigned APDU priority determines the priority of the LLC frame carrying the APDU and can be 0, 1 and 2. The priority value of 3 is assigned for pre-defined emergency data transfers only. All LLC frames carrying LCDUs (internal management communications) shall be assigned to priority 2.

The assignment of APDU priority depends on the application protocol but shall not exceed the number of MA priority queues supported by the node, which is 4 for standard profile and may be less for other profiles for other profiles (see clause 7).

Nodes of standard profile shall support:

– three MA priority queues for application data (priority 0, 1, and 2) and management data (priority 2);

– a MA priority queue for emergency signals (priority 3), that shall also be used for asynchronous beacons.

Nodes of other profiles shall support at least priority 0 for application data and priority 2 for management data (see clause 7); other supported priorities are for further study.

The MA priority of an MPDU shall be initially assigned the same priority as the LLC frame whose segment(s) it carries. If segments of an LLC frame are carried by multiple MPDUs, after transmission of the first MPDU carrying the LLC frame segments, the MA priority of all the remaining MPDUs carrying segments of the same LLC frame (including those carrying LPDUs assigned for retransmission) may be elevated from its initial value up to 2. All MPDUs sourced from an APDU with priority that is equal or greater than 2 shall use the initial priority.

8.2.2 Prioritized contention-based medium access

The prioritized contention-based medium access is defined in terms of contention periods (CP). The contention process starts at the beginning of the CP. The CP ends T_{IFG_MIN} after a node that won the contention completes transmission of the frame sequence, which includes the transmitted frame and the ACK frame, if required. A new CP starts immediately after the end of the previous CP.

The CP shall consist of 4 priority resolution periods that may overlap. Each such period is associated with a contention window (CW). The CW consists of a variable number of fixed duration time slots (TS). The number of slots assigned to each CW shall vary; the minimum number of TS is 1. The size of the CW for each priority shall be set as specified in clause 8.2.2.3.

The CP shall start with the CW associated with the highest priority. A CW associated with a certain priority shall not start prior to a CW associated with a higher priority. Setting of a CW size associated with a given priority shall be as specified in clause 8.2.2.3.

An MPDU assigned with a specific MA priority is allowed to be transmitted during the CW associated with the same or lower priority according to the back-off procedure specified in clause 8.2.2.2.

The medium access process includes the following steps:

1. Nodes shall contend for transmission of an MPDU based on the MA priority of the MPDU assigned by the scheduler.
2. Nodes shall contend for transmission of an MPDU during the CW that is associated with the MA priority assigned to this MPDU or during a CW associated with a lower MA priority.
3. Nodes that won the contention may transmit their MPDU.
4. Nodes that lost the contention shall use the back-off rules specified in clause 8.2.2.2.
5. The next CP shall start T_{IFG_MIN} period after the transmission of the frame sequence is complete, i.e., after the ACK frame is sent (in case ACK was requested by the transmitter) or after the frame was transmitted (in case no ACK was requested by the transmitter).

The procedure of transmission during the CP is illustrated in Figure 8-10. The figure presents the positions and sizes of CWs for a particular CP in which a node that won the contention for CW_0 sends a low priority MPDU followed by an Imm-ACK frame.

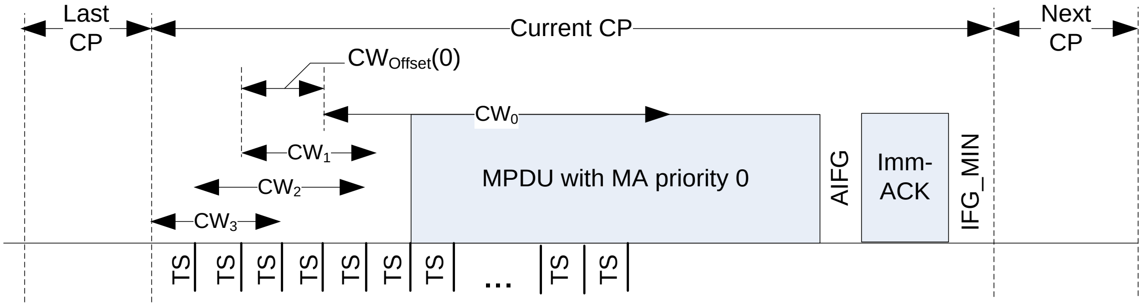


Figure 8-10 – Illustration of a typical transmission during the CP

8.2.2.1 CW offset setting

The CW offset is defined as a number of TS between the start of the CW and the start of the CW associated with the next higher priority. The values of CW offsets associated with a specific priority (relative to the CW associated with the next higher priority) is determined by the domain master and communicated to the nodes of the domain via the beacon (see clause 8.4.2.1) and set using the MAC_MGMT.REQ primitive CW_{Offset} (see clause 8.4.3.3.1). The default values of CW_{Offset} are presented in Table 8-11. The CW_{Offset} for priority 0 is shown in Figure 8-10.

8.2.2.2 CP back-off procedure

To support the back-off procedure described in this clause, each node shall maintain at least the following back-off parameters:

- A back-off counter (BC) per transmission attempt
- A CW size for each MA priority.

The BC determines the number of TS the node shall wait before it may begin the transmission if medium is idle. The duration of the TS is denoted as T_{TS} and shall be as specified in clause 8.1.4.

The CW size determines the range from which the back-off value shall be picked before transmission, which is the function of MA priority and further denoted CW(*x*), where *x* stands for the MA priority. The size of CW(*x*) is expressed in the number of TS.

Node that survives the priority resolution period preceding the start of the CW associated with the MA priority of the MPDU it intends to transmit (see clause 8.2.1), shall compete in the CW using its back-off parameters for that MA priority, and shall act according to the following procedure before starting a transmission in a CP:

1. Pick a random value of the BC in the range 0 to CW(*x*)-1.
2. If the picked value of BC is zero, the node shall start transmitting its frame within a time window of T_{TX_ON} (see clause 8.1.4) after the start of the first TS of the CW(*x*).
3. If the picked value of BC is not zero, the node shall decrement its BC upon completion of each TS in which it detects no PHY preamble (the medium is idle).
4. If, upon completion of certain TS, the value of BC is zero, the node shall start transmitting its frame within a time window of T_{TX_ON} (see clause 8.1.4) after the end of this TS. The BC shall be disabled till the next CP.
5. If a node detects a PHY preamble during a TS and its BC at this TS is not zero, it shall not transmit in this CP and shall defer transmission to the next CP.

6. In case a node gets an MPDU to transmit and no previous transmission was detected for more than $T_{CP\ MAX}$ (see clause 8.1.4) from the start of the CP, the node may transmit the MPDU immediately.

7. In the next CP, nodes shall repeat steps 1-6 above.

If a node that contends in a CP gets an MPDU to transmit after the start of a particular CW, it may still contend for this MPDU in this CW using the back-off procedure defined in this clause, but only if the MA priority of the MPDU is equal to or higher than the priority associated with the CW. The node shall pick a random value of the BC in the CW in the same way as nodes that had the MPDU ready to transmit prior to the start of the CW. Alternatively, the node may defer transmission to the next CP.

8.2.2.3 CW update procedure

The update procedure shall be performed by a node in the following cases:

– After every transmitted frame, for the MA priority of this frame.

– After every received frame with no error in MPH, for the MA priority of this received frame (indicated in MPH).

– After every $T_{CWUpdate}$ time period in which a node does not transmit and does not detect transmissions (i.e., no preamble was detected) from other nodes, for all those MA priorities, *x*, for which the CW(*x*) is greater than $CW_{InActMin}(x)$ associated with the MA priority *x*; the value of the parameter $CW_{InActMin}(x)$ is expressed in TS and its default value is indicated in Table 8-9. If $T_{CWUpdate}$ is set to 2550 ms the update procedure due to the $T_{CWUpdate}$ time period shall not be performed.

Upon initialization, the node shall set its CW value for each priority to an initial value, $CW_{Init}(x)$, expressed in TS.

The CW update procedure is iterative. The parameters of the update procedure for a CW of the n-th iteration, *CW*(*x*,*n*), associated with MA priority *x* are $CW_{tx}(x,n)$ and $CW_{min\_tx}(x,n)$ that represent the actual value of the CW and the value of CW_{min} expressed in TS used in the last transmitted frame before the CW update, where CW_{min} is the number of TS from the beginning of the CW to the instant of actual frame transmission.

For each transmission, the transmitter shall calculate the value of $N_{est}(x,n)$ as follows:

$$N_{est}(x) = \frac{CW_{tx}(x) - CW_{\min -tx}(x)}{CW_{\min -tx}(x)}$$

All nodes shall update the CW used for the last transmission of a MA priority, *x*, by using the following procedure:

1. Calculate parameter $\upsilon_{est}(x,n)$ at the *n*-th iteration as follows:

   - $\upsilon_{est}(x,n)$=1 if the value of the NEST field on the MPH (in the transmitted or in the received frame, respectively) is set to zero and $\upsilon_{est}(x,n)$=1/(4×NEST) if the value of NEST field is non-zero;
   - $\upsilon_{est}(x,n)$=1 after every $T_{CWUpdate}$ time period in which a node does not transmit and does not detect transmissions.

2. Calculate parameter α(*n*) at the *n*-th iteration as follows:

$$\alpha = \begin{pmatrix} \dfrac{\gamma \times \delta}{1+\gamma \times \delta}, & if\ \gamma < \gamma_{\max} \\ 1, & if\ \gamma = \gamma_{\max} \end{pmatrix}$$

where $\delta = \upsilon_{est}(x,n) / \upsilon_{avg}(x,n-1)$, $\upsilon_{avg}(x,n-1)$ is the weighted average value of $\upsilon_{est}(x)$ over the past n-1 iterations, and $\upsilon_{avg}(x,0) = \dfrac{A}{CW_{Init}(x) - B}$.

3. Calculate $\upsilon_{avg}(x,n)$ at the n-th iteration as follows:

$$\upsilon_{avg}(x) = \alpha \times \upsilon_{avg}(x) + (1-\alpha) \times \upsilon_{est}(x)$$

4. Update $CW(x)$ as follows:

$$CW(x) = \frac{A}{\upsilon_{avg}(x)} + B$$

The default values of the back-off update parameters shall be as specified in Table 8-11:

Table 8-11 – ITU-T G.9956 default backoff parameters

Parameter	Value
$CW_{Init}(0)$	160
$CW_{Init}(1)$	160
$CW_{Init}(2)$	8
$CW_{Init}(3)$	8
γ	9
A	2.7
B	0
$CW_{InActMin}$ (0)	8
$CW_{InActMin}$ (1)	8
$CW_{InActMin}$ (2)	160
$CW_{InActMin}$ (3)	160
TCWUpdate	10 ms
CW_{offset} (0)	3 TS
CW_{offset} (1)	2 TS
CW_{offset} (2)	2 TS

8.2.2.4 Adjustment of backoff parameters

Backoff parameters can be adjusted from their default values presented in Table 8-11 by the domain master or special nodes assigned by the domain master (proxies). The domain master (or its selected proxies) may assert the values of default back-off parameters only upon request from the upper layer management indicated by the A_MGMT.REQ primitive MAMode (see clause 8.4.3.4.1). The asserted values of the default back-off parameters are communicated to all nodes of the domain using special management messages. The details are for further study.

8.2.2.5 Starting and closing of a CP

Nodes shall start a new CP upon closure of the current CP, as defined in clause 8.2.1 or after a certain period of inactivity on the line. Nodes shall infer closure of a CP that was used for transmission according to one of the following methods:

– Duration-based

– ACK reception-based

– Timeout-based.

Starting of a CP after inferring persistent line inactivity is defined in 8.2.2.5.4.

8.2.2.5.1 Duration-based CP closure

Nodes that did not contend or backed off in a CP shall detect the frame transmitted during the CP. A node that received the preamble and the PFH of a transmitted frame without HCS error (see clause 7.2.3 in [ITU-T G.9955]), shall close the CP that was used for frame transmission T_{IFG_MIN} after the frame sequence ends, based on the duration that can be inferred from the related PFH fields of the transmitted frame in the frame sequence. If a request for ACK was indicated in the PFH, T_{IFG_MIN} shall be counted from the end of the ACK message transmission (assuming maximum value of T_{AIFG}); if no request for ACK was indicated in the PFH, T_{IFG_MIN} shall be counted from the end of the transmitted message.

The values of T_{IFG_MIN} and the valid minimum and maximum values for T_{AIFG} are defined in clause 8.1.4, the duration of the ACK shall be computed based on the Imm-ACK frame format, as defined in 8.3.3.

Figure 8-11 describes a duration-based CP closure (with and without ACK). The CP closes after the entire frame sequence duration.

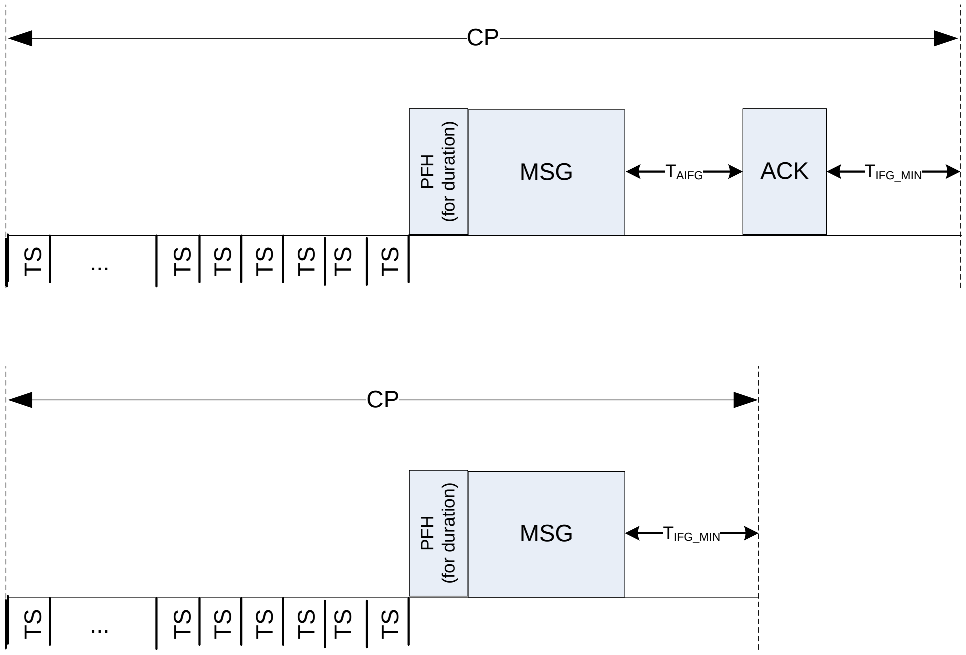


Figure 8-11 – Example of duration-based CP closure (with and without ACK)

8.2.2.5.2 Imm-ACK reception-based CP closure

Nodes that did not contend or backed off in a CP, and received an Imm-ACK frame, shall close a CP that was used for transmission at T_{IFG_MIN} + max(T_{AIFG}) – min(T_{AIFG}) after the end of the Imm-ACK frame. In case the received Imm-ACK frame was in response to a frame that originally inferred this receiver to close the CP based on duration, the receiver shall close the CP using the duration based CP.

Figure 8-12 describes an ACK reception-based CP closure.

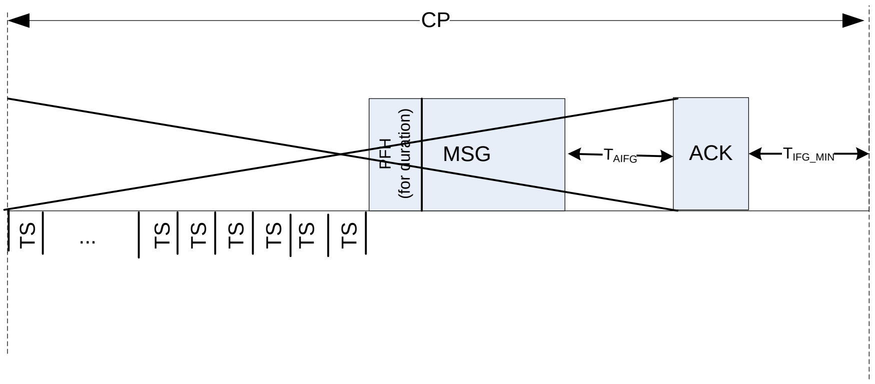


Figure 8-12 – Example of Imm-ACK reception-based CP closure

A node that transmitted a frame containing an Imm-ACK request, but did not receive an Imm-ACK during max(T_{AIFG}) period after the end of the frame, shall assume a collision of the transmitted frame and shall start a waiting period T_{CBIFG} counted from the end of the last symbol of the PFH of their transmitted frame. If during the T_{CBIFG} period no preamble of any subsequent frame was detected, the node shall consider the CP closed after T_{CBIFG}. The value of T_{CBIFG} is defined in clause 8.1.4.

Figure 8-13 illustrates this scenario.

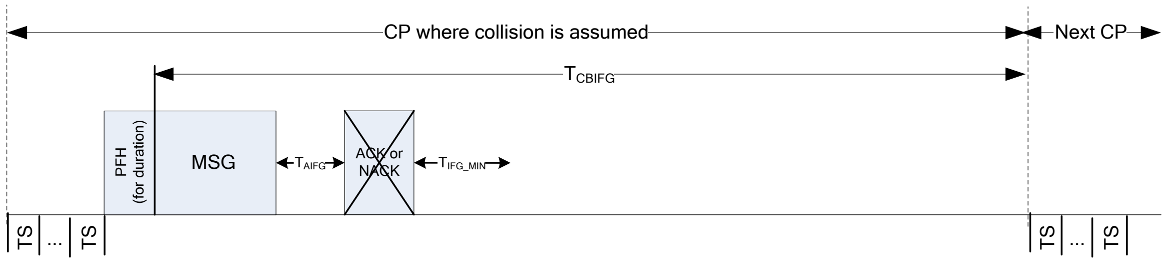


Figure 8-13 – Example of a CP closure in case no Imm-ACK is received

If during the T_{CBIFG} period a preamble of any subsequent frame was detected, the node shall re-set the CP closure based on that received frame, using methods described in clause 8.2.2.5.1, clause 8.2.2.5.2, or clause 8.2.2.5.3 depending on reception conditions.

8.2.2.5.3 Timeout-based closure

Nodes that did not contend or backed off in a CP shall attempt to receive the frame transmitted during the CP. A node that received a frame with the HCS of the PFH detected with error, and no Imm-ACK frame or preamble or PFH of any subsequent frame or any other frame related to the CP was detected for at least T_{CBIFG} from the end of the last symbol of the PFH, shall consider the CP closed after T_{CBIFG} (clause 8.1.4).

Figure 8-14 illustrates this scenario.

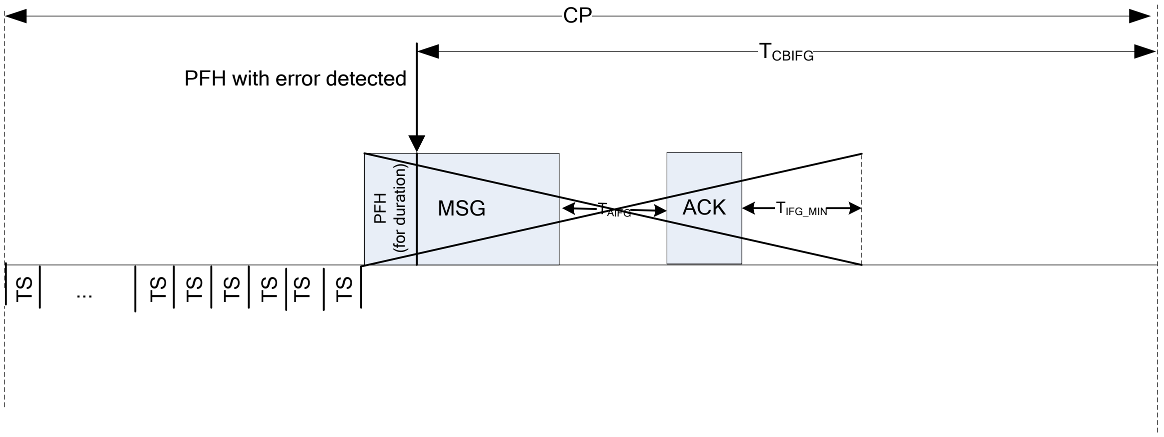


Figure 8-14 – Example of a timeout based CP closure

8.2.2.5.4 Starting of a CP upon persistent line inactivity

If a node has a new frame ready for transmission after no activity on the line was detected for longer than $T_{CP\text{-}MAX}$ period, the node shall consider that the previous CP as closed and thus it may immediately transmit the frame. Other nodes shall close the CP based on the reception conditions of this transmitted frame, as described in clause 8.2.2.5.1, clause 8.2.2.5.2 and clause 8.2.2.5.3. Nodes shall not close the CP if no preamble of a frame was detected.

Figure 8-15 describes this scenario.

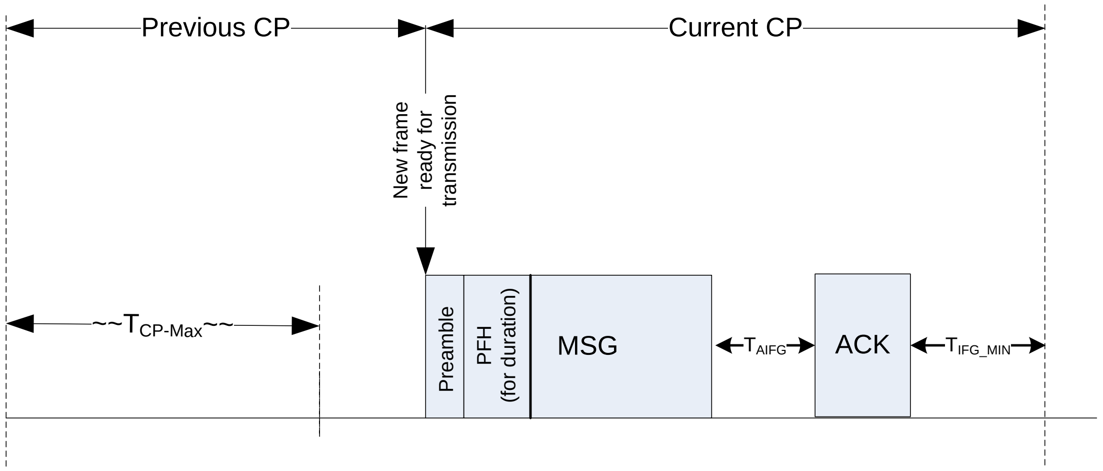


Figure 8-15 – Starting a CP after persistent line inactivity

8.3 Domain operation

8.3.1 Addressing scheme

8.3.1.1 Node identification

For the purpose of identification in the network:

1. Each node shall be assigned by the manufacturer with a 64-bit unique EUI-64 address (global address) using guidelines defined in [EUI-64].
2. Each node, after its registration into a particular domain (see clause 8.5.2), shall be assigned with a short address that includes:
   - a 16-bit DOMAIN_ID that is a unique identifier of the domain the node has registered with;

  - a 16-bit NODE_ID that is unique node identifier inside of the domain the node has registered with.

8.3.1.2 Addressing modes

The following addressing modes are defined:

- Short addressing mode
- Extended addressing mode

The defined addressing modes determine the node identification method to be used when transmitting or receiving a packet:

- Extended addressing mode:
  - By its EUI-64 address
- Short addressing mode:
  - By its NODE_ID and DOMAIN_ID
  - By its MULTICAST_ID(s) and DOMAIN_ID
  - By the BROADCAST_ID and DOMAIN_ID.

The Short Addressing mode shall be used in all cases when short address (NODE_ID, DOMAIN_ID) is available. Extended Addressing mode shall be used only when short addresses are not available (e.g., during node registration and network initialization, as defined in clause 8.5.2). Valid values of node and domain identifiers in short addressing mode are presented in Table 8-12.

8.3.1.2.1 DOMAIN_ID

DOMAIN_ID is an identifier for a particular domain and is assigned at the creation of the domain. A DOMAIN_ID can be assigned using manual mode of configuration or self configuration mode. Selection of DOMAIN_ID and DOMAIN_ID conflict detection and resolution shall be performed as detailed in clause 8.5.1.

A BROADCAST_DOMAIN_ID is reserved for broadcast to all domains.

8.3.1.2.2 NODE_ID

The NODE_ID is an identifier of a particular node in a domain. In conjunction with a DOMAIN_ID, a NODE_ID identifies the source and destination nodes in the network. Valid values of NODE_ID are summarized in Table 8-12.

A unique in the domain NODE_ID value is assigned to a node as a part of the registration to a domain, as described in clause 8.5.2, creating a unique DOMAIN_ID and NODE_ID combination. The assigned NODE_ID is valid until the node is resigned from the domain. After the node has resigned from the domain, the same value of the NODE_ID can be assigned to a new registered node in that domain.

The reserved NODE_IDs 0xFFFE and 0xFFFF are intended to identify a source node that has not yet been assigned a NODE_ID. The use of reserved value of NODE_ID is defined in clause 8.5.2.

8.3.1.2.3 MULTICAST_ID

The range of MULTICAST_IDs is from 0x8000 to 0x9FFF.

Details of multicast addressing are for further study.

8.3.1.2.4 BROADCAST_ID

BROADCAST_ID is the same for all nodes in all domains and intends for broadcast inside the domain. When used in conjunction with a BROACAST_DOMAIN_ID, as defined in Table 8-12, it allows sending a message to all nodes of all domains in the network.

8.3.1.2.5 Reserved IDs

Node identifiers in the range between 0xA000 and 0xFFFD are reserved by ITU-T.

Table 8-12 – Definition of node identifiers for short addressing mode

	Parameter	Valid values	Description
Node identification	Reserved IDs	0xA000-0xFFFD	Reserved by ITU-T
	NODE_ID	0x0000-0x7FFF	The NODE_ID is used by registered nodes for communicating with other nodes within their domain or in other domains.
		0xFFFE, 0xFFFF	Reserved (see clause 8.3.1.2.2)
	BROADCAST_ID	0xFFFF	This ID is for broadcasts within identified domain when used as the destination node ID. Other use is for further study.
	MULTICAST_ID	0x8000-0x9000	These IDs are reserved for multicast within identified domain when used as the destination node ID. Other use is for further study.
Domain Identification	DOMAIN_ID	0x0000-0xFFFF	Domain IDs for manual configuration or self-configuration modes
	BROADCAST_ DOMAIN_ID	0xFFFF	Domain ID reserved for broadcast to all domains.

8.3.2 LLC frame transmission methods

The transmission methods are defined for both intra-domain connections (between nodes of the same domain) and inter-domain connections (between nodes of different domains).

8.3.2.1 Intra-domain single-hop transmission

8.3.2.1.1 Unicast transmission

An intra-domain unicast transmission enables communication between a single source node and a single destination node that are both associated with the same domain and have a direct connection at the physical layer. An intra-domain unicast transmission may be for data or management. Unicast transmission is allowed for both SS-MPDU and MS MPDU frames.

An intra-domain unicast transmission is uniquely identified by the tuple (source/destination DOMAIN_ID, source NODE_ID, destination NODE_ID). Intra-domain unicast transmissions may require acknowledgement, according to the rules specified in clause 8.3.3.

To unicast an MPDU using short addressing mode, the transmitter shall set the following values of addressing fields in the MPH:

- The SDID and DDID fields – to the DOMAIN_ID of the domain the transmitter is associated with.
- The SNID field – to the NODE_ID of the source node.
- The DNID field – to the NODE_ID of the destination node.
- The MESH header shall not be transmitted (MHI field of LFH shall be set to 0).

A receiver with a DOMAIN_ID that matches the DDID field indicated in the MPH and with a NODE_ID that matches the DNID indicated in the MPH shall recover LPDUs contained in the received MPDU, reassemble the LLC frame and forward it to the upper layer. If either the DDID field or DNID field indicated in the MPH do not meet, respectively, the DOMAIN_ID and the NODE_ID of the receiver, the receiver shall drop the MPDU.

If the one-hop unicast transmission is performed using extended addressing mode, the source and destination addresses (SA, DA) shall be defined in the MESH field of the LFH; the corresponding address fields of the MPH (SNID, DNID) shall be set to the values of the corresponding NODE_ID, if available, or to 0xFFFF if either or both values of NODE_ID are not available. In case DNID = 0xFFFF, the MPDU shall be broadcasted, as described in clause 8.3.2.1.2. The RELI field of the LFH shall be set to 0.

NOTE – It is always beneficial to use short addressing mode, therefore extended addressing mode should be used only if short address of either the destination or the source or both are not available.

The receiver shall drop all LLC frames that contain a MESH header indicating single-hop transmission (RELI field of the LFH is set to 0) with a DA that is different from the EUI of the receiver.

Acknowledgements by the receiver and retransmissions by the transmitter, if required, shall be performed as specified in clause 8.3.3.

8.3.2.1.2 Broadcast transmission

An intra-domain one-hop broadcast transmission enables communication between a single source node and all other nodes in the same domain that have direct connection at the physical layer. An intra-domain broadcast transmission may be for data or management. Broadcast transmission is allowed only for SS-MPDU frames. Transmission parameters are vendor discretionary, but shall be limited to the mandatory values supported by nodes of all profiles.

A broadcast transmission is uniquely identified by the tuple (source/destination DOMAIN_ID, source NODE_ID).

Acknowledgements shall not be requested for intra-domain broadcast transmissions.

To broadcast an MPDU, the transmitter shall set following values of addressing fields in the MPH:

- The SDID and DDID fields – to the DOMAIN_ID of the domain the transmitter is associated with.
- The SNID field – to the NODE_ID of the source node.
- The DNID field – to the BROADCAST_ID=0xFFFF.

A receiver with a DOMAIN_ID that matches the DDID field indicated in the MPH shall recover LPDUs contained in the received MPDU, reassemble the LLC frame and forward it to the upper layer. Other receivers shall drop the MPDU.

8.3.2.1.3 Multicast transmission

An intra-domain multicast single-hop transmission enables a node to send LLC frames to a selected set of destination nodes in the domain.

The protocol for single-hop multicast transmission is for further study.

8.3.2.2 Intra-domain multi-hop transmission

This Recommendation allows performing mesh networking by using either L2 routing (by relaying the LLC frames, as defined in clause 8.1.2) or L3 routing (by relaying the protocol data units above the A-interface). Prior to initialization, each domain shall be set into a particular routing mode that either allows intra-domain L2 routing (L2 relaying mode) or that disallows it (no L2 relaying is allowed for any node of the domain). The mode is determined by the A_MGMT.REQ primitive RelayMode (see clause 8.4.3.4.1).

The domain master shall inform all nodes registering into the domain whether L2 relaying is enabled or not. If L2 relaying is enabled, all relayed communications in the domain shall be performed as defined in this clause. If L2 relaying is disabled, only single-hop communications in the domain (as per clause 8.3.2.1.1) are allowed, and nodes shall not relay any of the received frames, except beacon frames, if assigned by the domain master (see clause 8.3.4).

For a node, L2-relaying capability is optional. Nodes shall indicate to the domain master their L2-relaying capabilities as they register into the domain. If L2-relaying within the domain is enabled, nodes that are relay-capable may be assigned as domain relays for unicast, multicast, or broadcast using the procedure described in clause 8.5.6.

The L2 relaying functionality of nodes shall be as follows:

- All nodes (including not relay-capable) shall be capable to source the relayed transmission (create a mesh header) and to receive LLC frames that were relayed to it as a final destination (process a frame that includes a mesh header).
- When L2 relaying is enabled, all relay-capable nodes assigned as domain relays shall perform full relaying functionality (i.e., as an LLC frame originator, a relaying node, and a receiver of relayed LLC frames) for unicast, multicast, or broadcast using the procedure described in clause 8.3.2.2.1, clause 8.3.2.2.2, and clause 8.3.2.2.3, respectively.
- All relay-capable nodes that are not assigned as domain relays shall discard all frames for which this node is not a final destination.
- When L2 relaying is disabled, all nodes in the domain shall not generate LLC frames to be relayed and discard all received LLC frames for which that node is not the final destination.

  NOTE – A relay-capable node that is not assigned as a domain relay, may still be assigned as a beacon relay (BPR, see clause 8.3.4).

8.3.2.2.1 Unicast multi-hop transmission

Intra-domain unicast multi-hop transmission allows an originating node to unicast LLC frames to a final destination node via one or more intermediate L2-relay. The destination node and all relaying nodes shall be from the same domain as the originating node.

The next hop destination address and previous hop source address applied by originating nodes, relay nodes, and destination nodes shall comply with the relaying database (routing tables or routing information in the SR header of the transmitted LLC frame). The rules for generation and updating the relaying database shall be as described in clause 8.5.6.

8.3.2.2.1.1 Operation of the originating node

For LLC frames that require relaying, the originating node shall perform as follows:

- Set both the V-bit and F-bit of the ADM-fields of the mesh header to 1 (to indicate relaying based on short addresses).
- Set the HopsLft field of the mesh header to the maximum number of hops the LLC frame is allowed to be relayed. If the LLC frame reaches the final destination node before the field reaches 0, the LLC frame shall still be processed by the final destination node.
- Set the originator address field of the mesh header to the node's NODE_ID.
- Set the final address field of the mesh header to the NODE_ID of final destination node.
- Use the intra domain single-hop unicast transmission method described in clause 8.3.2.1 to transmit the LLC frame to the first relay node based on the information maintained in the relaying database (routing table or SR header) to reach the node with NODE_ID matching the final address field indicated by the mesh header.

8.3.2.2.1.2 Operation of a relay node

If a node assigned as a domain relay receives an LLC frame containing a mesh header, and the receiving node is not the final destination node (i.e., the address of the receiving node does not match the value indicated on the final address field DA of the mesh header), and the DNID of the node the LLC frame was sent to and the SNID of the node the LLC frame was received from meets the relaying database, and the values of SA and DA indicated in the mesh header meet the relaying database, the receiving node shall perform the following relaying procedure and, if either of the conditions above is not met, the received frame shall be discarded:

- If the HopsLft field in the received LLC frame is greater than 0:
  - The value of HopsLft field of the relayed frame shall be the value indicated on HopsLft field of the received frame decremented by 1.
  - Use the intra-domain single-hop unicast transmission method described in clause 8.3.2.1 to transmit the LLC frame to the next node based on the information maintained in the relaying database or in the SR header to reach the node with NODE_ID matching the final address field indicated on the mesh header.
  - Use LLC frame relaying rules shall be as defined in clause 8.1.2.4.
- If the HopsLft field in the received LLC frame equals 0, the received LLC frame shall be dropped.
- If the NODE_ID indicated on the DA field in the mesh header of the received LLC frame does not meet the current relaying database, this received LLC frame shall be dropped.
- If the NODE_ID of the node from which the frame was received (i.e., SNID field of the MPH) or the value of SA indicated in the mesh header does not meet the relaying database, the received LLC frame shall be dropped.

8.3.2.2.1.3 Operation of the destination node

If a node receives an LLC frame containing a mesh header, and the receiving node is the final destination node (i.e., the NODE_ID of the receiving node matches the value indicated on the final address field of the mesh header), and the NODE_ID of the source the frame was received (i.e., the SNID field of the MPH in the received frame) meets the relaying database, the receiving node shall accept the received LLC frame and pass it to the AE via the A-interface. If either of these conditions is not met, the received frame shall be dropped.

8.3.2.2.2 Broadcast multihop transmission

Intra-domain broadcast multi-hop transmission enables an originating node to broadcast LLC frames to all nodes of the domain via one or more intermediate L2-relay that all belong to the same domain.

The next hop destination address and previous hop source address applied by originating nodes, relay nodes, and destination nodes shall comply with the broadcast relaying database (routing tables or SR header). For some hops the DNID may get a value of BROADCAST_ID = 0xFFFF. The rules for generation and updating the relaying database shall be as described in clause 8.5.6.

NOTE – This broadcast mechanism can be converted to flooding if relaying database is set accordingly.

8.3.2.2.2.1 Operation of the originating node

For LLC frames that require relaying, the originating node shall perform as follows:

- Set both the V-bit and F-bit of the ADM -fields of the mesh header to 1 (to indicate relaying based on short addresses).
- Set the HopsLft field of the mesh header to the maximum number of hops the LLC frame is allowed to be relayed. If the LLC frame reaches the final destination node before the field

reaches 0, the LLC frame shall still be processed by the final destination node. Set the originator address field of the mesh header to the node's NODE_ID.

- Set the originator address field of the mesh header to the node's NODE_ID.
- Set the final address field of the mesh header to 0xFFFF.
- Use the intra domain single-hop unicast transmission method to transmit the LLC frame to the first relay node or single-hop broadcast transmission method, depending on the information maintained in the broadcast relaying database to reach all the nodes in the domain. The single-hop transmission shall be performed as described in clause 8.3.2.1.

8.3.2.2.2.2 Operation of a relay node

If a node assigned as a broadcast domain relay receives an LLC frame containing a mesh header with BROADCAST_ID, and the NODE_ID of the source node the LLC frame was sent from (i.e., the SNID field of the MPH in the received frame) meets the broadcast relaying database, and the value of SA indicated in the mesh header meets the relaying database the receiving node shall:

– accept the received LLC frame (if this frame was not already received once), recover the ADP, and pass it to the AE via the A-interface;

– perform the broadcast relaying procedure described in this clause.

If any of the conditions above is not met, the received frame shall not be relayed.

Broadcast relaying procedure:

- If the HopsLft field in the received LLC frame is greater than 0:
  - The value of HopsLft field of the relayed frame shall be the value indicated on HopsLft field of the received frame decremented by 1.
  - Use the intra-domain single-hop unicast transmission method to transmit the LLC frame to the next node or single-hop broadcast transmission, based on the information maintained in the broadcast relaying database. Other relaying rules shall be as defined in clause 8.1.2.4.
- If the HopsLft field in the received LLC frame equals 0, the received LLC frame shall be dropped.
- If the NODE_ID of the node from which the broadcast relay received the frame (i.e., the SNID field of the MPH in the received frame) does not meet the broadcast relaying database or the value of SA indicated in the mesh header does not meet the relaying database, the received LLC frame shall be dropped.

8.3.2.2.2.3 Operation of the destination node

If a node receives an LLC frame containing a mesh header with the final address equal to 0xFFFF, and this node is not assigned a broadcast relay, and the NODE_ID of the source the frame was received from (i.e., the SNID field of the MPH in the received frame) meets the broadcast relaying database, and the value of SA indicated in the mesh header meet the relaying database, the receiving node shall accept the received LLC frame (if this frame was not already received once), recover the ADP, and pass it to the AE via the A-interface. If either of the last two conditions is not met, the received frame shall be dropped.

8.3.2.2.3 Multicast multihop transmission

Intra-domain multicast multi-hop transmission enables an originating node to send LLC frames to a selected set of destination nodes in the domain via one or more intermediate L2-relaying nodes that all belong to the same domain.

The protocol for multi-hop multicast transmission is for further study.

8.3.2.3 Inter-domain single-hop transmission

Transmission methods for single-hop inter-domain communications are for further study.

8.3.2.4 Inter-domain multi-hop transmission

Transmission methods for multi-hop inter-domain communications are for further study.

8.3.3 Retransmission and acknowledgement

The retransmission and acknowledgment protocol comprises the following steps:

- The transmitter sends a frame with a request to acknowledge the MPDU and indicates the type of the acknowledgement in ACK_REQ field of the PFH and ACK_TYPE field of the MPH.
- The receiver acknowledges the MPDU by transmitting an Imm-ACK frame with format defined in 8.3.3.1.1, using the type of acknowledgement requested by the transmitter and using rules described in clause 8.3.3.1, clause 8.3.3.3. No Imm-ACK frame shall be sent if acknowledgement is not requested by the transmitter.
- The transmitter retransmits the MPDU or its tributary LPDUs indicated in the received Imm-ACK frame using the protocol described in clause 8.3.3.3.

8.3.3.1 Acknowledgment of unicast frames

To request acknowledgement on the MPDU carried by a unicast PHY frame, the transmitter shall indicate the request in the ACK_REQ field of the PFH, as defined in clause 7.2.3.2 of [ITU-T G.9955], and in the ACK_TYPE field of the MPH, as defined in clause 8.1.3.1.1. All the ACK types presented in Table 8-13 shall be supported:

Table 8-13 – Supported ACK request types

ACK_REQ (PFH)	ACK_TYPE	Type of request	Receiver action (Note)
0	N/A	No acknowledgement required	No Imm-ACK frame shall be sent
1	0	Requires an ACK/NACK for the whole MPDU only	An Imm-ACK frame shall be sent indicating whether or not the MPDU was received error-free. The bit "ACK/NACK" in the PFH of the Imm-ACK frame shall be set to 1 if the MPDU was received error-free and 0 otherwise. The value of 0 shall be interpreted as "NACK".
1	1	Requires an ACK for the MPDU with indication of each LPDU of the MPDU that was received in error	An Imm-ACK frame shall be sent indicating in the ACKI field of the PFH of the Imm-ACK frame all the LPDUs of the MPDU that were received in error. The coding of ACKI field shall be as defined in clause 8.3.3.1.1.
NOTE – Receiver actions in more details are described in clause 8.3.3.3.2.			

If acknowledgement is required for a unicast frame, the acknowledging node (receiver) shall respond to the reception of a frame with an Imm-ACK frame that is a Type 3 frame and consists of a preamble and a PFH, but no payload, as specified in clause 7.2.1 of [ITU-T G.9955]. The format of the Imm-ACK frame header shall be as defined in clause 8.3.3.1.1. The destination of the Imm-ACK frame shall be the same as the source address of the node requested the acknowledgement. The receiver shall not acknowledge incoming frames if not requested by the transmitter.

The Imm-ACK frame shall be transmitted T_{AIFG} after the end of the received PHY frame that has requested the Imm-ACK. If the MAC frame is received with errors as determined by FCS, the receiver may send a NACK to the originator only if an acknowledgment is requested and the destination address of the frame matches the receiver's device address. However, if the receiver can determine that the error is caused by collision, it may avoid sending a NACK (no response) to invoke a collision state on transmitting station. The transmitting station shall infer a collision from the absence of any response to a transmission when a response is expected. In this case the transmitting station shall attempt a retransmission after T_{CBIFG} interval.

The node that shall transmit Imm-ACK is not allowed to receive frames after the end of the received PHY frame until the Imm-ACK is sent.

All nodes in the domain shall refrain from transmission when Imm-ACK is expected and within T_{IFG_MIN} after the transmission of Imm-ACK frame transmission is complete. The refrain time shall be computed based on the PHY frame type and the status of the ACK_REQ bit in the PFH. If the sender indicates the PHY frame transmission interval using FL field of the PFH (PHY frame Type 2 or Type 4, see clause 7.2.3.2 of [ITU-T G.9955]), this interval shall include maximum value of T_{AIFG} and Imm-ACK transmission time, if Imm-ACK is requested by the sender. The sender of a frame shall ensure that the overall duration of the frame with the following Imm-ACK frame is in the range covered by the FL.

NOTE – The range of FL, as defined in [ITU-T G.9955], is capable to accommodate overall frame duration up to 360 ms, which is usually longer that the allowed time-on-wire defined by the parameter MaxMediumTime (see clause 8.1.4).

The timing of the Imm-ACK frame transmission is described in Figure 8-16. The values of T_{AIFG} and T_{IFG_MIN} shall be as defined in clause 8.1.4.

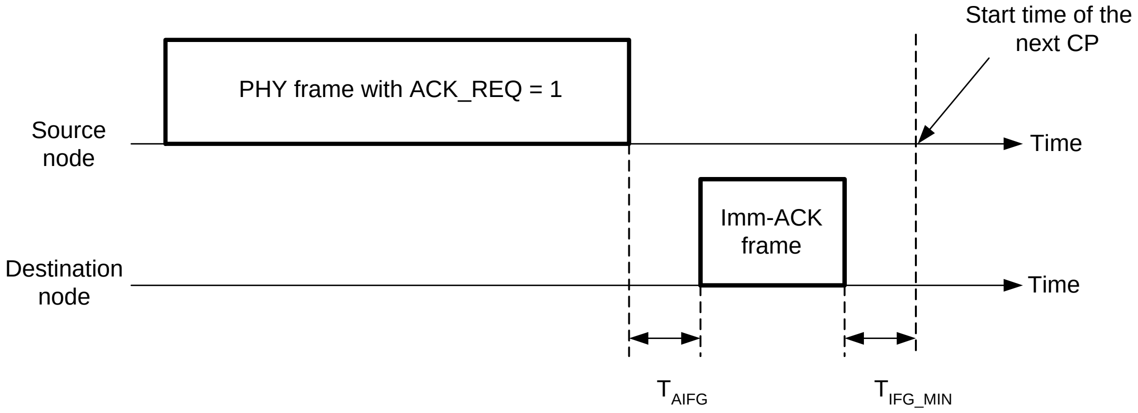


Figure 8-16 – Timing of the Imm-ACK frame

An MS-MPDU capable and SS-MPDU capable receivers shall respond using Imm-ACK frame of the corresponding type (see clause 8.3.3.1.1).

An MS-MPDU receiver that has no sufficient resources to provide a selective ACK (in case the ACK_TYPE = 1 in the received frame) may send an ACK for the whole MPDU (as per request ACK_TYPE = 0) that informs the source that its request for selective ACK cannot be accomplished. Each node shall be always capable to provide ACK and NACK for the whole MPDU (as required per ACK_TYPE = 0).

If a receiver that does not support decoding of MS-MPDU frames and receives an MS-MPDU frame with no errors in the MPH, it shall respond by sending an Imm-ACK as for SS-MPDU (Table 8-15) with an Unable-to-Comply (UTC) bit set, indicating that it does not support MS-MPDU.

If a node receives a frame that carries an MPDU that was already received correctly or carries an MPDU containing LPDUs that were already received correctly in the previously transmitted frames, the receiver shall be capable to filter out and drop this duplicate MPDU or all these duplicate LPDUs. If these duplicates are received in error, the receiver shall acknowledge them as received correctly (to avoid unnecessary retransmissions).

8.3.3.1.1 Format of the Imm-ACK frame

The Imm-ACK frame shall have a format of a PHY frame Type 3 with the fields defined in Table 7-12 of [ITU-T G.9955]. The reserved field of the PHY frame shall be used for the acknowledgement data. Four sub-types of Type 3 frames are defined, depending on the 2-bit Frame Type field (bits [3:2] of the PFH):

00 – Imm-ACK of a MS-MPDU capable node

01 – Imm-ACK of a SS-MPDU capable node

10 – Extended Imm-ACK frame

11 – Reserved

The Imm-ACK frame generated by an MS-MPDU capable node shall be as indicated in Table 8-14.

Table 8-14 – Imm-ACK data field in PFH (Frame Type 3)

Field	Bits	Description
Frame type	[3:2]	00
Reserved	[5:4]	Reserved by ITU-T (for flow control, see clause 8.5.7)
DNID	[11:6]	The 6 LSB of the NODE_ID of the node that requested the acknowledgement (Note 2)
LFSN	[19-12]	Sequence number of the LLC frame carried by the MPDU that is acknowledged by this Imm-ACK frame. Shall use the same format as defined in MPH of the MPDU that is acknowledged.
ACK TYPE	[20]	Type of the ACK data: 0 indicates per-MPDU acknowledgement, 1 indicates selective acknowledgement. In case a node is requested to provide selective acknowledgement but not capable, this bit shall be set to 0
ACK/NACK	[21]	If ACK TYPE = 0, indicates per-MPDU acknowledgement (see Table 8-13 for the definition of the acknowledgement types) and shall be set to 1 for ACK (all LPDUs and MPCS of the MPDU are received with no errors) and to 0 for NACK (error in at least one of the LPDUs). If ACK TYPE = 1, indicates selective acknowledgement on the first LPDU of the received MPDU, and shall be set to 1 if LPDU is received correctly and 0 otherwise.

Table 8-14 – Imm-ACK data field in PFH (Frame Type 3)

Field	Bits	Description
ACKI/TP-PR	[28:22]	ACKI operation Bit map of errored LPDUs (1= LPDU received correctly, 0 = LPDU received in error). The first bit of the bit map represents the second LPDU of the received MPDU (the first LPDU is the one that follows the MPH and its status is represented by the ACK/NACK field). Valid only if ACK TYPE = 1. In case ACK TYPE = 0 and the received value of TP-REQ ≠ 11, bits [28:22] shall be set to 0 and ignored by the receiver of the Imm-ACK frame. TP-PR operation: For CENELEC and FCC1 bandplans only, if the received value TP-REQ = 11, this field shall carry the TP-PR field value (regardless of ACK-TYPE assignment), using the format defined in Table 8-17 (Note 3)
LQI	[29]	Link quality indicator. Shall be set to 0 if the receiver does not recommend changes in transmission parameters and set to 1 otherwise. If ACK and all ACKI bits are set to 1, the LQI = 1 represents that the received SNR is substantially higher than the one required for used transmission parameters (increase of bit rate is possible). If ACK or one of ACKI bits is set to 0, the LQI = 1 represents that the received SNR is substantially lower than the one required for used transmission parameters (more redundancy shall be used to keep the link reliable). The evaluation of SNR for LQI setting is vendor discretionary.
TP-PR	[36:30]	Partial TP recommendation of the receiver, represented in a format defined in Table 8-17. Applicable for FCC and FCC2 bandplans only.
Reserved	[61:37]	Reserved by ITU-T (applicable to FCC and FCC2 bandplans only)

NOTE 1 – All bits reserved by ITU-T shall be set to 0 by the transmitter and ignored by the receiver.

NOTE 2 – The destination node is identified by the 6 LSB bits of the NODE_ID and the LFSN of the LLC frame to which belong the segments carried by the acknowledged LPDUs.

NOTE 3 – For CENELEC and FCC1 bandplans, in case the received value of TP-REQ=11, the ACK/NACK indicates the status of the whole MPDU (the transmitter shall set ACK_TYPE bit of the MPH to 0).

The format of an Imm-ACK frame generated by a node that is only capable of receiving SS-MPDUs (a node of low complexity profile) shall be as defined in Table 8-15.

Table 8-15 – Imm-ACK data field in PFH (Frame Type 3)

Field	Bits	Description
Frame type	[3:2]	01
Reserved	[5:4]	Reserved by ITU-T (for flow control, see clause 8.5.7)
DNID	[9:6]	The 6 LSB of the NODE_ID of the node that requested the acknowledgement (Note 2)
LFSN	[19-12]	Sequence number of the LLC frame carried by the MPDU that is acknowledged by this Imm-ACK frame. Shall use the same format as defined in MPH of the MPDU that is acknowledged.
UTC	[20]	Unable to comply – shall be set to 1 if the receiver is incapable to process the incoming MS-MPDU frame addressed to it and 0 otherwise
ACK/NACK	[21]	Shall be set to 1 for ACK (the MPDU received with no errors) and to 0 for NACK (MPDU received in error). Valid only if UTC bit is 0.
TP-PR	[25:22]	Partial TP recommendation of the receiver, represented in a format defined in Table 8-17 (applicable for all bandplans).
Reserved	[28:26]	Reserved by ITU-T
LQI	[29]	Link quality indicator. Shall be set to 0 if the receiver does not recommend changes in transmission parameters and set to 1 otherwise. If ACK/NACK bit is set to 1, the LQI = 1 represents that the received SNR is substantially higher than the one required for used transmission parameters (increase of bit rate is possible). If ACK/NACK bit is set to 0, the LQI = 1 represents that the received SNR is substantially lower than the one required for used transmission parameters (more redundancy shall be used to keep the link reliable). The evaluation of SNR for LQI setting is vendor discretionary.
Reserved	[61:30]	Reserved by ITU-T (only valid for FCC and FCC2 bandplans)
NOTE 1 – All bits reserved by ITU-T shall be set to 0 by the transmitter and ignored by the receiver. NOTE 2 – The destination node is identifies by 6 LSB of the NODE_ID and the LFSN of the LLC frame to which belong the segments carried by the acknowledged LPDUs.		

An Extended Imm-ACK frame contains regular ACK fields and extension fields attached after the regular ACK fields. If a node is incapable of providing the requested data for one or more extension fields, it shall respond with an unable-to-comply-extension (UTCE) bit set in the extension field of the Extended Imm-ACK. The ACK Type extension field indicates whether Extended Imm-ACK is SS-MPDU or MS-MPDU type. Format and content of other extension fields is for further study.

Table 8-16 – Extended Imm-ACK – extension data fields

Field	Bits	Description
ACK Type	[0]	0 – for MS-MPDU 1 – for SS-MPDU
UTCE	[1]	Set to 1 if unable to support the extension field, and 0 otherwise
Reserved	Reserved	For further study
NOTE – All bits reserved by ITU-T shall be set to 0 by the transmitter and ignored by the receiver.		

8.3.3.1.1.1 TP-PR field

The TP-PR field indicates the partial TP response, which is the receiver recommendation on particular set of TP. It shall be filled by the receiver in response to the request of the transmitter for a partial TP report (TP-REQ = 11, see clause 8.1.3.1.1). If the receiver is incapable to provide a recommendation for one or more parameters in the Imm-ACK associated with the frame where request was sent, it shall set the corresponding TP-PR fields to "No change" indication (see Table 8-17) and prepare to provide the report in the Imm-ACK associated with the next frame received from the same source. If no request for partial TP report from the transmitter was sent, the receiver shall put "No change" into all TP-RP fields, if applicable.

For CENELEC and FCC1 bandplans, the transmitter that requests a partial TP report for an MS-MPDU shall set ACK_TYPE = 0 (shall not request a selective ACK). For FCC and FCC2 bandplans, the transmitter of MS-MPDU may request both selective ACK (ACK_TYPE = 1) and partial TP report.

The coding of the TP-PR fields shall be as defined in Table 8-17.

Table 8-17 – Format of the TP-PR field

Field	Bits			Note
	MS-MPDU FCC, FCC2	**MS-MPDU CENELEC, FCC1**	**SS-MPDU All bandplans**	
RS codeword size (RSCW)	[30]	[22]	[22]	Same as RSCW field described in clause 7.2.3.2.4 of [ITU-T G.9955]
CC Rate (CCR)	[31]	[23]	N/A	Same as CCR field described in clause 7.2.3.2.5 of [ITU-T G.9955]
Change the number of Repetitions (REP)	[33:32]	[25:24]	[24:23]	00 – no change 01 – one step down 10 – one step up 11 – Reserved by ITU-T
Interleaving Mode (INTM)	[34]	[26]	[25]	Same as INTM field described in clause 7.2.3.2.7 of [ITU-T G.9955]
Modulation (MOD)	[36:35]	[28:27]	N/A	Same as MOD field described in clause 7.2.3.2.8 of [ITU-T G.9955]
NOTE – The coding of all TP-PR fields, except REP, shall be interpreted as: – if the receiver has got no recommendation (cannot estimate the parameter) it shall recommend the same value as it was assigned by the transmitter ("No change" indication); – if the receiver has got a recommendation to change the value, it shall put in the recommended value.				

8.3.3.2 Acknowledgement for multicast and broadcast frames

The source of multicast and broadcast frames shall not request acknowledgement (ACK_REQ bit of the PFH shall be set to 0).

8.3.3.3 Retransmission protocol for unicast frames

8.3.3.3.1 Requirements for the transmitter

If acknowledgement is required for an LLC frame, the transmitter shall request acknowledgement for each MPDU carrying segment(s) of this LLC frame. For different MPDUs, transmitter may request different type of ACK (per MPDU or per LPDU, if available). Upon reception of an Imm-ACK frame indicating that a particular MPDU or its tributary LPDUs were received in error, the transmitter shall suspend transmission of other MPDUs of the same destination, except MPDUs of MA priority 3, and retransmit the indicated erroneous MPDU or its tributary LPDU using the rules described in clause 8.1.2.3. No MPDU of an LLC frame is allowed to be transmitted until all LPDUs of previously transmitted LLC frame are cleared (received error free) or timed out. If an MPDU of MA priority 3 is ready for transmission, it shall be transmitted at the first opportunity, interrupting the transmission of a data or a management frame.

In case a node that transmitted a frame receives in response an Imm-ACK frame indicating that the receiver does not support MS-MPDU mode, the node shall retransmit the LPDUs of this MS-MPDU using SS-MPDU mode.

In case acknowledgements on all LPDUs of a particular LLC frame are not received by the transmitter (no Imm-ACK frame received or a persistent negative acknowledgement is received) after a total of *MaxRetransmissions* (see clause 8.1.4) attempts, the transmitter shall discard the LLC frame and notify upper layers that transmission of the LLC frame failed using the A_MGMT.IND primitive FrameTxFailure (see clause 8.4.3.4.3). Same notification shall be sent if transmitter does not get all the LPDUs of an LLC frame acknowledged during *MaxAckRxTime* (see clause 8.1.4) timeframe. The transmitter shall adjust this timeframe in case transmission of LLC frame is interrupted by a priority 3 frame by freezing the *MaxAckRxTime* counter of the interrupted frame till reception of the interrupting frame is complete.

NOTE – The upper layer may further decide to transmit the discarded LLC frame again. This functionality is beyond the scope of the Recommendation.

8.3.3.3.2 Requirements for the receiver

The receiver shall first identify whether the received frame with ACK_REQ = 1 is valid for acknowledgement by checking LPH and MPH. If MPH is received in error (MHCS does not fit or any of the MPH fields are invalid) or at least one of the address fields indicted in the MPH (DNID, SNID, DDID, SDID) does not fit, or segment sequence number in LPH is inconsistent, the value of MDPU type is invalid, the receiver shall not send Imm-ACK frame and shall drop the received frame. In the received MPDUs that are valid for acknowledgement the receiver shall verify the segmentation method of the received MPDU (i.e., whether this is an MS-MPDU or an SS-MPDU).

The receiver shall apply the following criteria to identify errored MPDUs and LPDUs. For a MS-MPDU the receiver shall identify errored LPDUs by checking their LPCS; the MPDU is considered in error if at least one LPCS is received in error. For an SS-MPDU, the receiver shall identify errored MPDUs by checking MPCS. If a receiver is only capable to process SS-MPDU, it shall respond on the received MS-MPDU by ACK frame type 01 with the UTC bit set to 1 or, if capable to receive an MS-MPDU, with an ACK/NACK as defined in Table 8-14.

If the received frame has an inconsistent sequence number and the PRI field of the MPH is set to 11_b, the receiver shall consider this frame as a MA priority 3 interrupting frame.

NOTE – To manage the receive buffers, the receiver may perform a time-limited reception of each LLC frame based on *MaxRetransmissions* and *MaxAckRxTime* parameters of the transmitter (see clause 8.3.3.3.1, clause 8.1.4). In particular, if during *MaxAckRxTime* not all of the LLC frame segments are received, the receiver may discard all previously received segments of this LLC frame. The count of *MaxAckRxTime* in this case shall start from the time the first MPDU carrying segments of an LLC frame was received.

8.3.4 Beacon operation

Beacons are special management frames used by the domain master to control and maintain the domain. The format of the beacon frame is defined in clause 8.3.4.4.

A domain shall be initialized in one of three beacon modes:

- Synchronous beacon mode (see clause 8.3.4.1).
- Asynchronous beacon mode (see clause 8.3.4.2).
- Beaconless mode (see clause 8.3.4.3).

During the domain operation, the domain master can change the beacon mode based on the corresponding request from the upper layer management (management primitive BeaconMode at A_MGMT interface defined in clause 8.4.3.4.1). The procedure of changing the beacon mode is for further study.

The following rules related to the beacon modes shall be respected:

- Reception and processing of received beacons frames shall be supported by all nodes.
- Support of synchronous beacon mode is optional.
- In synchronous beacon mode, nodes that support only asynchronous beacon mode shall respond to the asynchronous portion of the synchronous beacon (i.e., may ignore the BCN_SyncSchdle.req message – see clause 8.3.4.4).

The particular beacon mode used in the domain is indicated in each transmitted beacon frame (in the BM field of the BCN_DI.req message, Table 8-19). Each beacon frame also includes a serial number (in BCN_ID field of the BCN_DI.req message), in the range between 0 and 255. Whenever a new beacon frame is generated by the domain master, the BCN_ID field shall be increased by 1 and shall wrap back to 0x00 after reaching 0xFF.

The beacon frame shall always be sent using robust communication mode (RCM) as specified in [ITU-T G.9955]; values of the transmission parameters are defined in clause 8.3.4.4.

Only domain master is allowed to transmit beacons, and it is a responsibility of the domain master to provide beacons for all nodes of the domain. To reach hidden nodes, the domain master may assign beacon re-generation proxies (BPR).

8.3.4.1 Synchronous beacon mode

In synchronous beacon mode, transmission is organized by beacon cycles of fixed duration. Each cycle starts with a beacon frame (see clause 8.3.4.4) and is followed by time slots reserved for relayed beacons (R-beacons), followed by the time period assigned for frame transmission, Figure 8-17.

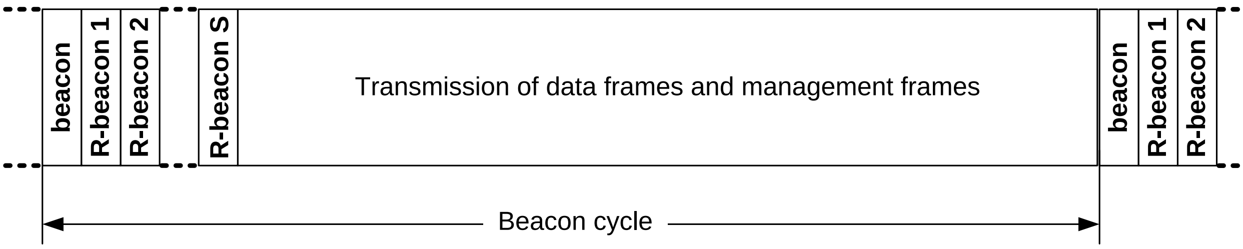


Figure 8-17 – Beacon cycle structure in synchronous beacon mode

The domain master shall transmit the beacon in the beacon slot only. By sending the next beacon, the domain master determines the duration of the beacon cycle. The duration of beacon cycle shall not exceed the duration that corresponds to the maximum value of the BCD field of the BCN_SyncSchdle.req message (see Table 8-21).

All assigned BPRs shall be capable of supporting synchronous beacon mode. The domain master shall also assign a number of slots in the beacon cycle for R-beacon transmissions following the beacon slot (see Figure 8-17). The number of R-beacon slots in a beacon cycle shall be limited to $S \leq 8$. If more than S slots for R-beacons are required, the domain master may assign a particular set of BPRs for each beacon cycle, ensuring that each BPR transmits R-beacon at least once during the period *BCNMaxT* which is an LLC_MGMT primitive (see clause 8.4.3.2) determined by the domain master and communicated to the BPR nodes in the corresponding field of the BCN_DI.req message (see Table 8-20).

NOTE – One simple way of BPR assignment can be round robin: a particular BPR may be assigned to transmit R-beacon #N at every m-th beacon cycle starting from the beacon cycle #D.

To facilitate an order in R-beacon generation, every beacon:

- Carries the serial number of the beacon cycle on the BCID field on the BCN_SyncSchdle.req message as described in Table 8-21. The serial number of the beacon is indicated in each of the corresponding R-beacons; this field shall be increased by 1 for each new beacon cycle, and shall wrap back to 0x00 after reaching 0xFF.
- Indicates the set of BPRs assigned to transmit R-beacon in the particular beacon cycle on the BRP_N and BPRi fields on the BCN_SyncSchdle.req message (see Table 8-21).

The duration of R-beacon slots and other timing and medium access details concerning synchronous beacons are for further study.

All end-nodes of the domain shall not transmit during the beacon and any of R-beacon time periods; BPRs are allowed to transmit R-beacons only in a particular R-beacon period assigned by the domain master and communicated in the corresponding BPR$_i$ field of the beacon.

8.3.4.2 Asynchronous beacon mode

In asynchronous beacon mode no particular beacon cycle boundaries are defined. Beacon frames are transmitted at the discretion of the domain master, and R-beacons are transmitted on discretion of the BPRs assigned by the domain master. In asynchronous beacon mode, the assigned BPR nodes may not support synchronous beacon mode.

Both beacons and R-beacons shall be transmitted with MA priority 3.

A beacon shall be transmitted at least once during a predefined time period specified on the *BCNMaxT* field of the BCN_DI.req message (see Table 8-20). The timing and other rules of R-beacon transmission are for further study.

8.3.4.3 Beaconless mode

In beaconless mode beacon frames shall not be sent (all domain controls are provided as a part of the corresponding L3 protocols, by L3 in-band management messages). More details on beaconless mode are for further study.

8.3.4.4 Beacon frame format

The beacon frame includes one or more beacon management messages (message category 001, as described in clause 8.4.2). No other management messages are allowed to be included into a beacon frame.

A beacon frame shall include the BCN_DI.req management message defined in Table 8-20, as the first management message of the beacon frame. More beacon management messages may follow. In synchronous beacon mode, the beacon frame shall also include the BCN_SyncSchdle.req message defined in Table 8-21.

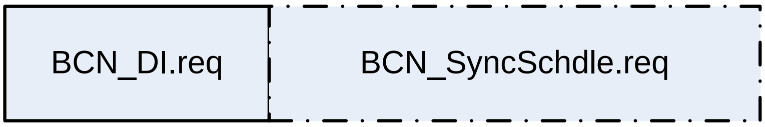


Figure 8-18 – Example of beacon frame format including BCN_DI.req message and one additional beacon message (BCN_SyncSchedle.req)

8.3.4.4.1 Transmission of beacon frames

The following setting and transmission parameters shall be set for transmission of beacon frames.

The LFH fields of an LLC frame that carries LCDU of a beacon frame shall be set as follows:

- LLCFT = 1 (LCDU)
- MHI = 1 (mesh header is present)
- CCMPI = 0 (not secure).

The mesh header fields shall be set as follows:

- RELI = 1 (relaying extension are present)
- ADM = 00 (short addressing mode for both source and destination node)
- LPRI = 3
- SRI = 0 (no source routing)
- SA = Node ID of the domain master
- DA = 0xFFFF (BROADCAST_ID)
- HopsLft = maximum number of hops to relay the beacon (equal to S for synchronous beacon).

The MPH fields of each MPDU comprising LPDUs of a beacon frame shall be set as follows:

- MPDU_TYPE = 000 (SS-MPDU).
- ACK_TYPE = 0.
- SNID = NODE_ID of the transmitter (domain master or BPR).
- DNID = 0xFFFF (BROADCAST_ID).
- SDID = DDID = DOMAIN_ID.

The transmission parameters of the beacon frames shall be the following:

- code rate = 1/2
- number of repetitions: 4, 6, or 12
- robust header and robust preamble as per clause 7.3.4 and clause 7.4.5.1 of [ITU-T G.9955], respectively.

Other transmission parameters are vendor discretionary, but shall be limited to the values supported by nodes of all profiles.

8.3.4.4.2 Transmission of R-beacon frames

In asynchronous beacon mode, the R-beacon frames shall use the same format, settings and transmission parameters as defined in clause 8.3.4.4.1. Exceptions and additional details related to synchronous beacon mode are for further study.

8.4 Management plane

8.4.1 Management communications

Two types of management communications are defined: internal and in-band (external). Internal management messages are carried by LCDUs and are exchanged between the peer management entities of the nodes in the domain via the LLC_MGMT SAP and A_MGMT interface (see clause 8.4.3.4.1, clause 8.4.3.4.3). Management messages intended for in-band communications, are carried by APDUs and exchanged between the peer management entities of the nodes via the A_DATA interface (see Annex I).

Both types of management communications shall use the same management message format presented in Figure 8-19. Figure 8-19 also shows the mapping of a management message into an LCDU and APDU: the APDU and LCDU payload may contain one or more concatenated management messages (MM). The FCS shall be 32-bit long and computed as defined in 8.1.2.1.3, and shall not be included when MIC is used in the LLC frame encapsulating the LCDU or the

APDU. All MM shall be encapsulated so that their first byte (octet 0 of the MMH) is transmitted first and the first byte of MM-1 is the first byte of the LCDU or the APDU payload, respectively. Further, encapsulation of the APDU or LCDU carrying a management message into an LLC frame shall be as shown in Figure 8-5.

NOTE – The format of APDU depends on the used L3 protocol and is defined in Annex I. For the case of IPv6, see Figure I.1.

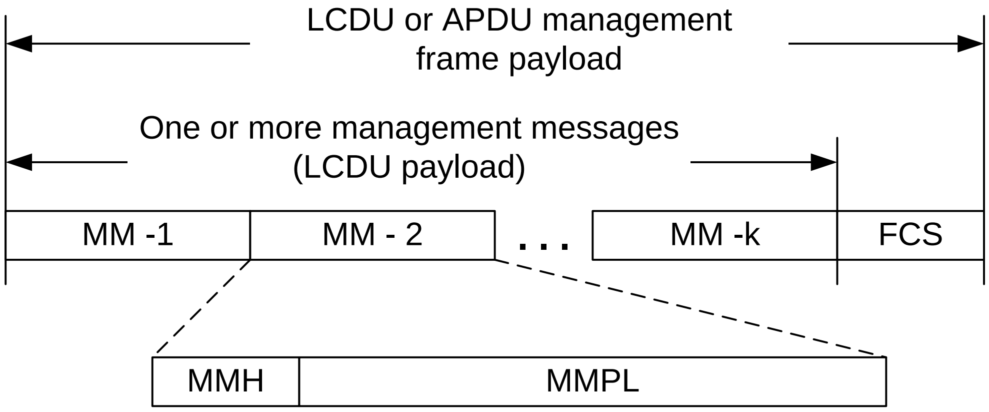

LCDU or APDU management frame payload
One or more management messages (LCDU payload)
MM -1
MM - 2
. . .
MM -k
FCS
MMH
MMPL

Figure 8-19 – Format of a management message and its mapping into LCDU and APDU

8.4.1.1 Management message format

All management messages shall be formatted as shown in Figure 8-19, and shall comprise a Management Message Header (MMH) and a Management Message Parameter List (MMPL).

The MMH defines the type, the length, the frame sequence number, and parameters related to message segmentation. The MMH also indicated whether the message is the last one mapped into an LCDU/APDU or it is followed by another management message. The type of the message is identified by an OPCODE associated with a particular management function, as presented in Table 8-19. The MMPL includes a list of parameters associated with a particular management function associated with the management message. The format of the MMH shall be as shown in Table 8-18.

The format of any MMPL may be revised in future versions of this Recommendation by appending additional fields. Furthermore, fields may be defined using bits that are currently indicated as reserved for ITU-T. Nodes shall indicate the version of the Recommendation that they support during the registration (see clause 8.5.2) and in their topology reports. Nodes shall be able to parse any MMPL (the length of the MMPL is specified in the MMH) by ignoring the fields associated with later versions of the Recommendation (which legacy nodes cannot interpret).

The maximum size of the management message carried by an APDU is determined by the particular AE protocol and thus always complies with the maximum APDU size. The maximum size of the management data carried by the LCDU is limited by the size of the LCDU to 992 bytes (see clause 8.1.2.1.3 and Table 8-15). If the amount of management data to be sent exceeds 992 bytes, segmentation shall be used, as defined in Table 8-15, providing a MMPL size that is less than 992 bytes.

NOTE – If channel conditions require to use shorter frames, this is provided by the segmentation mechanism of the LLC frame, see clause 8.1.2.2.

Table 8-18 – Format of MMH

	Content	Octet	Bits	Description
MMH	MMPL length (LG)	0 – 2	[11:0]	Length of the MMPL segment in octets, encoded as a 12-bit unsigned integer. The length value shall not exceed 992 bytes.
	OPCODE		[23:12]	12-bit OPCODE, indicates message type (Note 1)
	Reserved	3	[7:0]	Reserved by ITU-T
	Number of segments	4	[3:0]	Number of segments minus 1, represented as an unsigned integer between 0 and F_{16}. It shall be set to 00_{16} if the message is not segmented
	Segment number		[7:4]	Segment number, represented as an unsigned integer between 0_{16} and F_{16}; set to 0_{16} for the first segment and if message is not segmented (Note 2)
	Sequence number	5-6	[15:0]	Sequence number of the message in a format of 16-bit unsigned integer (Note 3)
	Next header	7	[0]	Shall be set to 0 if this message is the last one mapped in the APDU/LCDU, and set to 1 otherwise.
	Reserved	7	[7:1]	Reserved by ITU-T
MMPL	Message Parameters	8 to (LG+7)	[(8×LG-1):0]	Depends on the OPCODE, see Table 8-19
	Pad	> (LG+7)	[7:0]	Shall be filled by zeros (Note 4)

NOTE 1 – The OPCODES are defined in Table 8-19.

NOTE 2 – Segmentation shall be in the ascending order of octets, i.e., starting from octet #0. Segmentation shall not be applied segmented if the required MMPL is shorter than 992 bytes.

NOTE 3 – The sequence number identifies the relative time the message was sent: bigger sequence number corresponds to later transmission time. Actions associated with the sequence number of the received message depend on the OPCODE. All segments of the same message shall have the same sequence number. The value wraps around after reaching 2^{16}.

NOTE 4 – Zero bytes can be added at the end of the MMPL to reach the minimum size of the payload (if required) or for any other purpose.

8.4.1.2 Management message OPCODEs

OPCODEs shall be formatted as 12-bit unsigned integers, where the 8 MSBs indicate the category of the message and the 4 LSBs indicate the message ID in a category. Valid values of OPCODEs are presented in Table 8-19. Management messages are categorized by their associated protocol or procedure. If the number of messages for a particular protocol or procedure is more than 16, two or more 8-bit values can be assigned for the same category.

8.4.2 Management messages

A complete list of management messages used in the Recommendation is presented in Table 8-18. The MMPL content of each message is defined in the following clauses. Four types of messages are defined: Request (.req), Confirmation (.cnf), Indication (.ind), and Respond (.rsp), according to messaging convention in clause 6.2.

Table 8-19 – OPCODEs of management messages

Category	Message name	OPCODE (hex)	Description	MMPL Reference
Beacon (00X)	BCN_DI.req	000	Domain information	Table 8-20
	BCN_SyncSchdle.req	001	Synchronous beacon scheduling parameters	Table 8-21
Channel Estimation (03X)	Transmission Parameters Request (TP.req)	031	Request for transmission parameters	N/A
	Transmission Parameters (TP.ind)	032	Transmission parameters recommendation	Table 8-20
Request for Information (04X)	RFI_BcnInfo.req	000	Request for beacon information	Table 8-22
	RFI_BcnInfo.ind	001	Beacon information indication	Table 8-23

Other messages and message categories are for further study.

8.4.2.1 Beacon messages

The MMPL of the beacon messages are described in Tables 8-17 to 8-18.

8.4.2.1.1 Domain information message

The BCN_DL.req message is intended to distribute domain information to the nodes of the domain. The MMPL field of the BCN_DL.req message is presented in Table 8-20.

Table 8-20 – Description of the MMPL of the BCN_DI.req message

Field	Octet	Bits	Description
BCN_ID	0	[7:0]	Beacon ID, formatted as a non-signed integer
PFHM	1	[0]	Domain-wise PFH mode: 0 – normal PFH shall be used for transmission in the domain 1 – robust PFH shall be used for transmission in the domain
PRMBLM	1	[1]	Domain-wise preamble mode: 0 – normal preamble shall be used for transmission in the domain 1 – robust preamble shall be used for transmission in the domain
L2R	1	[2]	L2 Relaying support: 0 – no L2 relaying allowed in the domain 1 – L2 relaying is allowed in the domain
BM	1	[4:3]	Beacon mode: 00 – beaconless mode 01 – synchronous mode 10 – asynchronous mode 11 – reserved by ITU-T

Table 8-20 – Description of the MMPL of the BCN_DI.req message

Field	Octet	Bits	Description
SL	1	[6:5]	Security level: 00 – none (encryption disabled, no MIC) 01 – normal security (encryption enabled, 4 bytes MIC) 10 – high security (encryption enabled, 16 bytes MIC) 11 – reserved by ITU-T
Reserved	1	[7]	Reserved by ITU-T
REC	2	[7:0]	Version of the recommendation supported by the domain master. Set to 0x00.
DMID	3-4	[15:0]	Node ID of domain master
DMExtAddr	5-20	[7:0]	EUI-64 address of the domain master
BDMID	21-22	[15:0]	Node ID of the backup domain master (0xFFFF if not assigned)
DNI	23-24	[15:0]	Domain name identifier
BPL	25	[7:0]	Lowest tone index of the bandplan
BPU	26	[7:0]	Upmost tone index of the bandplan
BCNMaxT	27-30	[31:0]	Maximum time allowed in the domain without transmission of a beacon in resolution of 100 ms
RMSC_N	31	[7:0]	Number of RMSC index fields following this field
RMSC1	32	[7:0]	1st index of the RMSC set
…		…	…
RMSCn	31+n	[7:0]	nth index of the RMSC set
NOTE – All fields reserved by ITU-T shall be set to 0 by the transmitter and ignored by the receiver.			

8.4.2.1.2 Domain synchronization schedule message

The BCN_SyncSchedule.req message is intended to distribute the node transmission schedule and shall be used only if the domain operates in synchronized beacon mode. The MMPL field of the BCN_SyncSchedule.req message is presented in Table 8-21.

Table 8-21 – Description of the MMPL of the BCN_SyncSchdle.req message

Field	Octets	Bits	Description
BCD	0-3	[31:0]	Duration of beacon cycle in resolution of 100 ms
BCID	4	[7:0]	Synchronous beacon cycle ID
BPR_N	5	[7:0]	Number *n* of beacon regeneration proxy (BPR) slots in this beacon cycle (the *n* fields following this field define the NODE_ID of each BPR)
BPR1	6-7	[15:0]	Node ID of BPR1
…	…	…	…
BPRn	6+2xn – 7+2xn	[15:0]	Node ID of BPRn

8.4.2.2 Request for information (RFI) messages

8.4.2.2.1 Request for information message

The MMPL of an RFI__BcnInfo.req message is described in Table 8-22.

Table 8-22 – Description of the MMPL of the RFI_BcnInfo.req message

Field	Octets	Bits	Description
OPC-Req	0-1	[11:0]	Bit map of the opcodes of the category 00X (Beacon) management messages that are requested to be transmitted in response to the RFI_BcnInfo.Req message. The LSB of the bitmap represents opcode 000. If bit i is set to 1, the RFI_BcnInfo.Ind management message may include the management message with opcode 00i.
Reserved	1	[15:12]	Reserved by ITU-T. Shall be set to 0 by the transmitter and shall be ignored by the receiver.

8.4.2.2.2 Response on beacon information message

The MMPL of an RFI_BcnInfo.ind message is described in Table 8-23.

Table 8-23 – Description of the MMPL of the RFI_BcnInfo.ind message

Field	Octets	Bits	Description (Reference)
OPC-Resp	0-1	[11:0]	Bit map of the category 00X (Beacon) opcodes of the management messages that are transmitted in response to the RFI_BcnInfo.Req message. The LSB of the bitmap represents opcode 000. If bit i is set to 1, the RFI_BcnInfo.Ind management message shall include the management message with opcode 00i. If bit i is set to 0, the RFI_BcnInfo.Ind management message shall not include the management message with opcode 00i. (Note 1)
Beacon message 1	Variable	Variable	Beacon management message with opcode corresponding to the first LSB that is set to 1 in the OPC-Resp field
…			
Beacon message n	Variable	Variable	Beacon management message with opcode corresponding to the first MSB that is set to 1 in the OPC-Resp field
NOTE 1 – When bit i was set to 1 in the RFI_BcnInfo.Req and the same bit i is set to 0 in the RFI_BcnInfo.Ind sent in response, it indicates that the responding node is unable to comply with the request for information on opcode 00i.			

8.4.2.3 Channel estimation messages

8.4.2.3.1 Transmission parameter request message (TP.req)

MMPL field of TP.req message is empty.

8.4.2.3.2 Transmission parameter response message (TP.ind)

The MMPL filed of the TP.ind message shall be as presented in Table 8-24.

Table 8-24 – MMPL field of TP.ind message format

Field	Bits		Description (Reference)
	CENELEC, FCC-1	FCC, FCC-2	
Tone mask (TM)	[7:0]	[39:0]	TM recommendation, where the TM field is described in clause 7.2.3.2.3 of [ITU-T G.9955]
RS code word size (RSCW)	[8]	[40]	RSCW recommendation, where the RSCW field is described in clause 7.2.3.2.4 of [ITU-T G.9955]
CC Rate (CCR)	[9]	[41]	CCR recommendation, where the CCR field is described in clause 7.2.3.2.5 of [ITU-T G.9955]
Repetitions (REP)	[12:10]	[44:42]	REP recommendation, where the REP field is described in clause 7.2.3.2.6 of [ITU-T G.9955]
Interleaving Mode (INTM)	[13]	[45]	INTM recommendation, where the INTM field is described in clause 7.2.3.2.7 of [ITU-T G.9955]
Modulation (MOD)	[15:14]	[47:46]	MOD recommendation, where the MOD field is described in clause 7.2.3.2.8 of [ITU-T G.9955]
Req-LFSN	[23:16]	[55:48]	The value of the LFSN field from the request to which the response corresponds

8.4.3 DLL management and control primitives

The DLL management and control primitives related to all sub-layers of the DLL (APC_MGMT, LLC_MGMT, and MAC_MGMT) are defined in clause 5.2.2.3 and shown in Figure 8.1. This clause describes these primitives in detail, as presented in Table 8-25.

Table 8-25 – DLL management and control primitives

Category	Primitive	Description
DLL_MGMT.REQ	APC_MGMT.REQ	Requests from DLL management entity to APC, LLC, and MAC to apply particular parameters for the transmit frame and prompts for parameters of the received frame
	LLC_MGMT.REQ	
	MAC_MGMT.REQ	
	A_MGMT.REQ	Request from the upper layer management entity to the DLL management entity
DLL_MGMT.CNF	APC_MGMT.CNF	The DLL sublayers (APC, LLC, MAC) confirm to DLL management entity that the requested parameters are applied for the transmit frame
	LLC_MGMT.CNF	
	MAC_MGMT.CNF	
	A_MGMT.CNF	The DLL management entity confirms to the upper layer management that the requested parameters are applied
DLL_MGMT.IND	APC_MGMT.IND	The DLL sub-layers (PCS, PMA, PMD) report to the DLL management entity the particular parameters acquired from the transmitted and received frames
	LLC_MGMT.IND	
	MAC_MGMT.IND	
	A_MGMT.IND	The DLL management entity reports to the upper layer management the particular parameters acquired by DLL

Table 8-25 – DLL management and control primitives

Category	Primitive	Description
DLL_MGMT.RES	APC_MGMT.RES	The DLL management entity acknowledges the parameters it receives from the DLL sublayers
	LLC_MGMT.RES	
	MAC_MGMT.RES	
	A_MGMT.RES	The upper layer management acknowledges the parameters it receives from the DLL management entity

8.4.3.1 APC_MGMT primitives

The APC_MGMT primitives are defined in Annex I, clause I.1.6.2.

8.4.3.2 LLC_MGMT primitives

8.4.3.2.1 LLC_MGMT.REQ

This primitive requests the LLC to use particular parameters for frame processing. The attributes of the primitive are defined in Table 8-26.

Table 8-26 – The attributes of the LLC_MGMT.REQ primitive

Name	Type	Valid range	Description
TX-FrameType	4-bit integer	0_{16} – F_{16}	Indicates the type of the LLC frame to be transmitted (APC type, compressed/uncompressed, sourced from APC, see clause 8.1.2.1.2.1)
CCMPI	1-bit integer	0, 1	Indicates whether the frame shall be transmitted secured (1) or not (0), see clause 8.1.2.1.2.2 and Table 8-42.
RelayMode	1-bit integer	0, 1	Indicates whether the LLC relay function shall be enabled (1) or disabled (0)
LPRI	2-bit integer	00_2 – 11_2	Indicates the priority of the transmit LLC frame (sourced from APC)
DA	16-bit or 64-bit integer	See clause I.1.6.2	The final destination address of the transmit LLC frame (sourced from APC)
S_LLC	16-bit integer	0001_{16} – $05F2_{16}$	The maximum length of an LLC frame segment in bytes (see clause 8.1.2.2.1)
Frame length	16-bit integer	0001_{16} – $05F2_{16}$	Indicate the APDU length or the LCDU length in bytes of the transmit LLC frame
LCDU payload	String of bytes	See clause 8.4.1.1	The content of the management messages to transmit within the LCDU, with the format as detailed in clause 8.4.1.1
BCNMaxT	32-bit integer	0001_{16} – $FFFF_{16}$	The value of the BCNMaxT field as indicated in MMPL of the BCN_DI.req message, expressed in resolution of 100 ms.

Additional primitives are for further study.

8.4.3.2.2 LLC_MGMT.CNF

This primitive confirms to DLL entity management that the requested parameters are used by the LLC for frame processing. The attributes of the primitive are as defined in Table 8-27.

If the LLC is unable to comply with a particular attribute in the LLC_MGMT.REQ, it shall set this primitive to one, which means that the request is denied. Otherwise the value of the LLC_MGMT.CNF primitive shall be set to zero.

Table 8-27 – The attributes of the LLC_MGMT.CNF primitive

Name	Type	Valid range	Description
Status	Integer	0, 1	0: success 1: request is denied

Additional primitives are for further study.

8.4.3.2.3 LLC_MGMT.IND

This primitive provides the DLL management entity with particular parameters of the frames processed by the LLC in transmit and receive direction. The attributes of the primitive are defined in Table 8-28.

Table 8-28 – The attributes of the LLC_MGMT.IND primitive

Name	Type	Valid range	Description
RX-FrameType	4-bit integer	0_{16} – F_{16}	Indicates the type of the received LLC frame (APC type, compressed/uncompressed, sourced from APC, see clause 8.1.2.1.2.1)
LPRI	2-bit integer	00_2-11_2	Indicates the priority of the received LLC frame (if needs to be relayed)
SA	16-bit or 64-bit integer	See clause I.1.6.2	The originating source address of the received frame
LCDU payload	String of bytes	See clause 8.4.1.1	The content of the received management messages within the LCDU, with the format as detailed in clause 8.4.1.1

Additional primitives are for further study.

8.4.3.2.4 LLC_MGMT.RES

This primitive is for further study

8.4.3.3 MAC_MGMT primitives

8.4.3.3.1 MAC_MGMT.REQ

This primitive requests the MAC to use particular parameters for frame processing. The attributes of the primitive are defined in Table 8-29.

Table 8-29 – The attributes of the MAC_MGMT.REQ primitive

Name	Type	Valid range	Description
PRI	2-bit integer	$00_2 – 11_2$	Indicates the MA priority of the transmit MPDU (based on LPRI value sourced from APC)
CW_{Offset} (0)	16-bit	$0000_{16} – 03FF_{16}$	Indicates the CW offset of MA priority 0 expressed in TS (see clause 8.2.2.1)
CW_{Offset} (1)	16-bit	$0000_{16} – 03FF_{16}$	Indicates the CW offset of MA priority 1 expressed in TS (see clause 8.2.2.1)
CW_{Offset} (2)	16-bit	$0000_{16} – 03FF_{16}$	Indicates the CW offset of MA priority 2 expressed in TS (see clause 8.2.2.1)
$CW_{Init}(0)$	16-bit integer	$0000_{16} – 03FF_{16}$	Indicates the initial value of CW (see clause 8.2.2.3) for MA priority 0 expressed in TS.
$CW_{Init}(1)$	16-bit integer	$0000_{16} – 03FF_{16}$	Indicates the initial value of CW (see clause 8.2.2.3) for MA priority 1 expressed in TS.
$CW_{Init}(2)$	16-bit integer	$0000_{16} – 03FF_{16}$	Indicates the initial value of CW (see clause 8.2.2.3) for MA priority 2 0 expressed in TS
$CW_{Init}(3)$	16-bit integer	$0000_{16} – 03FF_{16}$	Indicates the initial value of CW (see clause 8.2.2.3) for MA priority 3 0 expressed in TS
γ	8-bit integer	$00_{16} – FF_{16}$	Indicates the value of γ (see clause 8.2.2.3). Valid only if MA mode = 1, where $\gamma_{max} = FF_{16}$.
A	8-bit integer	$00_{16} – FF_{16}$	Indicates the value of A (see clause 8.2.2.3) multiplied by 10 in steps of 0.1 (i.e., A can be set to 0-25.5). Valid only if MA mode = 1
B	8-bit integer	$00_{16} – FF_{16}$	Indicates the value of B (see clause 8.2.2.3). Valid only if MA mode = 1
$CW_{InActMin}$ (0)	16-bit integer	$0000_{16} – 0400_{16}$	Indicates the value of $CW_{InActMin}$ for MA priority 0 used as described in clause 8.2.2.3 expressed in TS
$CW_{InActMin}(1)$	16-bit integer	$0000_{16} – 0400_{16}$	Indicates the value of $CW_{InActMin}$ for MA priority 1 used as described in clause 8.2.2.3 expressed in TS
$CW_{InActMin}(2)$	16-bit integer	$0000_{16} – 0400_{16}$	Indicates the value of $CW_{InActMin}$ for MA priority 2 used as described in clause 8.2.2.3 expressed in TS
$CW_{InActMin}(3)$	16-bit integer	$0000_{16} – 0400_{16}$	Indicates the value of $CW_{InActMin}$ for MA priority 3 used as described in clause 8.2.2.3 expressed in TS
TCWUpdate	8-bit integer	$00_{16} – FF_{16}$	Indicates the time period for potential reduction of the CW for all priorities as described in clause 8.2.2.3, expressed in units of 10 ms.
Destination NODE_ID	16-bit integer	See clause 8.3.1.2.2	Short address of the destination node ID (DNID)
Destination DOMAIN_ID	16-bit integer	See clause 8.3.1.2.1	Destination node DOMAIN_ID (DDID)
MPDU type	1-bit integer	0, 1	Indicates the type of the transmit MPDU: 0 – SS-MPDU 1 – MS-MPDU

Table 8-29 – The attributes of the MAC_MGMT.REQ primitive

Name	Type	Valid range	Description
ACK_TYPE	1-bit integer	0, 1	The ACK type required from the receiver of the transmitted MPDU encoded as detailed in clause 8.1.3.1.1.2
TP-REQ	2-bit integer	00_2-11_2	The type of TP report required from the receiver of the transmitted MPDU encoded as detailed in clause 8.1.3.1.1.10
Virtual carrier sense	Integer	0–1024	Indicates the number of symbols in the payload of the frame sequence during which the medium will be busy (Note 2)

Physical carrier sense	Integer	0–3	Indicates the status of the medium: (physical carrier sense based on preamble detection)
			0 – IDLE;
			1 – BUSY by ITU-T G.9955 transmission;
			2 – BUSY by non- ITU-T G.9955 transmission;
			3 – BUSY due to both ITU-T G.9955 and non-ITU-T G.9955 transmission

NOTE 1 – Values of 2 and 3 are only valid if preamble-based coexistence mechanism is enabled (see [ITU-T G.9955] clause 5.1.2.3); otherwise the valid values of the primitive are 0 and 1 only. NOTE 2 – Virtual carrier sense is for ITU-T G.9955 frames only. In case of frames Type 2 and Type 4, this value is provided by the "Virtual Carrier Sense" attribute of the PMI_MGMT.IND primitive. In case of Type 1 frame, the DLL management system shall derive the number of symbols from the "PHY parameters" attribute of the PMI_MGMT.IND defined in clause 7.8.2.3.3 of [ITU-T G.9955] (the relevant PCS_MGMT,IND attributes: "MPDU size, "Payload Modulation", "Payload Repetitions", etc.)			

Additional primitives are for further study.

8.4.3.3.2 MAC_MGMT.CNF

This primitive confirms to DLL entity management that the requested parameters are used by the MAC for frame processing. The attributes of the primitive are as defined in Table 8-30.

If the MAC is unable to comply with a particular attribute in the MAC_MGMT.REQ, it shall set this primitive to one, which means that the request is denied. Otherwise the value of the MAC_MGMT.CNF primitive shall be set to zero.

Table 8-30 – The attributes of the MAC_MGMT.CNF primitive

Name	Type	Valid range	Description
Status	Integer	0, 1	0: success 1: request is denied

Additional primitives are for further study.

8.4.3.3.3 MAC_MGMT.IND

This primitive provides the DLL management entity with particular parameters of the frames processed by the MAC in transmit and receive direction. The attributes of the primitive are defined in Table 8-31.

Table 8-31 – The attributes of the MAC_MGMT.IND primitive

Name	Type	Valid range	Description
MPDU Type	1-bit integer	0, 1	Type of MPDU in the received frame: 0 – SS-MPDU 1 – MS-MPDU
ACK_TYPE	1-bit integer	0, 1	Indicates the ACK type required by the received frame: 0 – ACK per frame is required 1 – ACK per LPDU is required (see clause 8.3.3.1)
Source NODE_ID	16-bit integer	See clause 8.3.1.2.2	Short address of the source node ID (SNID)
Source DOMAIN_ID	16-bit integer	See clause 8.3.1.2.1	Source node DOMAIN_ID (SDID)
TP-REQ	2-bit integer	002-112	The type of TP report required for the receiver frame, encoded as defined in clause 8.1.3.1.1.10
NOS	6-bit integer	1-64	Number of segments comprising the LLC frame associated with the received MPDU
LSPL	1	[15:10]	Last segment pad length, in bytes
LFSN	2	[7:0]	LLC frame sequence number
PRI	13	[14:13]	MA priority of the MPDU
NEST	8-bit integer	00_{16}-FF_{16}	Indicates the received value of NEST parameter expressed in TS. See clause 8.1.3.1.1 for details

Additional primitives are for further study.

8.4.3.3.4 MAC_MGMT.RES

This primitive is for further study

8.4.3.4 A_MGMT primitives

The A_MGMT primitives that are specific for a particular type of APC are defined in I.1.6.1.

8.4.3.4.1 A_MGMT.REQ

This primitive requests the DLL or PHY management to use particular parameters for associated functions. The attributes of the primitive are defined in Table 8-32.

Table 8-32 – The attributes of the A_MGMT.REQ primitive

Name	Type	Valid range	Description
NewDomainID	16-bit integer	See Table 8-7	The value of the Domain ID asserted for the domain by the upper layers of the management system (for user configuration or Domain ID resolution)
SecurityLevel	2-bit integer	002-112	Asserts the security level for domain operation, as defined in Table 8-36 (valid for domain master only): 00 – non-secure 01 – low 10 – medium 11 – high

Table 8-32 – The attributes of the A_MGMT.REQ primitive

Name	Type	Valid range	Description
BeaconMode	2-bit integer	002-112	Asserts the beacon mode of the domain, as defined in clause 8.3.4 (valid for domain master only): 00 – beaconless mode 01 – asynchronous beacon mode 10 – synchronous beacon mode 11 – reserved
MaxAckRxTime	8-bit integer	001A16-14FF16	The value that determines the maximum time to transmit an LLC frame, as defined in clause 8.3.3.3.1 and clause 8.1.4. Represented as unsigned integer, in 10 ms units, with the range 250 ms-51000 ms. The value FFFF16 is a special value that indicates that the MaxAckRxTime limitation is disabled.
RelayMode	1-bit integer	0, 1	Indicates whether L2 relaying is enabled in the domain (valid for the domain master only)
MAMode	1-bit integer	0, 1	Indicates whether centralized medium access mode (see clause 8.2. or distributed medium access mode shall be used (valid for domain master only)
TransmMode	2-bit integer	002-112	Indicates whether normal or robust transmission shall be used in the domain, as per [ITU-T G.9955] (valid for domain master only): 00 – normal length for both PFH and preamble 10 – extended length for PFH and normal for preamble 01 – normal length for PFH and extended for preamble 11 – extended length for both PFH and preamble
Enable CX mechanism	1-bit integer	0-1	Enables coexistence mechanism (see clause 8.2.4): 0 – Disabled 1 – Enabled
Management message	Byte string	1-992 bytes	Internal management message sourced by upper layers (for transmission using LCDU)

Additional primitives are for further study.

8.4.3.4.2 A_MGMT.CNF

This primitive confirms the particular parameters used by the DLL or PHY. The attributes of the primitive are as defined in Table 8-33.

If the DLL or PHY is unable to comply with a particular attribute in the A_MGMT.REQ, it shall set this primitive to one, which means that the request is denied. Otherwise the value of the A_MGMT.CNF primitive shall be set to zero.

Table 8-33 – The attributes of the A_MGMT.CNF primitive

Name	Type	Valid range	Description
Status	Integer	0, 1	0: success 1: request is denied

Additional primitives are for further study.

8.4.3.4.3 A_MGMT.IND

This primitive provides the AE management with management parameters received or derived by the DLL or PHY management and submitted to the A-interface. The attributes of the primitive are defined in Table 8-34.

Table 8-34 – The attributes of the A_MGMT.IND primitive

Name	Type	Valid range	Description
DomainID	16-bit integer	Table 8-7	The value of the Domain ID selected by the DLL (self-configuration mode)
DomainIDConflict	1-bit flag	N/A	Indicates that same Domain ID detected and change of the Domain ID is required
NeighborDomainsList	m 16-bit integers	Table 8-7	Values of m discovered neighbour Domain IDs, each formatted as defined in Table 8.7
FrameTxFailure	1-bit flag	N/A	Shall be raised by the DLL management entity to indicate that the transmitted LLC frame was dropped due to persistent error or timeout. Details are described in clause 8.3.3.3.1
TxSuspended	Flag and a 16-bit integer	For the 16-bit integer, see Table 8-7	Shall be raised by the DLL management entity to indicate that transmission to the destinations address attached in the 16-bit field is not allowed. Details are described in clause 8.3.6
ResourceLimit	1-bit flag	N/A	Shall be raised by the DLL management entity to indicate that the node lacks resources to continue receive and transmit frames. Details are described in clause 8.3.3.3.1.
SecurityLevel	2-bit integer	002-112	Indicates to the upper layers the security level asserted in the domain, as defined in Table 8-36 (valid for end-nodes only): 00 – non-secure 01 – low 10 – medium 11 – high
BeaconMode	2-bit integer	002-112	Indicates to the upper layer the beacon mode of the domain, as defined in clause 8.3.4 (valid for end-nodes only): 00 – beaconless mode 01 – asynchronous beacon mode 10 – synchronous beacon mode 11 – reserved

Table 8-34 – The attributes of the A_MGMT.IND primitive

Name	Type	Valid range	Description
MaxAckRxTime	8-bit integer	001A16-14FF16	Indicates to the upper layers the value of the maximum allowed time to transmit an LLC frame, as defined in clause 8.3.3.3.1 and clause 8.1.4. Represented as unsigned integer, in 10 ms units, with the range 250 ms-51000 ms. The value FFFF16 is a special value that indicates that the MaxAckRxTime limitation is disabled. Valid for end-nodes only.
MAMode	1-bit integer	0, 1	Indicates whether centralized medium access mode (1) or distributed medium access mode (0) shall be used (see clause 8.2.2.3). This parameter is valid for the domain master only)
TransmMode	2-bit integer	002-112	Indicates whether a node shall use normal or robust transmission, as per [ITU-T G.9955] (valid for end-nodes only): 00 – normal length for both PFH and preamble 10 – extended length for PFH and normal for preamble 01 – normal length for PFH and extended for preamble 11 – extended length for both PFH and preamble
Management message	Byte string	1-992 bytes	Internal management message recovered from the received LCDU

Additional primitives are for further study.

8.4.3.4.4 A_MGMT.RES

This primitive is for further study

8.5 Domain management protocols

8.5.1 Setup of a domain

Domain in considered to be established if, and only if, there is a node operating as a domain master with a particular DOMAIN_ID. Therefore, to establish a domain the following has to be done:

– one of the nodes takes a role of the domain master;

– the domain master selects a DOMAIN_ID for its domain;

– all other nodes intended to be part of the established domain take a role of end-nodes and register into the domain.

At any time, only one node in the domain may take the role of a domain master, while multiple nodes may be capable to operate as a domain master (DM-capable nodes). A domain master selection protocol defined in clause 8.5.1.1 shall be used to pick a single domain master in the presence of multiple DM-capable nodes.

Each domain is uniquely identified by its domain name which is set by the user or generated autonomously by the domain master that initializes the domain. Setup of the domain name is beyond the scope of this Recommendation. The Domain Name Indicator (DNI) transmitted in the beacon frame indicates the domain name. The format of the DNI is defined in clause 8.5.1.3. Once generated, the DNI shall not change when the role of the domain master is passed to another node.

NOTE – The DNI is similar to SSID in 802.11 networks.

The domain master shall set a DOMAIN_ID for its domain using the DOMAIN_ID selection protocol defined in clause 8.5.1.2; this protocol provides, for a new-established domain, a value of DOMAIN_ID that is different from those in neighbouring domains.

8.5.1.1 Domain master selection protocol

This is for further study.

8.5.1.1.1 Domain master selection at initialization

This is for further study.

8.5.1.1.2 Domain master recovery if no backup domain master is assigned

This is for further study.

8.5.1.2 Domain ID selection protocol

This is for further study.

8.5.1.2.1 Domain ID conflicts detection

This is for further study.

8.5.1.2.2 Domain ID conflicts resolution

This is for further study.

8.5.1.3 Parameters of domain setup procedures

This is for further study.

8.5.2 Admission to the domain

The procedure of admission a new node to the domain (also called "registration") is for further study.

8.5.3 Backup of the domain master

Assignment of a node to backup the domain master and domain master backup procedures are for further study.

8.5.4 Channel estimation

The channel estimation procedure allows the transmitter to request for a recommendation from the receiver on the values of at least the following transmission parameters:

– Tone mask

– Modulation

– Repetition

– Interleaver mode

– Convolution coding rate

– Maximum RS code word size

NOTE – The format of the channel estimation messages presented in clause 8.4.2.3 allows extending of the list of the parameters, if necessary.

The transmitter may, at its own discretion, request for a recommendation from the receiver using one of the following two methods:

– Transmission parameter request message (TP.req), as specified in clause 8.4.2.

– Setting the TP-REQ indicator on the MPH, as specified in clause 8.1.3.1.1.

Upon reception of a TP.req message, the receiver shall respond with a TP.ind message (see clause 8.4.2.3, containing its recommendation for transmission parameters (see Table 8-24).

The transmitter may, at its own discretion, set the TP-REQ indicator on the MPH of any regular data or management frame. No special setting for transmission parameters in frames containing the TP-REQ indicator is required.

Upon reception of a frame with an MPH without error and with the TP.REQ indicator set to 01, the receiver shall respond with a TP.ind message containing its recommendation for transmission parameters in the format defined in Table 8-24.

Upon reception of a frame with an MPH without error and with the TP.req indicator set to 10, the receiver, if capable, shall respond by using an Extended Imm-ACK frame (see clause 8.3.3.1.1) containing its recommendation for transmission parameters as defined in Table 8-24.

Upon reception of a frame with an MPH without error and with the TP.req indicator set to 11, the receiver shall respond with an Imm-ACK frame containing its recommendation for partial set of transmission parameters. The format of partial response is defined in Table 8-16.

If the receiver is incapable to provide the report by sending an Extended Imm-ACK frame, it shall reply to the transmitter with an Extended Imm-ACK frame indicating unable to comply, as described in clause 8.3.3. A node that requested a TP-recommendation using Extended Imm-ACK, if necessary, may further send another type of TP request.

The TP.ind message shall always be transmitted using RCM, as defined in [ITU-T G.9955], with the following transmission parameters:

– code rate: 1/2

– number of repetitions: at least 4.

NOTE – Upon reception of the TP.ind message from the receiver, the transmitter may use the receiver's recommendation of transmission parameters for subsequent transmissions to that receiver.

NOTE – Using the robust communication mode (RCM) for transmission of TP.req message improves performance of channel estimation.

8.5.5 Beacon related management procedures

Beacon related management procedures are for further study.

8.5.6 Relaying database management

Management procedures related to relaying database are for further study.

8.5.7 Flow control

Flow control procedures are for further study.

8.6 Security

Security is provided by encryption and authentication of the relevant data and management frames communicated between the nodes of the domain. The specified encryption method is based on AES-128 and is described in clause 8.6.1.

Node authentication, generation and distribution of encryption keys between nodes, encryption key updates, and node authentication updates are provided by a set of Authentication and Key Management (AKM) procedures, described in clause 8.6.2. The AKM procedure can establish both group keys (i.e., a unique set of keys for a particular group of nodes) and pair-wise keys (i.e., a unique set of keys per every pair of communicating nodes).

8.6.1 Encryption

The specified encryption method is based on the Advanced Encryption Standard (AES) according to [NIST PUB 197] and Counter Mode with Cipher Block Chaining Message Authentication Code algorithm (CCM), according to [NIST SP 800-38C]. The encrypted LLC frame is communicated using CCM protocol (CCMP) that includes a CCMP header, encrypted frame, and Message Integrity Code (MIC) for frame authentication. CCMP header includes information necessary to facilitate decryption.

8.6.1.1 Description of CCMP

8.6.1.1.1 CCM encryption

The CCM encryption algorithm complies with [NIST SP 800-38C].

Prerequisite:

- Block-cipher algorithm AES-128, [NIST PUB 197].
- Encryption key K: 128 bits long.
- Counter-generation function: produces 128-bit counter blocks (*Ctr*).
- Length of the message integrity code (MIC), *Tlen*.

Input:

- Nonce N: a bit-string of less than 128 bits long.
- Payload P of length *Plen* bytes: the part of the data unit (APDU or LCDU) to be both encrypted and protected by the MIC.
- Associated data A of length *Alen* bytes: the unencrypted part of the data unit and additional data to be protected by the MIC.

Output:

- ciphertext (encrypted payload) C
- MIC of the length *Tlen*.

Steps of the algorithm:

1. Apply the formatting function, as described in clause 8.6.1.1.3 to the input variables N, A, and P to produce the 128-bit blocks B_0, B_1, …, B_r.
2. Set Y_0 = CIPH$_K$(B_0): apply the block-cipher algorithm with the key K.
3. For i = 1 to r, do Y_i = CIPH$_K$($B_i \oplus Y_{i-1}$): chaining the blocks.
4. Set T = MSB$_{Tlen}$(Y_r): the *Tlen* most significant bits of the final round of this computation.

   NOTE – These first 4 steps constitute the cipher-block chaining that calculates the value of T to generate MIC. If the value of T computed for the received *frame* will still match the received value of MIC, it assures authenticity (integrity) of the received message with sufficiently high probability.
5. Generate the counter blocks Ctr_0, Ctr_1,…, Ctr_m, where m = *ceiling* (*Plen*/128).
6. For j = 0 to m, do S_j = CIPH$_K$(Ctr_j): apply the block-cipher algorithm with the key K.
7. Set S = $S_1 \| S_2 \| \ldots \| S_m$: this defines the string of encrypted counter blocks. Note that S_0 is skipped.
8. Compute C = ($P \oplus$ MSB$_{Plen}$(S)) || ($T \oplus$ MSB$_{Tlen}$(S_0)): the ciphertext is the string of counter blocks XOR'd with the payload data; the message-authentication code is produced by XOR'ing T with S_0.

NOTE – The second 4 steps constitute generation of the actual ciphertext of encrypted data concatenated with the MIC. The associated data, *A*, are not incorporated into the ciphertext *C*, but incorporated in the calculation of the MIC, and thus are protected against undetected alteration. Thus, relevant part of the data that are sent unencrypted, as described in clause 8.6.1.2.1, are included to be a part of the *A*-data.

A block diagram illustrating the CCM encryption and MIC generation algorithm described above for the case of $33 \le Alen \le 48$ bytes is presented in Figure 8-20.

The *B*-blocks from B_3 onwards contain payload bits (*P*) and blocks B_0, B_1, and B_2 contain associated data bits (*A*). The AES-blocks stand for AES-128 functions. Those are fed by 128-bit counter blocks (Ctr_0 – Ctr_m). The PAD compliments the last payload block to 128 bits. If *Alen* is longer than 48 bytes or shorter than 32 bytes, the number of blocks in Figure 8-20 containing associated data has to be, respectively, increased or decreased. The output (ciphertext) includes 128-bit blocks $C_1 – C_{m-1}$; the last block is shorter by the size of the PAD.

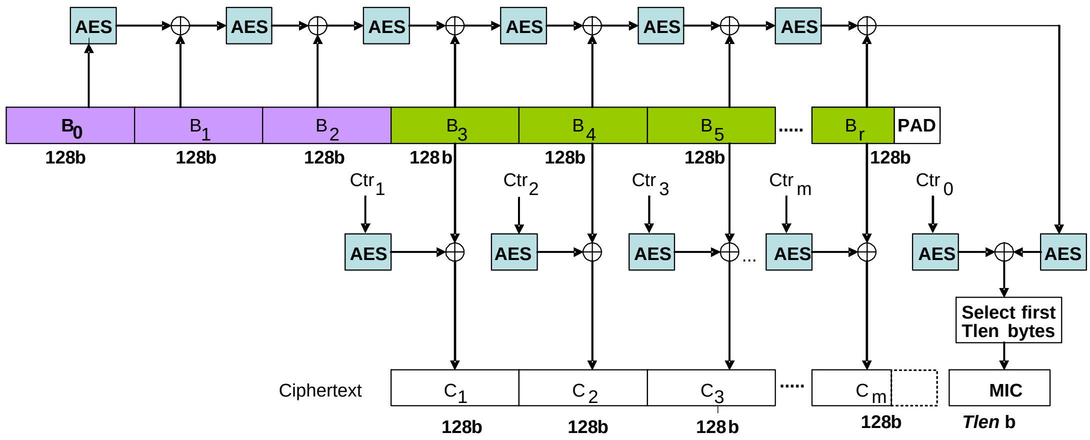


Figure 8-20 – Functional diagram of CCM encryption & message authentication with 3 blocks of associated data

8.6.1.1.2 Parameters

Valid values of the CCM encryption parameters are presented in Table 8-35.

Table 8-35 – CCM parameters

Parameter	Valid values
MIC size (*Tlen*), bytes	4, 8, 16
Payload size (*Plen*), bytes	$\le (2^{11} – 1)$
Associated data size (*Alen*), bytes	Variable

NOTE – Selection of MIC size should be based on the guidance provided in [NIST SP 800-38C].

8.6.1.1.3 Input variables

The input variables to support CCM encryption are:

- Counter blocks (Ctr_n)
- Associated data blocks (B_0, $B_1 – B_M$)
- Payload blocks (B_{M+1} to B_r)
- Encryption key.

The 16-byte counter blocks Ctr_0, Ctr_1…, Ctr_m shall have the format presented in Table 8-36. Each block shall comprise a 1-byte Flag, a 13-byte nonce, and a 2-byte counter block number (in the range from 0 to *m*) expressed as a 16-bit binary integer. All bytes of the counter block shall be formatted MSB first: the first bit of the byte 0 is MSB (bit 7) and the last bit of the byte 15 is LSB (bit 0).

Table 8-36 – Format of the Ctr blocks

Byte number	**0**	**1, 2…, 13**	**14, 15**
Contents	Flags (Note)	Nonce	Counter block number
NOTE – The content of the Flags-byte is: bits [7:6] – reserved by ITU-T for NIST, shall be set to 00 bits [5:3] – shall be set to 000 bits [2:0] – shall be set to 001.			

The 13-byte nonce shall be constructed as presented in Table 8-37. The MSB of byte 0 of the nonce in Table 8-37 shall be mapped to the MSB of byte 1 of the *Ctr* block. The value and format of the frame number (FN) shall be as specified in clause 8.6.1.2, Table 8-41. The LSB of the FN shall be mapped to the LSB of byte 12 of the nonce. The source address field of the nonce shall have a format defined in Table 8-38, which depends on the selected addressing option. All bytes of the nonce shall be formatted MSB first: the first bit of the byte 0 is MSB (bit 7) and the last bit of the byte 12 is LSB (bit 0).

Table 8-37 – Format of the nonce

Byte number	**0**	**1 – 8**	**9-12**
Contents	Flags (Note)	Source address	Frame number (FN)
NOTE – The content of the Flags-byte is: Bits [7:3] – The same bits of the Byte 0 in the CCMP header Bits [1:0] – The LLC frame priority (see clause 8.2.1, clause I.1.3) Bits [2] – Reserved by ITU-T. All reserved bits of the Flags byte shall be set to zero.			

Table 8-38 – Format of the nonce Source Address field

Bytes	**MESH header present (MHI = 1)**		**No MESH header present (MHI = 0)**
	V-bit = 0	**V-bit = 1**	
1-2	SA field of the LFH	SA field of the LFH	SNID field of MPH
3-4		SDID field of the MPH	SDID field of MPH
5-6		0x00	0x00
7-8		0x0000	0x0000

The value of the nonce (for the given key) shall never be the same for different encrypted payloads, and shall always be the same for identical encrypted payloads (e.g., when APDU or LCDU is retransmitted or relayed). The encryption key shall be changed promptly to avoid repetition of the nonce.

The 16-bytes block B_0 shall have a format as presented in Table 8-39. The length of the encrypted payload in octets (*Plen*) shall be represented as 16-bit unsigned integer with LSB mapped to the LSB of byte 15 of B_0.

Table 8-39 – Format of block B_0

Byte number	0	1, 2…, 13	14-15
Content	Flags (Note)	Nonce	Length of the payload (*Plen*)
NOTE – The content of the Flags byte: Bit [7] – Reserved by ITU-T for NIST, shall be set to zero Bit [6] – Shall be set to one Bits [5:3] – Shall indicate the length of the MIC encoded as: 001 – 4-byte MIC 011 – 8-byte MIC 111 – 16-byte MIC All other values are reserved by ITU-T Bits [2:0] – shall be set to 001			

Blocks B_1-B_M shall be all 16-byte long, and shall have a format as presented in the Table 8-40. Byte 0 is the first byte of block B_1, byte (M×16-1) is the last byte of block B_M. The number of blocks, M, shall be: M = *ceiling*(*Alen*/16).

The receiver shall derive the value of the *Alen* from the corresponding settings in the LFH (see Table 8-1), the type of the encrypted data (APDU or LCDU) and the APDU format determined by the used application protocol (both parameters indicated in the LPH).

Table 8-40 – Format of blocks B_1-B_M

Block	Bytes	Contents
$B_1 – B_M$	0-1	Reserved by ITU-T (Note)
	2-V_1	LFH, except field HopsLft, if present in LFH, shall be set to 0x00; byte 0 of LFH shall be mapped on byte 2 of $B_1 – B_M$ block
	$(V_1+1) – V_2$	MPH fields from byte 0 to byte 10; byte 0 of MPH shall be mapped to byte V_1+1 and bits [3:0] of byte 0 and all bits of byte 2 shall be set to 0. NOTE – $V_2 = V_1 + 15$
	$(V_2+1) – V_3$	Unencrypted part of APDU or LCDU as defined in clause 8.6.1.2.2. The byte 0 of the unencrypted part shall be mapped on byte (V_2+1)
	(V_3+1) – (M*16-1)	Reserved by ITU (Note)
NOTE – All reserved bits shall be set to zeros by the transmitter and ignored by the receiver.		

All bytes of the block B_0 and the associated data blocks B_1 – B_M shall be formatted MSB first: the first bit of the byte 0 is MSB (bit 7) and the last bit of the last byte of B_M block is LSB (bit 0).

Payload blocks (B_{M+1} to B_r) are 16-byte long and shall contain bytes of the APDU or LCDU to be encrypted (see clause 8.6.1.2.2, Encrypted part of APDU or LCDU). The APDU or LCDU bytes shall be mapped to Payload blocks in sequential order, so that the first byte of the APDU or LCDU to be encrypted shall be mapped on byte 0 of B_{M+1}, the second byte of the payload is mapped on byte 1 of B_{M+1}, the 17-th byte shall be mapped on byte 0 of B_{M+2}, and so on. If the last byte of the payload does not fall on 15-th byte of B_r, the payload shall be padded to fill the last block by appending zero bytes (00_{16}). All bytes of the payload blocks shall be formatted MSB first: the first bit of byte 0 of block B_{M+1} is MSB (bit 7) and the last bit of the byte 15 of block B_r is LSB (bit 0). The first byte of the ciphertext, C_1, shall be the first byte of the encrypted part of the LLC frame (Encrypted APDU or LCDU in Figure 8-5).

Encryption key is 128 bits long and shall be generated and assigned as described in clause 8.6.2.

8.6.1.2 CCM encryption protocol (CCMP)

8.6.1.2.1 Functional description

The functional model of the CCMP is presented in Figure 8-21. The incoming APDU or LCDU is encrypted by the CCM, as described in clause 8.6.1.1 and clause 8.6.1.2.2. The LFH is sent unencrypted. The relevant parts of LFH and the MPH, and the unencrypted part of the APDU (or LCDU) are protected by the MIC as a part of Associated Data, as defined in Table 8-40.

The key ID, the frame number (FN), and the length of the MIC associated with the encrypted LLC frame are conveyed to the receiver in the CCMP header to assist decryption; CCMP header is sent unencrypted and described in clause 8.6.1.2.3, but is also protected by the MIC. Construction of the nonce (*N*) and the Associated data is as described in clause 8.6.1.1. Construction of the CCMP header is defined in clause 8.6.1.2.3.

When encryption is not required, the CCMP header can be used to protect the packet order by discarding packets that arrive out of order (e.g., in case of relaying). The packet order protection (POP) settings are defined in Table 8-42.

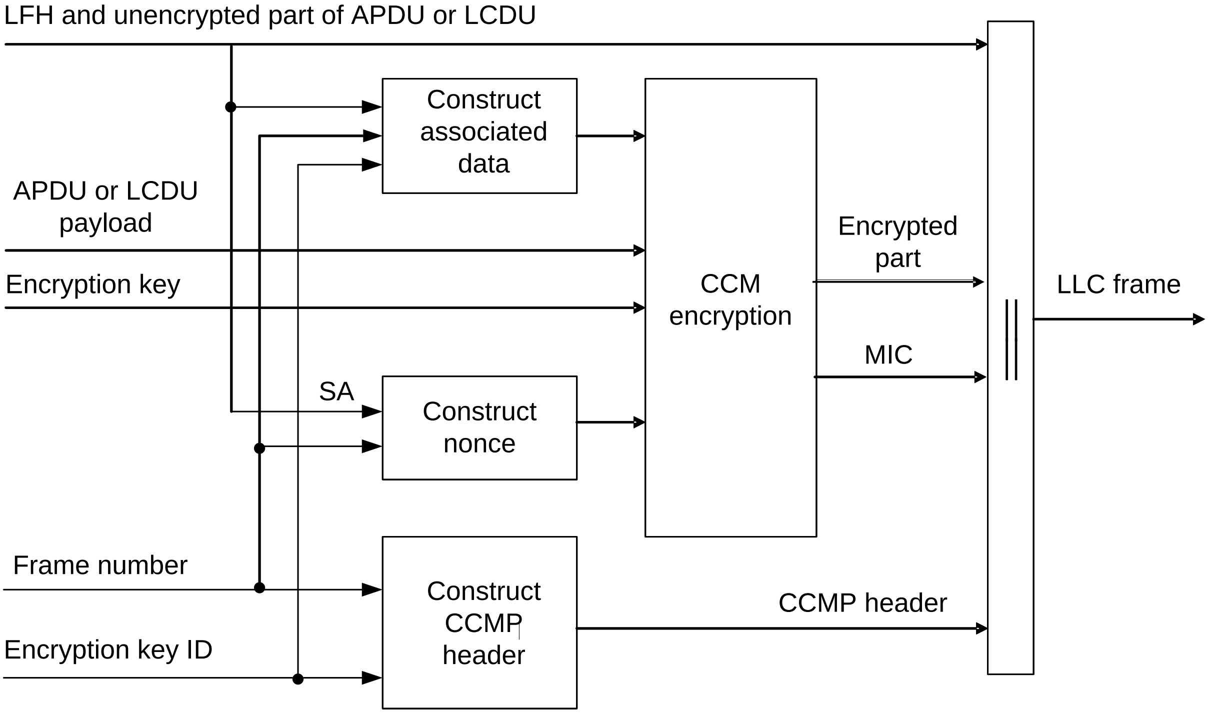


Figure 8-21 – Functional diagram of CCMP encryption

8.6.1.2.2 CCMP encryption format

The format of the encrypted LLC frame is presented in Figure 8-22 (see also Figure 8-5). The encrypted APDU or LCDU consists of four parts: CCMP header, unencrypted part, encrypted part (ciphertext, see clause 8.6.1.1.1), and MIC.

LFH	CCMP header	Unencrypted part of APDU or LCDU	Encrypted part of APDU or LCDU	MIC
Clause 8.1.2.1.2	Clause 8.6.1.2.3	This clause	Clause 8.6.1.1.1	Clause 8.6.1.1.1

Figure 8-22 – Format of a CCMP-encrypted LLC frame

The format of the LFH shall be as described in clause 8.1.2.1.2. The format of CCMP header shall be as described in clause 8.6.1.2.3. Generation of the ciphertext and the MIC shall be as described in clause 8.6.1.1.1.

The unencrypted part of the APDU depends on the type of APC and described per APC in Annex I. No unencrypted parts shall be present in encrypted LCDU.

8.6.1.2.3 CCMP header

The CCMP header consists of 5 bytes and shall have a format as presented in Table 8-41. It conveys the encryption key identification number (key ID), the length of the MIC, and the security frame number (FN). These three parameters are necessary for decryption and authentication of the received frame.

The length of the MIC shall be selected in the range defined in Table 8-35, according to the procedure defined in clause 8.6.2.

Table 8-41 – CCMP header format

Field	Octet	Bits	Description
Security control	0	[2:0]	Length of the MIC encoded as: 001 – 4-byte MIC 011 – 8-byte MIC 111 – 16-byte MIC A value of 000 indicates POP node (no encryption, see Table 8-42, Note 1) All other values are reserved by ITU-T
		[4:3]	The type of encryption key: 00 – Node-to-Node (NN) key or Domain Membership Key (DMK) 10 – Domain Broadcast (DB) key 01 – Node Authentication (NA) key 11 – Reserved by ITU-T
		[6]	Encryption key ID: 0 – first key 1 – second key
		[7]	Reserved by ITU-T (Note 2)
Frame number	1-4	[31:0]	32-bit FN, formatted as an unsigned binary integer
NOTE 1 – If bits [2:0] of octet 0 are set to 0, all other bits of octet 0 shall be ignored by the receiver. NOTE 2 – Bits that are reserved by ITU-T shall be set to zero by the transmitter and ignored by the receiver.			

The format of the key type is a 2-bit unsigned binary integer, intended to identify the specific type of the key used to encrypt the frame. Two encryption keys of each type shall be established, with key IDs 0 and 1, respectively, during the AKM procedure, as described in clause 8.6.2. Keys assigned for communication with different peers may have the same key IDs. Use of encryption keys of different types and different IDs is described in clause 8.6.2.

NOTE – Two keys are necessary to facilitate the key update procedure without interruption of the connection: while the pool of valuable FN values is expired for one key, the transmitting nodes change the key to the second one, allowing for the first key to get updated. The details are described in clause 8.6.2.

The FN is a serial number of the encrypted frame and shall be represented as a 32-bit unsigned binary integer (LSB mapped on bit 0 of byte 4). The FN shall be set to its initial value when a new encryption key is established and increased by 1 with every encrypted frame passed using this key. In case of POP, FN shall be increased by 1 with every frame passed and wrap around to value 1. FN shall never be repeated for the same value of the key: the key shall be changed prior to FN reaching its maximum value. The default initial value is 1.

8.6.1.3 Requirements for the receiver

The receiver shall decrypt the received message and compute the MIC using the encryption key which type and ID is indicated in the CCMP header. If the computed value of the MIC does not match the value of the MIC in the received frame, the receiver shall discard the frame.

If the receiver detects that encryption rules in the received frame are violated, it shall discard the frame. Those violations are:

- The FN is 0 or is smaller than or equal to the FN values received in previous LLC frames of the same priority from the same originating node with no change of the encryption key.
- The length of the MIC does not fit the security level assigned in the domain.

NOTE – The LLC frame priority is equal to the value of the PRI field communicated in the MPH of the first MPDU carrying segments of the LLC frame (this PRI value is the minimal value of PRI over all MPDUs carrying segments of the LLC frame, and is equal to the value of LPRI field of the MESH header, if present).

8.6.1.4 Configuration of security level

The valid security levels and corresponding parameters are defined in Table 8-42.

Table 8-42 – Valid security levels

Security level	Encryption	MIC, bytes
High	Yes	16
Medium	Yes	8
Low	Yes	4
POP	No	0

All nodes of the domain shall be set to the same level of security, i.e., encryption and the size of the MIC. The required level of security in the domain is determined by the user via the A_MGMT.REQ primitive SecurityLevel (see clause 8.4.3.4.1). Further, the asserted security level is conveyed by the domain master to every node in the domain as defined in clause 8.3.4, clause 8.4.2.1, clause 8.5.5). Nodes indicate the adopted security level to the upper layers using A_MGMT.IND primitive SecurityLevel (see 8.4.3.4.3). Nodes shall discard all frames which security level is below the one required in the domain.

8.6.2 Authentication and key management procedures

For further study.

Annex A

G3-PLC DLL specification

(This annex forms an integral part of this Recommendation.)

NOTE – This is a stand-alone annex which can be implemented independently from the main body of this Recommendation.

A.1 Normative references

The following referenced documents are indispensable for the application 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.

[IEEE 802-2001] IEEE Std 802-2001 (R2007), *IEEE Standard for Local and Metropolitan Area Networks. Overview and Architecture*.

[IEEE 802.15.4] IEEE 802.15.4:2006, *IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 15.4: Wireless Medium Access (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs).*

[IEEE 802.2] IEEE 802.2:1998, *Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 2: Logical Link Control*.

[IETF RFC 2284] IETF RFC 2284 (1998), *PPP Extensible Authentication Protocol (EAP).*

[IETF RFC 2865] IETF RFC 2865 (2000), *Remote Authentication Dial In User Service (RADIUS).*

[IETF RFC 3748] IETF RFC 3748 (2004), *Extensible Authentication Protocol (EAP).*

[IETF RFC 4291] IETF RFC 4291 (2006), *IP Version 6 Addressing Architecture*.

[IETF RFC 4764] IETF RFC 4764 (2007), *The EAP-PSK Protocol: A Pre-Shared Key Extensible Authentication Protocol (EAP) Method*.

[IETF RFC 4862] IETF RFC 4862 (2007), *Ipv6 Stateless Address Autoconfiguration*.

[IETF RFC 4944] IETF RFC 4944 (2007), *Transmission of IPv6 Packets over IEEE 802.15.4 Networks*.

A.2 Acronyms

AAA	Authentication, Authorization, and Accounting
ACK	Acknowledge
ADP	Adaptation
AGC	Automatic Gain Control
BPSK	Binary Phase Shift Keying
CFA	Contention Free Access
CP	Cyclic Prefix
CRC	Cyclic Redundancy Check

DBPSK	Differential Binary Phase Shift Keying
DQPSK	Differential Quadrature Phase Shift Keying
D8PSK	Differential Eight Phase Shift Keying
FCH	Frame Control Header
GI	Guard Interval
GMK	Group Master Key
IEEE	Institute of Electrical and Electronics Engineers
IEC	International Electrotechnical Commission
LBD	LoWPAN BootStrapping Device
LBP	LoWPAN Boostrapping Protocol
LSF	Last Segment Flag
MAC	Media Access Control
MIB	Management Information Base
MPDU	MAC Protocol Data Unit
NACK	Negative ACKnowledge
NIB	Neighbour Information Base
NSDU	Network Service Data Unit
OFDM	Orthogonal Frequency Division Multiplexing
PAN	Personal Area Network
PDC	Phase Detection Counter
PHY	PHYsical layer
PIB	PAN Information Base
PLC	Power Line Communication
POS	Personal Operating Space
RADIUS	Remote Authentication Dial in User Service
RC	Repetition Code
RES	Reserved (bit fields)
S-FSK	Spread Frequency Shift Keying
SC	Segment Count
SNR	Signal to Noise Ratio
TMR	Tone Map Request
TX	Transmit
6LoWPAN	IPv6 over Low power Wireless Personal Area Networks

Furthermore, the abbreviations given in the following clauses apply also:

– Clause 4 of [IEEE 802.15.4],

– Clause 1.2 of [IETF RFC 4944].

A.3 Data link layer specification

A.3.1 Introduction

The G3-PLC data link layer specification comprises two sublayers:

- The MAC sublayer based on [IEEE 802.15.4]; and
- The Adaptation sublayer based on [IETF RFC 4944]: Transmission of IPv6 Packets over IEEE 802.15.4 Networks (6LoWPAN).

The present standard specifies the necessary selections from and extensions to these standards.

A.3.2 Conventions

In the present clause, the status of each requirement from the reference documents is given using the following convention:

– I = "Informative": The statements of the reference document are provided for information only;

– N = "Normative": The statements of the reference document shall apply without modifications or remarks;

– S = "Selection": The statements of the reference document shall apply with the selections specified;

– E = "Extension": The statements of the reference document shall apply with the extensions (modifications and remarks noted under the part title) specified;

– N/R = "Not Relevant": The statements of the reference document do not apply. An explanation may be given under the part title.

A.3.3 MAC sublayer specification

A.3.3.1 MAC sublayer service specification (based on [IEEE 802.15.4] clause 7.1)

A.3.3.1.1 Selections from [IEEE 802.15.4] clause 7.1: MAC sublayer service specification

The MAC services and primitives are as given in clause 7.1.1 to 7.1.17 of [IEEE 802.15.4] together with the following statements and modifications as shown in Table A.1.

References to clauses in the "Clause" column refer to the referenced document, while references to clauses/annexes in the "Title & Remarks" column refer to this standard unless specifically indicated otherwise. The interpretation of the statement column is given in clause A.3.2.

Table A.1 – Selections from [IEEE 802.15.4] clause 7.1

Clause	Title & remarks/modifications	Statement
7.1	MAC sublayer service specification	N
7.1.1	MAC data service – MCPS-PURGE primitives are not used in this specification.	S
7.1.1.1	MCPS-DATA.request	N
7.1.1.1.1	Semantics of the service primitive – Extension: Additional QualityOfService parameter: see A.3.3.1.2. – Only non beacon-enabled PAN is used; – Bit b2 of TxOptions parameter shall always be 0 See G3 Annex D of the present document for complete semantics description of this primitive.	S, E
7.1.1.1.2	Appropriate usage	N

Table A.1 – Selections from [IEEE 802.15.4] clause 7.1

Clause	Title & remarks/modifications	Statement
7.1.1.1.3	Effect on receipt – GTS transmission is not used; – Only unslotted CSMA-CA for nonbeacon-enabled PAN is used; – Indirect transmission is not supported	S
7.1.1.2	MCPS-DATA.confirm	N
7.1.1.2.1	Semantics of the service primitive – Modification: Timestamp is optional and defined as the absolute time in milliseconds at which the frame was created and eventually after the encryption (32 bit value).	S, E
7.1.1.2.2	When generated	N
7.1.1.2.3	Appropriate usage	N
7.1.1.3	MCPS-DATA.indication	N
7.1.1.3.1	Semantics of the service primitive – Extension: Additional QualityOfService parameter: see clause A.3.3.1.2. – Modification: Timestamp is optional and defined as the absolute time in milliseconds at which the frame was received and constructed, decrypted (assuming encryption was valid) (32 bit value). See G3 Annex D of the present document for complete semantics description of this primitive.	S, E
7.1.1.3.2	When generated	N
7.1.1.3.3	Appropriate usage	N
7.1.1.4	MCPS-PURGE.request	N/R
7.1.1.5	MCPS-PURGE.confirm	N/R
7.1.1.6	Data service message sequence chart	N
7.1.2	MAC management service	N
7.1.3	Association primitives	N/R
7.1.3.1	MLME-ASSOCIATE.request – MLME-ASSOCIATE.request is not used in this specification. Association is performed by the 6LoWPAN Bootstrap Protocol described in clause A.3.4.5 of the present document.	N/R
7.1.3.2	MLME-ASSOCIATE.indication – MLME-ASSOCIATE.indication is not used in this specification. Association is performed by the 6LoWPAN Bootstrap Protocol described in clause A.3.4.5 of the present document.	N/R
7.1.3.3	MLME-ASSOCIATE.response – MLME-ASSOCIATE.response is not used in this specification. Association is performed by the 6LoWPAN Bootstrap Protocol described in clause A.3.4.5 of the present document.	N/R
7.1.3.4	MLME-ASSOCIATE.confirm – MLME-ASSOCIATE.confirm is not used in this specification. Association is performed by the 6LoWPAN Bootstrap Protocol described in clause A.3.4.5 of the present document.	N/R

Table A.1 – Selections from [IEEE 802.15.4] clause 7.1

Clause	Title & remarks/modifications	Statement
7.1.3.5	Association message sequence chart – The association message sequence chart described in Figure 31 shall be ignored for this specification, as association is performed using the bootstrap mechanism described in clause A.3.4.5 of the present document.	N/R
7.1.4	Disassociation primitive	N/R
7.1.4.1	MLME-DISASSOCIATE.request – MLME-DISASSOCIATE.request is not used in this specification. Disassociation is performed by the 6LoWPAN Bootstrap Protocol described in clause A.3.4.5 of the present document.	N/R
7.1.4.2	MLME-DISASSOCIATE.indication – MLME-DISASSOCIATE.indication is not used in this specification. Disassociation is performed by the 6LoWPAN Bootstrap Protocol described in clause A.3.4.5 of the present document.	N/R
7.1.4.3	MLME-DISASSOCIATE.confirm – MLME-DISASSOCIATE.confirm is not used in this specification. Disassociation is performed by the 6LoWPAN Bootstrap Protocol described in clause A.3.4.5 of the present document.	N/R
7.1.4.4	Disassociation message sequence chart – The disassociation message sequence chart described in Figure 31 shall be ignored for this specification, as disassociation is performed using the bootstrap mechanism described in clause A.3.4.5 of the present document.	N/R
7.1.5	Beacon notification primitive	N
7.1.5.1	MLME-BEACON-NOTIFY.indication – Only nonbeacon-enabled PANs are used. – This primitive is generated upon reception of a beacon during an active scan	S
7.1.5.1.1	Semantics of the service primitive MLME-BEACON-NOTIFY.indication (PANDescriptor) PANDescriptor is described in Table A.F.6.	S
7.1.5.1.2	When generated – This primitive is generated upon reception of a beacon during an active scan.	S
7.1.5.1.3	Appropriate usage	N
7.1.6	Primitives for reading PIB attributes	N
7.1.6.1	MLME-GET.request	N
7.1.6.1.1	Semantics of the service primitive	N
7.1.6.1.2	Appropriate usage	N
7.1.6.1.3	Effect on receipt	N
7.1.6.2	MLME-GET.confirm	N
7.1.6.2.1	Semantics of the service primitive	N

Table A.1 – Selections from [IEEE 802.15.4] clause 7.1

Clause	Title & remarks/modifications	Statement
7.1.6.2.2	When generated	N
7.1.6.2.3	Appropriate usage	N
7.1.7	GTS management primitives – GTS are not used in the present specification	N/R
7.1.8	Primitives for orphan notification – Beacon synchronization is not used in the present specification	N/R
7.1.9	Primitives for resetting the MAC sublayer	N
7.1.9.1	MLME-RESET.request	N
7.1.9.1.1	Semantics of the service primitive	N
7.1.9.1.2	Appropriate usage	N
7.1.9.1.3	Effect on receipt	N
7.1.9.2	MLME-RESET.confirm	N
7.1.9.2.1	Semantics of the service primitive	N
7.1.9.2.2	When generated	N
7.1.9.2.3	Appropriate usage	N
7.1.10	Primitives for specifying the receiver enable time – The primitives for specifying the receiver enable time are not used in the present application of the norm. The receiver is always enabled	N/R
7.1.11	Primitives for channel scanning	N
7.1.11.1	MLME-SCAN.request	N
7.1.11.1.1	Semantics of the service primitive – The only supported values for the ScanType parameter is 0x01 for active scan. – The ScanChannels parameter is not used, and all of its 27 bits shall be set to 0. – The ChannelPage parameter is not used and shall be set to 0. – The SecurityLevel shall be 0. Thus the KeyIdMode, KeyIndex and KeySource parameters can be ignored and set to 0.	S
7.1.11.1.2	Appropriate usage – Only active scan is supported – ED scans, passive scans and orphan scans are not used. All devices shall be capable of performing active scans.	S
7.1.11.1.3	Effect on receipt – Only active scan is supported, – ED scan, passive scan and orphan scan are not supported, – There is no physical channel notion during the scans, as the underlying PHY layer does not support multiple channels.	S
7.1.11.2	MLME-SCAN.confirm During active scan, MLME-BEACON-NOTIFY.indication is generated in response to MLME-SCAN.request as soon as a beacon is received.	N
7.1.11.2.1	Semantics of the service primitive	S
7.1.11.2.2	When generated	S

Table A.1 – Selections from [IEEE 802.15.4] clause 7.1

Clause	Title & remarks/modifications	Statement
7.1.11.2.3	Appropriate usage	N
7.1.11.3	Channel scan message sequence chart – Figure 79 shall be ignored (ED scan not supported) – Figure 82 shall be ignored (passive scan not supported) – Figure 86 shall be ignored (orphan scan not supported) – Active scan message sequence chart is specified in clause A.3.4.5.2.2 of the present document, and replaces figure 83 of the reference document.	S
7.1.12	Communication status primitive	N
7.1.12.1	MLME-COMM-STATUS.indication	N
7.1.12.1.1	Semantics of the service primitive – Valid values for the status parameters are: SUCCESS, CHANNEL_ACCESS_FAILURE, NO_ACK, COUNTER_ERROR, FRAME_TOO_LONG, IMPROPER_KEY_TYPE, IMPROPER_SECURITY_LEVEL, SECURITY_ERROR, UNAVAILABLE_KEY, UNSUPPORTED_LEGACY, UNSUPPORTED_SECURITY or INVALID_PARAMETER	S
7.1.12.1.2	When generated – This primitive is not used to notify the upper layer about association, disassociation, indirect transmission and transactions management	S
7.1.12.1.3	Appropriate usage	N
7.1.13	Primitives for writing PIB attributes	N
7.1.13.1	MLME-SET.request	N
7.1.13.1.1	Semantics of the service primitive	N
7.1.13.1.2	Appropriate usage	N
7.1.13.1.3	Effect on receipt	N
7.1.13.2	MLME-SET.confirm	N
7.1.13.2.1	Semantics of the service primitive	N
7.1.13.2.2	When generated	N
7.1.13.2.3	Appropriate usage	N
7.1.14	Primitives for updating the superframe configuration – This primitive is only used on the PAN coordinator in case of network formation (see clause A.3.5.1 of the present document).	S
7.1.14.1	MLME-START.request – This primitive is only used to initiate a new PAN.	S
7.1.14.1.1	Semantics of the service primitive – Primitive parameters shall be set as described in clause A.3.5.1 of the present document.	S
7.1.14.1.2	Appropriate usage	N
7.1.14.1.3	Effect on receipt – Primitive parameters shall be set as described in clause A.3.5.1 of the present document.	S

Table A.1 – Selections from [IEEE 802.15.4] clause 7.1

Clause	Title & remarks/modifications	Statement
7.1.14.2	MLME-START.confirm	N
7.1.14.2.1	Semantics of the service primitive	N
7.1.14.2.2	When generated	N
7.1.14.2.3	Appropriate usage	N
7.1.14.3	Message sequence chart for updating the superframe configuration – Figure 38 shall be ignored.	N/R
7.1.15	Primitives for synchronizing with a coordinator – This part is used to inform the upper layers in case of a PAN ID conflict or PAN realignment.	S
7.1.15.1	MLME-SYNC.request	N/R
7.1.15.2	MLME-SYNC-LOSS.indication – PAN ID conflict detection is performed by the 6LoWPAN Bootstrap Protocol as described in clause A.3.5.2 of the present document.	N/R
7.1.15.3	Message sequence chart for synchronizing with a coordinator – Synchronization with beacons is not used in the present specification	N/R
7.1.16	Primitives for requesting data from a coordinator – Indirect transmission and transactions are not supported by the present specification	N/R
7.1.17	MAC enumeration description	N
NOTE – Timestamp shall refer to a free running counter in milliseconds. The counter is initialized to zero at node start-up.		

A.3.3.1.2 Extensions to [IEEE 802.15.4] clause 7.1: additional QualityOfService parameter

As shown in Table A.2, the Quality of Service (QOS) parameter defines the level of priority assigned to the MSDU to be transmitted. The G3 Annex C defines the priority mechanism of the G3-PLC.

Table A.2 – QualityOfService parameter definition

Name	Type	Valid range	Description
QualityOfService	Integer	0x00 – 0x02	The QOS (Quality of Service) parameter of the MSDU to be transmitted by the MAC sublayer entity. This value can take one of the following values: 0 = Normal priority, 1 = High priority, 2 = Contention free

A.3.3.2 MAC frame formats (based on [IEEE 802.15.4] clause 7.2)

A.3.3.2.1 Selections from [IEEE 802.15.4] clause 7.2: MAC frame formats

The MAC frame formats as described in clause 7.2 of [IEEE 802.15.4] apply, with the selections specified in Table A.3.

Table A.3 – Selections from clause 7.2 of [IEEE 802.15.4]

Clause	Title & remarks/modifications	Statement
7.2	MAC frame formats	N
7.2.1	General MAC frame format – Segment Control fields are added to MHR (see A.3.3.2.2) – Detailed description of the Segment Control fields is shown in Table A.5.	E
7.2.1.1	Frame control field NOTE – the Ack Request field must be set with a value consistent with the PHY layer DT field.	N
7.2.1.1.1	Frame type subfield – The present specification does not use acknowledgement frame type value. – The detailed ACK implementation is described in G3 Annex E of the present document. An acknowledgment can be sent by invoking the PD-ACK.request primitive (see [ITU-T G.9955] Annex A).	S
7.2.1.1.2	Security Enabled subfield	N
7.2.1.1.3	Frame Pending subfield – Indirect transmission is not supported, so this bit shall be set to 0.	S
7.2.1.1.4	Acknowledgement Request subfield – The present specification translates the Acknowledgment Request subfield to the proper delimiter type of frame control header. – The detailed ACK implementation is described in G3 Annex E of the present document. An acknowledgement can be sent by invoking the PD-ACK.request primitive (see [ITU-T G.9955] Annex A).	S
7.2.1.1.5	PAN ID compression subfield	N
7.2.1.1.6	Destination Addressing Mode subfield	N
7.2.1.1.7	Frame Version subfield – These 2 bits are reserved for future use. In this version of the specification they shall be set to 0.	S
7.2.1.1.8	Source Addressing Mode subfield	N
7.2.1.2	Sequence Number field	N
7.2.1.3	Destination PAN Identifier field	N
7.2.1.4	Destination Address field	N
7.2.1.5	Source PAN Identifier field	N
7.2.1.6	Source Address field	N
7.2.1.7	Auxiliary Security Header field – Possible lengths for the Auxiliary Security Header are 0 and 6 bytes (see clause A.4)	S
7.2.1.8	Frame Payload field	N
7.2.1.9	FCS field	N
7.2.2	Format of individual frame types	N
7.2.2.1	Beacon frame format	N
7.2.2.1.1	Beacon frame MHR fields	N
7.2.2.1.2	Superframe Specification field	S

Table A.3 – Selections from clause 7.2 of [IEEE 802.15.4]

Clause	Title & remarks/modifications	Statement
	– Beacons are not transmitted at regular time intervals (beaconless network). Therefore the Beacon Order parameter of the Superframe Specification field is not used and shall be set to 0. – The receiver is active all the time when not transmitting. Therefore the Superframe Order parameter of the Superframe Specification field is not used and shall be set to 0. – No superframe structure is used for communication, so the Final CAP Slot parameter of the Superframe Specification field is not used and shall be set to 0. – Devices will not be operating on batteries, so the Battery Life Extension subfield of the Superframe Specification field is not used and shall be set to 0. – Within the framework of the present standard, the association is performed by the 6LoWPAN Bootstrap Protocol in the upper layer, so the Association Permit parameter of the Superframe Specification field is meaningless here, and shall be set to 1. If another profile is used, this field shall be set as described in clause 7.2.2.1.2 of [IEEE 802.15.4].	
7.2.2.1.3	GTS Specification field – The GTS Descriptor Count shall be set to 0 (GTS are not supported). – The PAN coordinator never accepts GTS request, therefore the GTS Permit parameter of the GTS Specification field shall be set to 0.	S
7.2.2.1.4	GTS Direction field – The GTS feature is not used, and the GTS Direction field shall not be present in the frame	N/R
7.2.2.1.5	GTS List field – The GTS feature is not used, and considering the values of the GTS specification field described in clause 7.2.2.1.3 of the [IEEE 802.15.4], this list shall be empty	N/R
7.2.2.1.6	Pending Address specification field – Indirect transmission is not supported in this specification. Consequently, the Number of Short Addresses Pending is always 0, and Number of Extended Addresses Pending is also 0.	S
7.2.2.1.7	Address List field – Indirect transmission is not used, and this field shall not be present in beacons.	N/R
7.2.2.1.8	Beacon Payload field – The Beacon payload field is comprised of a one byte estimate of the route cost to the coordinator (RC_COORD). The route cost shall be based on the route cost calculation in LOAD. RC_COORD may be approximated by saving the lowest route cost extracted from the PREP, RREQ or RREP packets that originated from the PAN coordinator. If a device has failed to communicate with the PAN coordinator it shall set RC_COORD to its maximum value of 0xFF. A device shall initialize RC_COORD to 0x7F on association. The PAN coordinator shall set its RC_COORD to 0x00.	S,E
7.2.2.2	Data frame format	N
7.2.2.2.1	Data frame MHR fields	N

Table A.3 – Selections from clause 7.2 of [IEEE 802.15.4]

Clause	Title & remarks/modifications	Statement
7.2.2.2.2	Data payload field	N
7.2.2.3	Acknowledgement frame format – Acknowledgement frame format described in clause 7.2.2.3 of [IEEE 802.15.4] is not relevant. – The detailed ACK implementation is described in G3 Annex E of the present document. An acknowledgement can be sent by invoking the PD-ACK.request primitive (see [ITU-T G.9955] Annex A).	S
7.2.2.4	MAC command frame format	N
7.2.2.4.1	MAC command frame MHR fields	N
7.2.2.4.2	Command Frame Identifier field	N
7.2.2.4.3	Command Payload field	N
7.2.3	Frame compatibility – The use of the Frame Version subfield is reserved.	N/R

A.3.3.2.2 Extensions to [IEEE 802.15.4] clause 7.2: MAC frame formats

Table A.4 and Table A.5 define the Segment Control field added in the MAC Header (MHR) specified in [IEEE 802.15.4] clause 7.2.

Table A.4 – General MAC frame format

Octets: 3	2	1	0/2	0/2/8	0/2	0/2/8	0/6	Variable	2
Segment Control	Frame Control	Sequence Number	Destination PAN	Destination Address	Source PAN	Source Address	Auxiliary Security Header	Frame payload	FCS
MHR								MAC payload	MFR

Table A.5 – Segment control fields

Field	Byte	Bit number	Bits	Definition
RES	0	7-4	4	Reserved
TMR	0	3	1	Tone map request 1: Tone map is requested 0: Tone map is not requested
CC	0	2	1	Contention Control: 0: contention is allowed in next contention state 1: contention free access
CAP	0	1	1	Channel access priority: 0: Normal 1: High
LSF	0	0	1	Last Segment Flag 0: Not last segment 1: Last segment

Field	Byte	Bit number	Bits	Definition
SC	1	7-2	6	Segment Count
SL[9-8]	1	1-0	2	Segment Length of MAC frame
SL[7-0]	2	7-0	8	Segment Length of MAC frame

A.3.3.3 MAC command frames (based on [IEEE 802.15.4] clause 7.3)

A.3.3.3.1 Selections from [IEEE 802.15.4] clause 7.3: MAC command frames

The MAC frame formats as described in clause 7.3 of [IEEE 802.15.4] apply, with the selections specified in Table A.6.

Table A.6 – Selections from clause 7.3 of [IEEE 802.15.4]

Clause	Title & remarks/modifications	Statement
7.3	MAC command frames – All devices are Full Function Devices – The supported command list is defined in A.3.3.3.2.1	S, E
7.3.1	Association request command – Within the framework of the present standard, association is performed by the 6LoWPAN Bootstrap protocol described in clause A.3.4.5.2.2 of the present document, so the clause 7.3.1 of [IEEE 802.15.4] is not relevant.	N/R
7.3.2	Association response command – Within the framework of the present standard, association is performed by the 6LoWPAN Bootstrap protocol described in clause A.3.4.5.2.2 of the present document, so the clause 7.3.2 of [IEEE 802.15.4] is not relevant.	N/R
7.3.3	Disassociation Notification command – Within the framework of the present standard, association is performed by the 6LoWPAN Bootstrap protocol described in clause A.3.4.5.2.2 of the present document, so the clause 7.3.2 of [IEEE 802.15.4] is not relevant.	N/R
7.3.4	Data Request command	N/R
7.3.5	PAN ID Conflict Notification command – PAN ID Conflict Notification is performed by the adaptation layer, see A.3.5.2.	N/R
7.3.6	Orphan notification command – Orphan notification is not used in the present specification	N/R
7.3.7	Beacon request command – This command shall be implemented in every device	S
7.3.8	Coordinator realignment command – The coordinator realignment command is not used in the present notification	N/R
7.3.9	GTS request command – GTS are not used in the present specification	N/R

A.3.3.3.2 Extensions to [IEEE 802.15.4] clause 7.3: MAC command frames

A.3.3.3.2.1 MAC command frames supported

The present standard supports the MAC command frames described in Table A.7:

Table A.7 – MAC command frames

Command frame identifier	Command name	Clause
0x00-0x06	Reserved by ITU-T	–
0x07	Beacon request	See clause 7.3.7 of [IEEE 802.15.4]
0x08-0x09	Reserved by ITU-T	–
0x0A	Tone map response	See clause A.3.3.3.2.2
0x0B-0xFF	Reserved by ITU-T	–

A.3.3.3.2.2 The tone map response

The MAC sublayer generates Tone Map Response command if Tone Map Request (TMR) bit of received packet Segment Control field is set. It means that a packet originator requested tone map information from destination device. The destination device has to estimate this particular communication link between two points and choose optimal PHY parameters. The Tone Map Response contains the number of used tones and allocation (Tone Map), modulation mode and TX power control parameters. The Tone Map Response command frame shall be formatted as illustrated in Table A.8.

The channel estimation response command frame shall be formatted as illustrated in Table A.8:

Table A.8 – Tone map response format

Octets: (see clause 7.2.2.4 of [IEEE 802.15.4])	1	7 (for CENELEC A band) 15 (for FCC band)	2
MHR fields	Command frame identifier (see Table A.9)	Tone Map response payload (see Table A.9)	MFR Fields

The Tone Map Response message parameters are shown in Table A.9 for the case of Cenelec A and in Table A.10 for the case of FCC.

Table A.9 – Tone map response message description for CENELEC A band

Field	Byte	Bit number	Bits	Definition
TXRES	0	7	1	Tx Gain resolution corresponding to one gain step. 0: 6 dB 1: 3 dB
TXGAIN	0	6-3	4	Desired Transmitter gain specifying how many gain steps are requested.
MOD	0	2-1	2	Modulation type: 0 – Robust Mode 1 – DBPSK 2 – DQPSK

Field	Byte	Bit number	Bits	Definition
				3 – D8PSK
TM[8]	0	0	1	Tone Map [8]
TM[0:7]	1	7-0	8	Tone Map [0:7]
LQI	2	7-0	8	Link Quality Indicator
TXCOEF[3:0]	3	7-4	4	Specifies number of gain steps requested for the tones represented by TM[0] (optional)
TXCOEF[7:4]	3	3-0	4	Specifies number of gain steps requested for the tones represented by TM[1] (optional)
TXCOEF[11:8]	4	7-4	4	Specifies number of gain steps requested for the tones represented by TM[2] (optional)
TXCOEF[15:12]	4	3-0	4	Specifies number of gain steps requested for the tones represented by TM[3] (optional)
TXCOEF[19:16]	5	7-4	4	Specifies number of gain steps requested for the tones represented by TM[4] (optional)
TXCOEF[23:20]	5	3-0	4	Specifies number of gain steps requested for the tones represented by TM[5] (optional)
Reserved by ITU-T	6	7-0	8	Shall be set to zero
NOTE – As also mentioned in clause A.5.5/ [ITU-T G.9955], TM[8] to TM[6] are set to zero and are not used in CENELEC A band.				

Table A.10 – Tone map response message description for FCC band

Field	Byte	Bit number	Bits	Definition
TXRES	0	7	1	Tx Gain resolution corresponding to one gain step. 0: 6 dB 1: 3 dB
TXGAIN	0	6-3	4	Desired Transmitter gain specifying how many gain steps are requested.
MOD	0	2-0	3	Modulation type: 0 – Robust Mode 1 – DBPSK 2 – DQPSK 3 – D8PSK 4 – 16-QAM (Note 1) 5-7: reserved
TM[0:7]	1	7-0	8	Tone Map [0:7]
TM[8:15]	2	7-0	8	Tone Map [8:15]
TM[16:23]	3	7-0	8	Tone Map [16:23]
LQI	4	7-0	8	Link Quality Indicator
TXCOEF[1:0]	5	7-6	2	Specifies number of gain steps requested for the tones represented by TM[0] (optional)
TXCOEF[3:2]	6	5-4	2	Specifies number of gain steps requested for the tones

Table A.10 – Tone map response message description for FCC band

Field	Byte	Bit number	Bits	Definition
				represented by TM[1] (optional)
TXCOEF[5:4]	6	3-2	2	Specifies number of gain steps requested for the tones represented by TM[2] (optional)
TXCOEF[7:6]	6	1-0	2	Specifies number of gain steps requested for the tones represented by TM[3] (optional)
….	…	…	…	…
TXCOEF[47:46]	10	1-0	2	Specifies number of gain steps requested for the tones represented by TM[23] (optional)
Reserved	11	8	8	Reserved
NOTE 1 – The coherent mode specified in Annex E/G.9955 is optional.				

Where:

- MOD: Parameter that specifies the desired modulation type. The receiver computes the SNR of the *Tone Map Request* message that it receives from the transmitter and it decides which of the four modulation modes (DBPSK, DQPSK, D8PSK or Robust Mode) it wants the transmitter to use when sending next data frame. Table A.11 (parts a and b) list the allowed bit values and the modulation modes they correspond to;

Table A.11a – Modulation method field for CENELEC A band

MOD Value	Interpretation
00	Robust Modulation
01	DBPSK Modulation
10	DQPSK Modulation
11	D8PSK Modulation

Table A.11b – Modulation method field for FCC band

MOD Value	Interpretation
000	Robust Modulation
001	DBPSK Modulation
010	DQPSK Modulation
011	D8PSK Modulation
100-111	Reserved by ITU-T

- TXRES: Parameter that specifies the transmit gain resolution corresponding to one gain step;
- TXGAIN: Parameter that specifies to the transmitter the total amount of gain that it shall apply to its transmitted signal. The value in this parameter shall specify the total number of gain steps needed. The receiver computes the received signal level and compares it to a VTARGET (pre-defined desired receive level). The difference in dB between the two values is mapped to a 4-bit value that specifies the amount of gain increase or decrease that the transmitter shall apply to the next frame to be transmitted. A "0" in the most significant

bit indicates a positive gain value, hence an increase in the transmitter gain and a 1 indicates a negative gain value, hence a decrease in the transmitter gains. A value of TXGAIN = 0 informs the transmitter to use the same gain value it used for previous frame (Default value);

- TM: Parameter that specifies the Tone Map. The receiver estimates the per-tone quality of the channel and maps each sub-band (6 tones per sub-band) to a one-bit value where a value of 0 indicates to the remote transmitter that dummy data shall be transmitted on the corresponding subcarrier while a value of "1" indicates that valid data shall be transmitted on the corresponding subcarrier;
- TXCOEF (optional): Parameter that specifies transmitter gain for each group of tones represented by one valid bit of the tone map. The receiver measures the frequency-dependent attenuation of the channel and may request the transmitter to compensate for this attenuation by increasing the transmit power on sections of the spectrum that are experiencing attenuation in order to equalize the received signal. Each group of tones is mapped to a 4-bit value for CENELEC A or a 2-bit value for FCC where a "0" in the most significant bit indicates a positive gain value, hence an increase in the transmitter gain scaled by TXRES is requested for that section, and a "1" indicates a negative gain value, hence a decrease in the transmitter gain scaled by TXRES is requested for that section. Implementing this feature is optional and it is intended for frequency selective channels. If this feature is not implemented, the value zero shall be used.
- The LQI value is computed in the PHY and passed to the MAC with the PD-DATA.indication primitive through the ppduLinkQuality parameter – see Table A.16/ [ITU-T G.9955].

On reception of a Tone Map Response command frame, the MAC sublayer updates the neighbour table with the corresponding Tone Map and communication parameters for that device. If no entry already exists in the table for that device a new entry may be added, based on implementation dependant limitations. The neighbour table is defined in Table A.17.

The following procedure shall be used to perform the adaptive tone mapping function:

a) When a station is ready to transmit data it will first check if the neighbour table already has a record related to the destination device address. If the record does not exist or is aged (Age counter is 0), the MAC sublayer sets the TMR bit of outgoing packet Segment Control field and requests new Tone Map information. In this case the MAC data shall be sent in Robust Mode;

   If a neighbour table record exists and it is not aged the MAC sublayer does not need to send Tone Map Request message. In this case, the MAC sublayer uses information from the neighbour table to properly configure physical TX in transmitting mode and construct Frame Control Header (FCH) of the outgoing frame;

   When the destination station receives a data frame it shall check the Tone Map Request bit in the Segment Control field. If the bit is set, the destination station shall measure the per-carrier quality of the channel, construct and send a Tone Map Response message back to the originator station. The destination station shall not send a Tone Map Response message if the Tone Map Request bit is not set. The Tone Map Response message shall always be transmitted using default Robust modulation. The destination device uses parameters from the Frame Control Header to decode the MAC data fields;

   The destination station shall attempt to send a Tone Map Response message as soon as possible after receiving a Tone Map Request message from the source station;

   If the source station receives a Tone Map Response message, it will update a neighbour table record related to the destination address with new Tone Map, modulation and TX gain parameters. If the record does not exist, the MAC sublayer will create a new one. The Age counter shall be set to defined value (see A.3.3.4). After receiving a Tone Map Response message, a device shall begin to use the updated neighbour table information for all transmissions to the associated destination until the Age counter will reach the value "0";

If the source station does not receive a Tone Map Response message after transmitting a Tone Map Request message to a certain destination, it shall set the Tone Map Request bit in the Segment Control of the next MAC data frame that it wants to transmit to the same destination. In other words, the MAC sublayer will continue to transmit a Tone Map Request message to the same destination;

The MAC sublayer shall not send a Tone Map Request message to the destination device if no data sent to this device.

The Tone Map request/response message sequence chart is shown in A.3.3.7.2.4.

A.3.3.4 MAC constants and PIB attributes (based on [IEEE 802.15.4] clause 7.4)

A.3.3.4.1 Selections from [IEEE 802.15.4] clause 7.4: MAC constants and PIB attributes

The MAC frame formats as described in clause 7.4 of [IEEE 802.15.4] apply, with the selections specified in Table A.12.

Table A.12 – Selections from clause 7.4 of [IEEE 802.15.4]

Clause		Title & remarks/modifications	Statement
7.4		MAC constants and PIB attributes	N
7.4.1		MAC constants – The aBaseSlotDuration parameter is not used and shall be set to 0. – The aBaseSuperframeDuration parameter is not used and shall be set to 0. – The aExtendedAddress parameter shall be equal to the EUI-48 address of the device mapped to a EUI-64 address. – The aGTSDescPersistenceTime parameter is not used and shall be set to 0. – The aMaxBeaconOverhead parameter shall be set to 0. – The aMaxBeaconPayloadLength parameter is not used and shall be set to. – The aMaxLostBeacons parameter is not used and shall be set to 0. – The aMaxMACSafePayloadSize parameter is not used and shall be set to 0. – The aMaxMACPayloadSize parameter is fixed to 400 bytes by the present standard. – The aMaxMPDUUnsecuredOverhead parameter is not used and shall be set to 0. – The aMaxSIFSFrameSize parameter is not used and shall be set to 0. – The aMinCAPLength parameter is not used and shall be set to 0. – The aMinMPDUOverhead parameter is not used and shall be set to 0. – The aNumSuperframeSlots parameter is not used and shall be set to 0. – The aUnitBackoffPeriod parameter shall be set to aSlotTime. – Extensions: Additional MAC sublayer constants are defined in A.3.3.4.2.1.	S, E

Table A.12 – Selections from clause 7.4 of [IEEE 802.15.4]

Clause		Title & remarks/modifications	Statement
7.4.2		MAC PIB attributes – The macAckWaitDuration parameter shall be set according to the following formula: macAckWaitDuration = aRIFS + aAckTime + aCIFS – The macAssociatedPANCoord parameter is not used and shall be set to FALSE. – The macAssociationPermit parameter is not used and shall be set to TRUE. – The macAutoRequest parameter is not used and shall be set to FALSE. – The macBattLifeExt parameter is not used; changing it has no effect on the behaviour of the device. Its default value shall be FALSE. – The macBattLifeExtPeriods parameter is not used; changing it has no effect on the behaviour of the device. Its default value shall be 0. – The macBeaconPayload parameter is not used; changing it has no effect on the behaviour of the device. Its default value shall be NULL. – The macBeaconPayloadLength parameter is not used and shall be set to 0. – The macBeaconOrder parameter is not used; changing it has no effect on the behaviour of the device. Its default value shall be left to 15. – When the macBeaconTxTime parameter reaches 0xFFFFFF, it shall not change anymore. – The macGTSPermit parameter is not used; changing it has no effect on the behaviour of the device. Its default value shall be FALSE. – The macMaxBE parameter is fixed to 5 by the present standard. – The macMaxCSMABackoffs default value is fixed to 8 by the present standard. – The macMaxFrameTotalWaitTime parameter is not used shall be set to 0. – The macMinBE parameter is fixed to 3 by the present standard. – The macMinLIFSPeriod parameter is not used; changing it has no effect on the behaviour of the device. – The macMinSIFSPeriod parameter is not used; changing it has no effect on the behaviour of the device. – The macResponseWaitTime parameter shall be set to macAckWaitDuration. – The macRxOnWhenIdle parameter shall be set to TRUE. – The macSecurityEnabled parameter shall be set to TRUE. – The macShortAddress parameter shall be equal to 0xFFFF when the device does not have a short address. An associated device necessarily has a short address, so that a device cannot be in the state where it is associated but does not have a short address. – The macSuperframeOrder parameter is not used, and shall be left to 15. – The macSyncSymbolOffset is not used and shall be set to 0. – The macTimestampSupported parameter shall be set to TRUE. – The macTransactionPersistenceTime parameter is not used and shall be set to 0. – Extensions: Additional set of IB attributes are defined in A.3.3.4.2.	S, E

A.3.3.4.2 Extensions to [IEEE 802.15.4] clause 7.4: MAC constants and PIB attributes

A.3.3.4.2.1 Additional MAC sublayer constants to [IEEE 802.15.4] clause 7.4.1

Table A.13 defines the list of MAC sublayer constants added by the present standard:

Table A.13 – Additional MAC sublayer constants to [IEEE 802.15.4] clause 7.4.1

Constant	Description	Value
aPreamSymbolTime	Defines the duration of one preamble symbol on physical layer (in microseconds).	640
aSymbolTime	Defines the duration of one data symbol on physical layer (in microseconds).	695
aSlotTime	The duration of the contention slot time (in data symbols)	2
aCIFS	Defines the Contention InterFrame Space (number of data symbols). It is defined in G3 Annex C of the present document.	8 for CENELEC A 10 for FCC
aRIFS	Defines the response interframe space (number of data symbols). It is defined in G3 Annex C of the present document.	8 for CENELEC A 10 for FCC
aEIFS	Defines the duration of Extended InterFrame space. It is defined in G3 Annex C of the present document.	aSymbolTime * (aMaxFrameSize + aRIFS + aCIFS) + aAckTime
aMinFrameSize	Defines the minimum MAC frame size in data symbols.	4
aMaxFrameSize	Defines the maximum MAC frame size in data symbols.	252
aAckTime	Defines the duration of acknowledgment: N_{PRE} – number of preamble symbols is defined in clause A.5.2/ [ITU-T G.9955] N_{FCH} – number of FCH symbols is defined in clause A.5.2/ [ITU-T G.9955]	N_{PRE} * aPremSymbolTime + N_{FCH} *aSymbolTime

A.3.3.4.2.2 Additional MAC sublayer attributes to [IEEE 802.15.4] clause 7.4.2

Table A.14 defines the list of MAC sublayer attributes added by the present standard:

Table A.14 – Additional attributes to [IEEE 802.15.4] clause 7.4.2

Attribute	Identifier	Type	Range	Description	Default value
macHighPrioirtyWindow Size	0x01000113	Unsigned Integer	1-7	The high priority contention window size in number of slots. Default value is 7*aSlotTime	7
macTxDataPacketCount	0x02000101	Unsigned Integer	0 – 4 294 967 295	Statistic counter of successfully transmitted MSDUs	0

Table A.14 – Additional attributes to [IEEE 802.15.4] clause 7.4.2

Attribute	Identifier	Type	Range	Description	Default value
macRxDataPacketCount	0x02000102	Unsigned Integer	0 – 4 294 967 295	Statistic counter of successfully received MSDUs	0
macTxCmdPacketCount	0x02000201	Unsigned Integer	0 – 4 294 967295	Statistic counter of successfully transmitted command packets	0
macRxCmdPacketCount	0x02000202	Unsigned Integer	0 – 4 294 967 295	Statistic counter of successfully received command packets	0
macCSMAFailCount	0x02000103	Unsigned Integer	0 – 4 294 967 295	Statistic counter of failed CSMA transmit attempts	0
macCSMACollisionCount	0x02000104	Unsigned Integer	0 – 4 294 967 295	Statistic counter of collision due to channel busy or failed transmission	0
macBroadcastCount	0x02000106	Unsigned Integer	0 – 4 294 967 295	Statistic counter of the number of broadcast frames sent	0
macMulticastCount	0x02000107	Unsigned Integer	0 – 4 294 967 295	Statistic counter of the number of multicast frames sent	0
macBadCRCCount	0x02000108	Unsigned Integer	0 – 4 294 967 295	Statistic counter of the number of frames received with bad CRC	0
macMaxOrphanTimer	0x02000109	Unsigned Integer	0 – 4 294 967 295	The maximum number of seconds without communication with a particular device after which it is declared as an orphan.	0
macNeighbourTable	0x0000006B	Set	–	The neighbour table defined in A.3.3.5.2.	–
macNumberOfHops	0x0000006C	Unsigned Integer	0 – 14	The number of hops to reach the PAN coordinator.	8

Table A.14 – Additional attributes to [IEEE 802.15.4] clause 7.4.2

Attribute	Identifier	Type	Range	Description	Default value
macFreqNotching	0x00006D	Bool	FALSE TRUE	S-FSK 63 and 74 kHz frequency notching. Default value is FALSE (disabled)	FALSE
macCSMAFairnessLimit	0x02000112	Unsigned integer	0-255	Specifies how many failed backoff attempts, backoff exponent is set to minBE	15
macMaxAgeTime	0x02000113	Unsigned Integer	0 – 255	Maximum life time of a device in neighbour table in minutes before sending a new Tone Map request	2
macMaxNeighborValidTime	0x02000114	Unsigned Integer	0 – 255	Maximum time of validity for an entry in neighbour table in minutes	255
macRCCoord	0x02000115	Unsigned Integer	0 – 255	Route Cost to Coordinator to be used in the beacon payload as RC_COORD	255

A.3.3.4.2.3 MAC sublayer attributes and their associated ID

Table A.15 defines the new identifier associated to [IEEE 802.15.4] MAC sublayer attributes used by the present standard:

Table A.15 – MAC sublayer attributes and their associated ID

Attribute	Identifier[1]	Type	Range	Description	Default value
macAckWaitDuration	0x01000103	Integer	0x0 – 0xFFFF	Duration of acknowledgement in microseconds	aSymbolTime*(aRIFS + aCIFS)+ aAckTime
macBSN	0x01000105	Integer	0x0 – 0xFF	Beacon frame sequence number	random
macCoordExtended Address	0x01000106	Set of 8 bytes	–	Coordinator extended address	0x0
macCoordShortAddress	0x01000107	Integer	0x0 – 0xFFFF	Coordinator short address	0x0
macDSN	0x01000108	Integer	0x0 – 0xFF	Data frame sequence number	random

Table A.15 – MAC sublayer attributes and their associated ID

Attribute	Identifier[1]	Type	Range	Description	Default value
macMaxBE	0x0100010A	Integer	0 – 20	Maximum value of backoff Exponent. It should always be > macMinBE	8
macMaxCSMABackoffs	0x0100010B	Integer	0 – 0xFF	Maximum number of backoff attempts	50
macMaxFrameRetries	0x0100010D	Integer	0 – 10	Maximum number of retransmission	5
macMinBE	0x0100010E	Integer	0 – 20	Minimum value of backoff exponent	3
macPanId	0x0100010F	Integer	0x0 – 0xFFFF	PAN Id	0xFFFF
macResponseWaitTime	0x01000110	Integer	0x0 – 0xFFFF	Response waiting time in microseconds, set to macAckWaitDuration	aSymbolTime*(aRIFS + aCIFS)+ aAckTime
macSecurityEnabled	0x01000111	Boolean	–	Security enabled	TRUE
macShortAddress	0x01000112	Integer	0x0 – 0xFFFF	Device short address	0xFFFF
macPromiscuousMode	0x01000115	Boolean	–	Promiscuous mode enabled	FALSE
macTimeStampSupport	0x0000005C	Boolean	–	MAC frame time stamp support enable	TRUE

[1] These are new identifiers associated to the IEEE 802.15.5 MAC sublayer attributes that are used by this specification.

A.3.3.5 MAC functional description (based on [IEEE 802.15.4] clause 7.5)

A.3.3.5.1 Selections from [IEEE 802.15.4] clause 7.5: MAC functional description

The MAC functional description as described in clause 7.5 of [IEEE 802.15.4] applies, with the selections specified in Table A.16.

Table A.16 – Selections from clause 7.5 of [IEEE 802.15.4]

Clause		Title & remarks/modifications	Statement
7.5		MAC functional description – beacon-enabled PAN and GTS are not supported – GTS contention free access is not supported	S
7.5.1		Channel access – See G3 Annex C of the present document for channel access functional description.	E
7.5.1.1		Superframe structure	N/R
7.5.1.2		Incoming and outgoing frame structure	N/R

Table A.16 – Selections from clause 7.5 of [IEEE 802.15.4]

Clause	Title & remarks/modifications	Statement
7.5.1.3	Interframe (IFS) spacing – See G3 Annex C of the present document for Interframe spacing description.	E
7.5.1.4	CSMA-CA algorithm – See G3 Annex C of the present document for description of CSMA-CA algorithm (including priority, ARQ, segmentation and reassembly overview).	E
7.5.2	Starting and maintaining PANs	N
7.5.2.1	Scanning through channels – Passive scanning is not supported – Orphan scanning is not supported – ED scanning is not supported – Active scanning is the only supported scanning mode – As there is no channel page or channel list notion at the physical level, a scan request is agnostic to a physical channel.	S
7.5.2.1.1	ED channel scan – ED channel scan is not supported by the present standard	N/R
7.5.2.1.2	Active channel scan – Active channel scan is only used by an un-associated device prior to starting association and by the PAN coordinator prior to starting a new network. – As there is no channel page or channel list notion at the physical level, a scan request does not care about a particular channel.	S
7.5.2.1.3	Passive channel scan – Passive channel scan is not supported by the present standard	N/R
7.5.2.1.4	Orphan channel scan – Orphan channel scan is not supported by the present standard	N/R
7.5.2.2	PAN identifier conflict resolution PAN conflict handling is as described in A.3.5.2 of the present document.	N/R
7.5.2.2.1	Detection – PAN conflict detection is performed by scanning all incoming PAN Id of frames received by the devices as described in A.3.5.2 of the present document.	N/R
7.5.2.2.2	Resolution – On detection of a PAN identifier conflict, a device shall generate a CONFLICT frame as described in A.3.5.2 of the present document.	N/R
7.5.2.3	Starting and realigning a PAN	N
7.5.2.3.1	Starting a PAN – A PAN coordinator cannot lose its MAC address. It can however be changed based on criteria out of the scope of this standard, for example in case of PAN ID conflict detection.	S
7.5.2.3.2	Realigning a PAN – PAN realignment is not supported by the present specification.	N/R

Table A.16 – Selections from clause 7.5 of [IEEE 802.15.4]

Clause	Title & remarks/modifications	Statement
7.5.2.3.3	Realignment in a PAN – PAN realignment is not supported by the present specification.	N/R
7.5.2.3.4	Updating superframe configuration and channel PIB attributes – The macBeaconOrder parameter shall be set to 15 to have a beaconless PAN. – The phyCurrentPage and phyCurrentChannel parameters are not used, and shall be set to 0.	S
7.5.2.4	Beacon generation – Only non beacon-enabled PAN are used – Beacon shall be transmitted using the Robust modulation	S
7.5.2.5	Device discovery – Device discovery is done using the active scanning procedure described in A.3.4.5.2.2.2, to force a coordinator to send a beacon.	E
7.5.3	Association and disassociation	N
7.5.3.1	Association – Association is fully described in A.3.4.5 of the present document.	N/R
7.5.3.2	Disassociation – Disassociation is fully described in A.3.4.5 of the present document.	N/R
7.5.4	Synchronization	N/R
7.5.4.1	Synchronization with beacons – Beacon synchronization is not used in the present standard.	N/R
7.5.4.2	Synchronization without beacons	N/R
7.5.4.3	Orphaned device realignment – Orphaned device realignment is not used in the present specification. – Orphaned device detection is performed at the application level using a timer, which is reset each time the device receives a frame with the Destination Address field of the MAC header equal to the MAC address (either short or extended) of the device. If this timer reaches its maximum value (macMaxOrphanTimer), then the device loses its short MAC address, and shall begin an association procedure.	S
7.5.5	Transaction handling – Transactions are not supported in the present standard.	N/R
7.5.6	Transmission, reception and acknowledgement	N
7.5.6.1	Transmission	N
7.5.6.2	Reception and rejection	N
7.5.6.3	Extracting pending data from a coordinator	N/R
7.5.6.4	Use of acknowledgements and retransmissions	N
7.5.6.4.1	No acknowledgement – The present standard defines an acknowledgement differently. The detailed ACK implementation is described in G3 Annex E.	E
7.5.6.4.2	Acknowledgement – The present standard defines an acknowledgement differently. The detailed ACK implementation is described in G3 Annex E.	E

Table A.16 – Selections from clause 7.5 of [IEEE 802.15.4]

Clause	Title & remarks/modifications	Statement
7.5.6.4.3	Retransmissions	N
7.5.6.5	Promiscuous mode	N
7.5.6.6	Transmission scenario	N
7.5.7	GTS allocation and management – GTS are not used in the present specification	N/R
7.5.8	Frame security	N
7.5.8.1	Security related MAC PIB attributes – Key Table contains two 16-octets keys. They represent current and preceding GMK as described in clause A.4.5.3. The KeyIndex parameter selects the actual key. – Device Table is not used – Security level table is not used – Automatic request attributes are not used – Default key source is not used	S
7.5.8.1.1	Key table – Key Table contains two 16-octets keys. They represent current and preceding GMK as described in clause A.4.5.3. The KeyIndex parameter selects the actual key.	S
7.5.8.1.2	Device table	N/R
7.5.8.1.3	Minimum security level table	N/R
7.5.8.1.4	Frame counter	N
7.5.8.1.5	Automatic request attributes	N/R
7.5.8.1.6	Default key source	N/R
7.5.8.1.7	PAN coordinator address	N/R
7.5.8.2	Functional description	N
7.5.8.2.1	Outgoing frame security procedure	N
7.5.8.2.2	Outgoing frame key retrieval procedure	N/R
7.5.8.2.3	Incoming frame security procedure – The KeyIndex parameter selects the actual key from Key Table.	S
7.5.8.2.4	Incoming frame security material retrieval procedure	N/R
7.5.8.2.5	KeyDescriptor lookup table	N/R
7.5.8.2.6	Blacklist checking procedure	N/R
7.5.8.2.7	DeviceDescriptor lookup procedure	N/R
7.5.8.2.8	Incoming security level checking procedure	N/R
7.5.8.2.9	Incoming key usage policy checking procedure	N/R

A.3.3.5.2 Extensions to [IEEE 802.15.4] clause 7.5: Neighbour table

Every device shall maintain a Neighbour Table, which contains information about all the devices within the POS of a device. Similar to [IEEE 802.15.4], the POS of an ITU-T G.9955/6 Annex A device is the reception range of an ITU-T G.9955/6 Annex A packet transmission. This table is actualized each time any frame is received from a neighbouring device, and each time a Tone Map

Response command is received. This table shall be accessible by the Adaptation, MAC sublayers and physical layer. Each entry of this table contains the fields listed in Table A.17:

Table A.17 – Neighbour table for Cenelec A

Field Name	Size/Type	Description
Short Address	16 bits	The MAC Short Address of the node which this entry refers to.
ToneMap	9 bits	The Tone Map parameter defines which frequency sub-band can be used for communication with the device. A bit set to 1 means that the frequency sub-band can be used, and a bit set to 0 means that frequency sub-band shall not be used.
Modulation	2 bits	Defines the modulation type to use for communicating with the device. 0x00: Robust Mode 0x01: DBPSK 0x02: DQPSK 0x03: D8PSK
TxGain	4 bits	Defines the Tx Gain to use to transmit frames to that device
TxRes	1 bit	Defines the Tx Gain resolution corresponding to one gain step. 0: 6 dB 1: 3 dB
TxCoeff	8 x 4 bits	The Tx gain for each 10 kHz-wide spectrum band
TXCOEF[3:0]	4 bits	Specifies number of gain steps requested for the tones represented by TM[0] (optional)
TXCOEF[7:4]	4 bits	Specifies number of gain steps requested for the tones represented by TM[1] (optional)
TXCOEF[11:8]	4 bits	Specifies number of gain steps requested for the tones represented by TM[2] (optional)
TXCOEF[15:12]	4 bits	Specifies number of gain steps requested for the tones represented by TM[3] (optional)
TXCOEF[19:16]	4 bits	Specifies number of gain steps requested for the tones represented by TM[4] (optional)
TXCOEF[23:20]	4 bits	Specifies number of gain steps requested for the tones represented by TM[5] (optional)
Reserved	8 bits	Reserved by ITU-T
LQI	8 bits	Link Quality Indicator
Age	8 bits	The remaining lifetime of the device in minutes. – When the entry is created, this value shall be set to the default value 0; – When it reaches 0, a Tone Map request may be issued if data is sent to this device. Upon successful reception of a tone map response, this value is set to macMaxAgeTime (see Table A.14).
IsNeighbour	8 bits	The remaining lifetime of the validity of this entry in the table in minutes. Every time an entry is created or a frame (data or ACK) is received from this neighbour, it is set to macMaxNeighborValidTime. When it reaches zero, this entry is no longer valid in the table and may be removed.
Reserved	8 bits	Reserved by ITU-T

Table A.17a – Neighbour table for FCC

Field Name	Size/Type	Description
Short Address	16 bits	The MAC Short Address of the node which this entry refers to.
ToneMap	24 bits	The Tone Map parameter defines which frequency sub-band can be used for communication with the device. A bit set to 1 means that the frequency sub-band can be used, and a bit set to 0 means that frequency sub-band shall not be used.
Modulation	3 bits	Defines the modulation type to use for communicating with the device. 0x00: Robust Mode 0x01: DBPSK 0x02: DQPSK 0x03: D8PSK 0x04: 16-QAM (Note 1) 0x05-0x07: reserved
TxGain	4 bits	Defines the Tx Gain to use to transmit frames to that device
TxRes	1 bit	Defines the Tx Gain resolution corresponding to one gain step. 0: 6 dB 1: 3 dB
TXCOEF[1:0]	2 bits	Specifies number of gain steps requested for the tones represented by TM[0] (optional)
TXCOEF[3:2]	2 bits	Specifies number of gain steps requested for the tones represented by TM[1] (optional)
TXCOEF[5:4]	2 bits	Specifies number of gain steps requested for the tones represented by TM[2] (optional)
TXCOEF[7:6]	2 bits	Specifies number of gain steps requested for the tones represented by TM[3] (optional)
TXCOEF[9:8]	2 bits	Specifies number of gain steps requested for the tones represented by TM[4] (optional)
TXCOEF[11:10]	2 bits	Specifies number of gain steps requested for the tones represented by TM[5] (optional)
TXCOEF[13:12]	2 bits	Specifies number of gain steps requested for the tones represented by TM[6] (optional)
TXCOEF[15:14]	2 bits	Specifies number of gain steps requested for the tones represented by TM[7] (optional)
TXCOEF[17:16]	2 bits	Specifies number of gain steps requested for the tones represented by TM[8] (optional)
TXCOEF[19:18]	2 bits	Specifies number of gain steps requested for the tones represented by TM[9] (optional)
TXCOEF[21:20]	2 bits	Specifies number of gain steps requested for the tones represented by TM[10] (optional)
TXCOEF[23:22]	2 bits	Specifies number of gain steps requested for the tones represented by TM[11] (optional)
TXCOEF[25:24]	2 bits	Specifies number of gain steps requested for the tones represented by TM[12] (optional)

Table A.17a – Neighbour table for FCC

Field Name	Size/Type	Description
TXCOEF[27:26]	2 bits	Specifies number of gain steps requested for the tones represented by TM[13] (optional)
TXCOEF[29:28]	2 bits	Specifies number of gain steps requested for the tones represented by TM[14] (optional)
TXCOEF[31:30]	2 bits	Specifies number of gain steps requested for the tones represented by TM[15] (optional)
TXCOEF[33:32]	2 bits	Specifies number of gain steps requested for the tones represented by TM[16] (optional)
TXCOEF[35:34]	2 bits	Specifies number of gain steps requested for the tones represented by TM[17] (optional)
TXCOEF[37:36	2 bits	Specifies number of gain steps requested for the tones represented by TM[18] (optional)
TXCOEF[39:38]	2 bits	Specifies number of gain steps requested for the tones represented by TM[19] (optional)
TXCOEF[41:40]	2 bits	Specifies number of gain steps requested for the tones represented by TM[20] (optional)
TXCOEF[43:42]	2 bits	Specifies number of gain steps requested for the tones represented by TM[21] (optional)
TXCOEF[45:44]	2 bits	Specifies number of gain steps requested for the tones represented by TM[22] (optional)
TXCOEF[47:46]	2 bits	Specifies number of gain steps requested for the tones represented by TM[23] (optional)
LQI	8 bits	Link Quality Indicator
Age	8 bits	The remaining lifetime of the device in minutes. – When the entry is created, this value shall be set to the default value 0; – When it reaches 0, a Tone Map request may be issued if data is sent to this device. Upon successful reception of a tone map response, this value is set to macMaxAgeTime (see Table A.14).
IsNeighbour	8 bits	The remaining lifetime of the validity of this entry in the table in minutes. Every time an entry is created or a frame (data or ACK) is received from this neighbour, it is set to macMaxNeighborValidTime. When it reaches zero, this entry is no longer valid in the table and may be removed.
Reserved	8 bits	Reserved for future development
NOTE 1 – The coherent mode specified in Annex D/ [ITU-T G.9955] is optional.		

If the device receives a frame whose source address field (MAC sublayer header) does not exist in the neighbour table, it shall add a new entry for that device with the following default values:

- Modulation = 0 (Robust Mode);
- ToneMap = (all bits set to 1) AND (*adpToneMask*);
- TxGain = 0b0000;
- TxCoeff = 0x0;
- LQI = 0;

- Age = 0.
- IsNeighbour = macMaxNeighborValidTime

The Neighbour Table is available in the Information Base under the attribute *macNeighbourTable* (see A.3.3.4.2.2).

A.3.3.6 MAC security suite specifications (selections from [IEEE 802.15.4] clause 7.6)

The security suite specifications as described in clause 7.6 of [IEEE 802.15.4] apply, with the selections specified in Table A.18.

Table A.18 – Selections from clause 7.6 of [IEEE 802.15.4]

Clause	Title & remarks/modifications	Statement
7.6	Security suite specification	N
7.6.1	PIB security material – Key Table contains two 16 octets keys. They represent current and preceding GMK as described in clause A.4.5.3. The KeyIndex parameter selects the actual key. – Automatic request attributes are not used – Default key source is not used – Device Table is not used – Security level table is not used	S
7.6.2	Auxiliary security header	N
7.6.2.1	Integer and octet representation	N
7.6.2.2	Security Control field	N
7.6.2.2.1	Security Level subfield – Two values are allowed by the present standard: 0x00 = "none", 0x05 = "ENC-MIC-32".	S
7.6.2.2.2	Key Identifier Mode subfield – One Key Identifier Mode is allowed by the present standard: 0x01 = "Key determined from the 1-octet Key Index subfield" The number of keys is limited to 2 (KeyIndex value is 0x0 – 0x1)	S
7.6.2.3	Frame Counter field	N
7.6.2.4	Key Identifier field	N
7.6.2.4.1	Key Source subfield	N/R
7.6.2.4.2	Key Index subfield – Key Index value is 0x0 – 0x1	N
7.6.3	Security operations	N
7.6.3.1	Integer and octet representation	N

Clause	Title & remarks/modifications	Statement
7.6.3.2	CCM* Nonce Nonce is formatted as follows, with the first field defining the most significant byte and the last the least significant byte: • PAN-ID (2 bytes) • Source Short Address (2 bytes) • PAN-ID (2 bytes) • Source Short Address (2 bytes) • Frame Counter (4 bytes) • Security Level (1 bytes) NOTE 1 – The encrypted frame shall contain the source short address and PAN-ID in the MAC header. NOTE 2 – Fields bigger than a single byte are used in the order from the byte containing the highest numbered bits to the byte containing the lowest numbered bits (Big Endian).	S, E
7.6.3.3	CCM* prerequisites	N
7.6.3.3.1	Authentication field length	N
7.6.3.4	CCM* transformation data representation	N
7.6.3.4.1	Key and nonce data inputs	N
7.6.3.4.2	a data and m data – Two values are allowed by the present standard: 0x00 = "none", 0x05 = "ENC-MIC-32".	S
7.6.3.4.3	c data output – Two values are allowed by the present standard: 0x00 = "none", 0x05 = "ENC-MIC-32".	S
7.6.3.5	CCM* inverse transformation data representation	N
7.6.3.5.1	Key and nonce data inputs	N
7.6.3.5.2	c data and a data	N
7.6.3.5.3	m data output	N

A.3.3.7 Message sequence chart illustrating MAC – PHY interaction (based on [IEEE 802.15.4] clause 7.7)

A.3.3.7.1 Selections from [IEEE 802.15.4] clause 7.7: Message sequence chart illustrating MAC

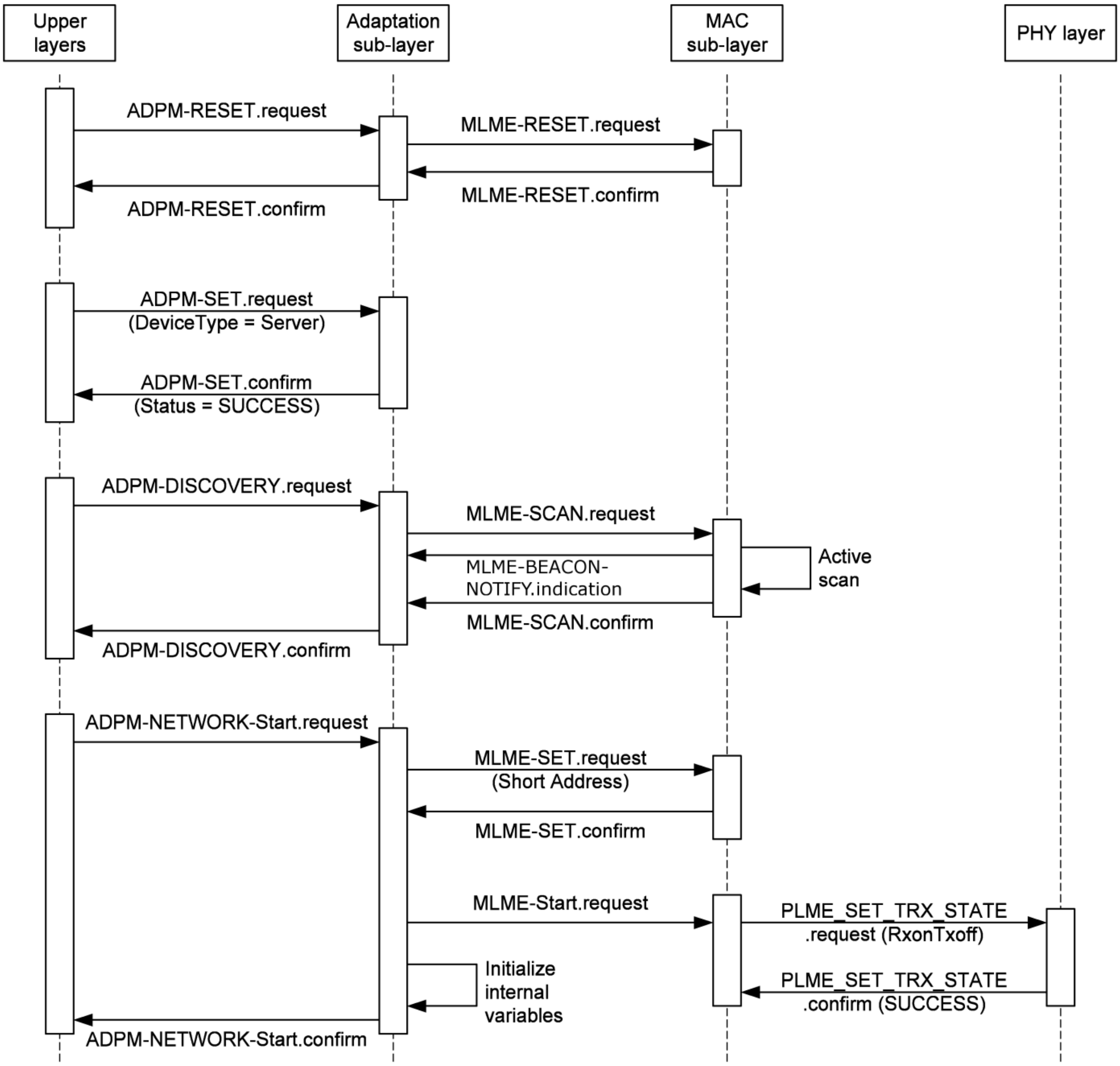
The message Sequence Chart Illustrating MAC – PHY interaction as described in clause 7.7 of [IEEE 802.15.4] apply, with the selections specified in Table A.19.

Table A.19 – Selections from clause 7.7 of [IEEE 802.15.4]

Clause	Title & remarks/modifications	Statement
7.7	Message sequence chart illustrating MAC-PHY interaction – Figure 78: Replaced by clause A.3.3.7.2.1 of this document – Figure 79: N/R – Figure 80: N/R – Figure 81: N/R – Figure 82: N/R – Figure 83: Replaced by clause A.3.3.7.2.2 of this document – Figure 84 & Figure 85: Replaced by clause A.3.3.7.2.3 of this document – Figure 86: N/R – Additional figure about channel estimation in clause A.3.3.7.2.4 of this document	S, E

A.3.3.7.2 Extensions to [IEEE 802.15.4], clause 7.7: Message sequence chart illustrating MAC

A.3.3.7.2.1 PAN start message sequence chart for PAN coordinators



Upper layers
Adaptation sub-layer
MAC sub-layer
PHY layer
ADPM-RESET.request
MLME-RESET.request
MLME-RESET.confirm
ADPM-RESET.confirm
ADPM-SET.request (DeviceType = Server)
ADPM-SET.confirm (Status = SUCCESS)
ADPM-DISCOVERY.request
MLME-SCAN.request
Active scan
MLME-BEACON-NOTIFY.indication
MLME-SCAN.confirm
ADPM-DISCOVERY.confirm
ADPM-NETWORK-Start.request
MLME-SET.request (Short Address)
MLME-SET.confirm
MLME-Start.request
PLME_SET_TRX_STATE .request (RxonTxoff)
Initialize internal variables
PLME_SET_TRX_STATE .confirm (SUCCESS)
ADPM-NETWORK-Start.confirm

Figure A.1 – PAN start message sequence chart

A.3.3.7.2.2 Active scan message sequence chart

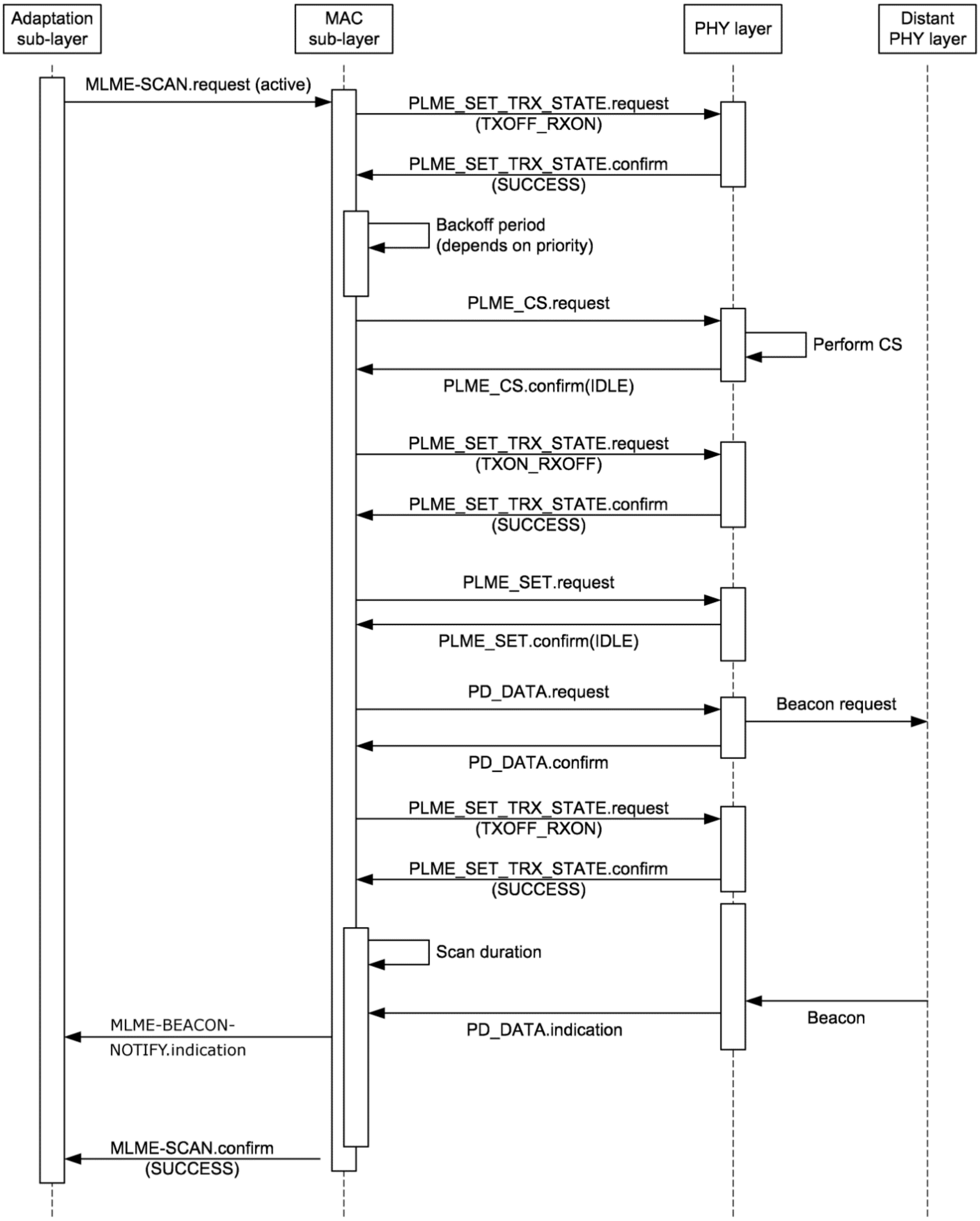

Adaptation sub-layer
MAC sub-layer
PHY layer
Distant PHY layer
MLME-SCAN.request (active)
PLME_SET_TRX_STATE.request (TXOFF_RXON)
PLME_SET_TRX_STATE.confirm (SUCCESS)
Backoff period (depends on priority)
PLME_CS.request
Perform CS
PLME_CS.confirm(IDLE)
PLME_SET_TRX_STATE.request (TXON_RXOFF)
PLME_SET_TRX_STATE.confirm (SUCCESS)
PLME_SET.request
PLME_SET.confirm(IDLE)
PD_DATA.request
Beacon request
PD_DATA.confirm
PLME_SET_TRX_STATE.request (TXOFF_RXON)
PLME_SET_TRX_STATE.confirm (SUCCESS)
Scan duration
Beacon
PD_DATA.indication
MLME-BEACON-NOTIFY.indication
MLME-SCAN.confirm (SUCCESS)

Figure A.2 – Active scan message sequence chart

A.3.3.7.2.3 Data transmission message sequence chart

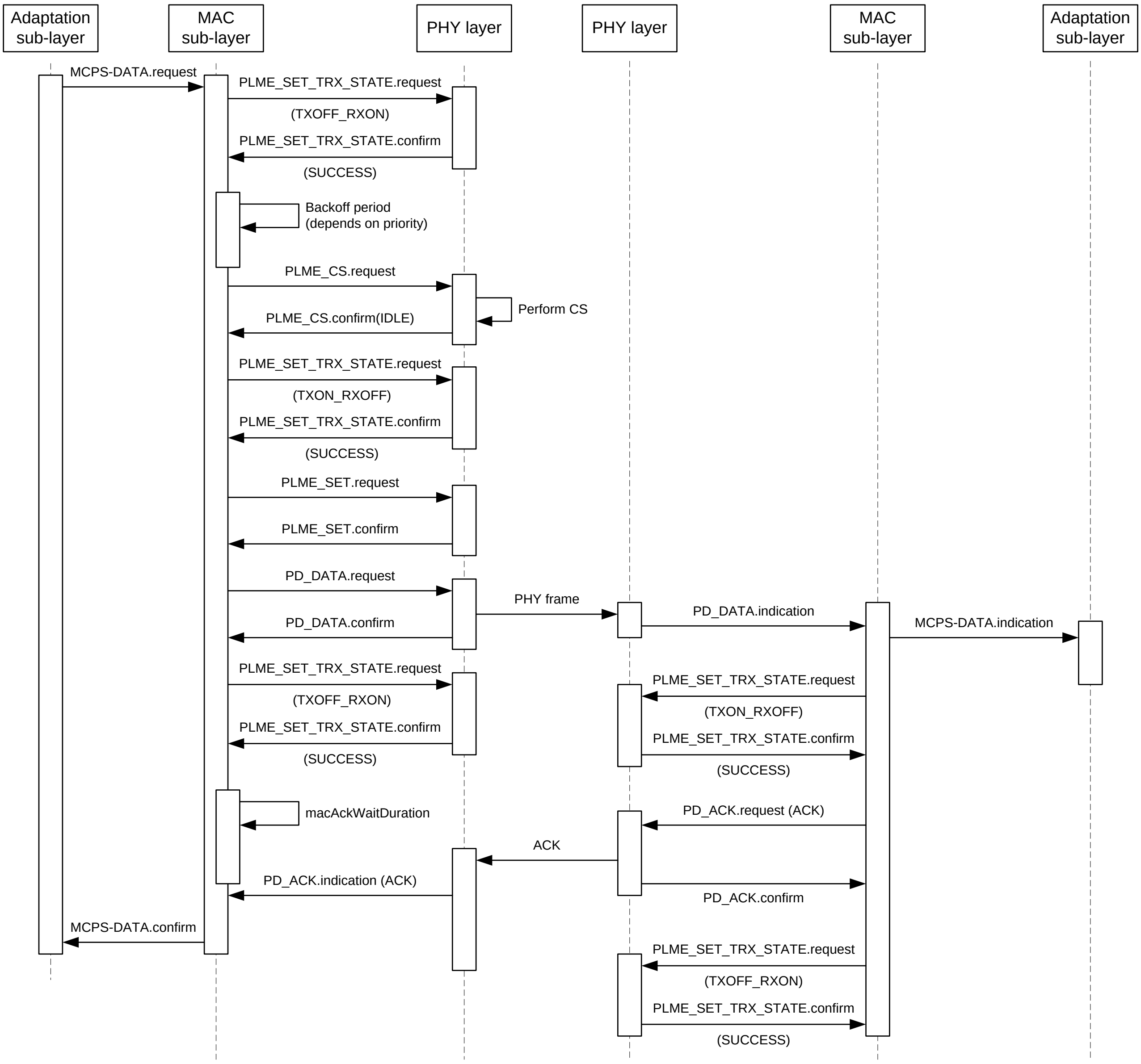


Figure A.3 – Data transmission message sequence chart

A.3.3.7.2.4 Channel estimation message sequence chart

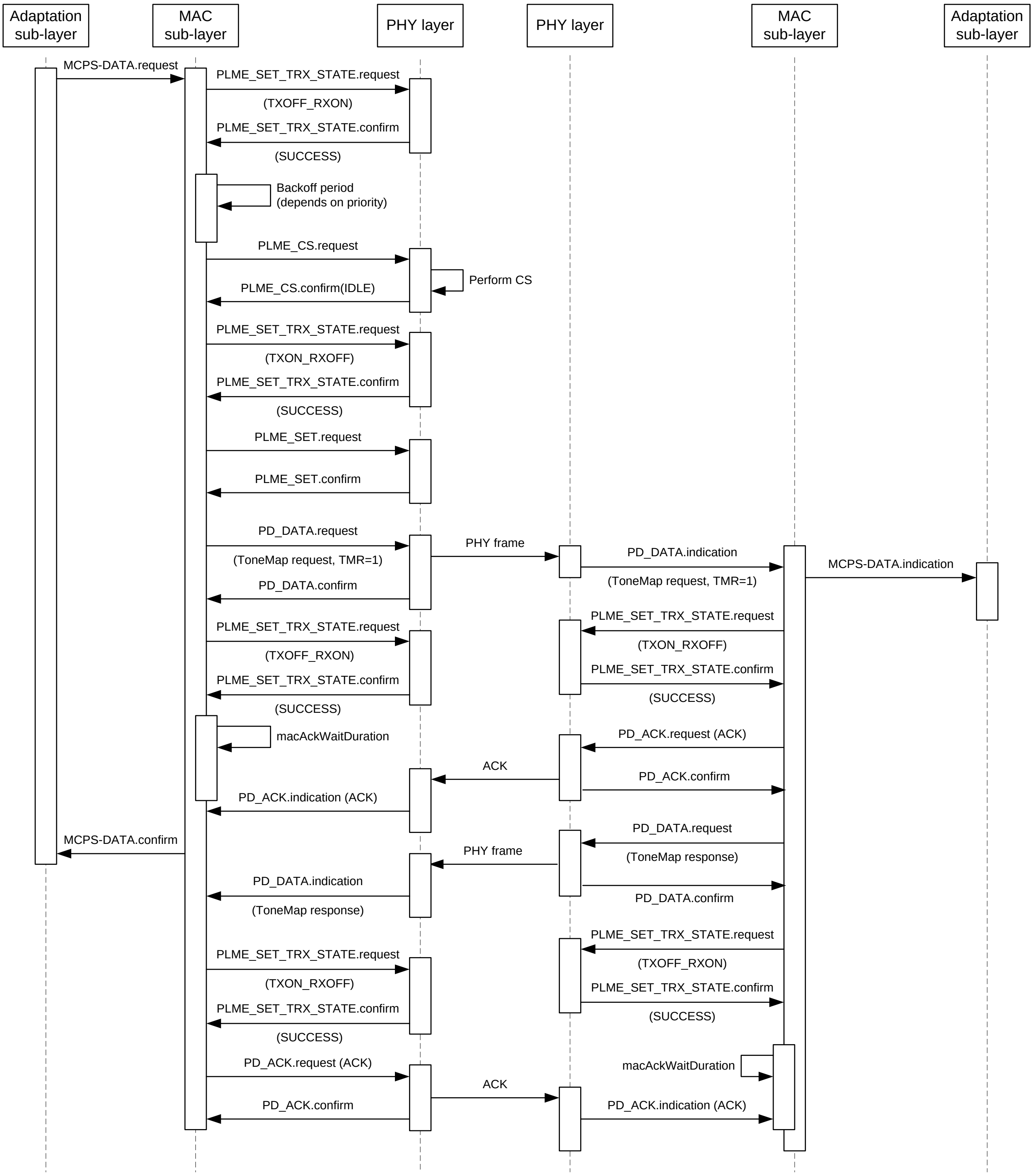


Figure A.4 – Channel estimation (tone map request) message sequence chart

A.3.3.8 MAC annexes (based on IEEE 802.15.4 annexes)

The MAC annexes of [IEEE 802.15.4] apply, with the selections specified in Table A.20.

Table A.20 – Selections from MAC annexes of [IEEE 802.15.4]

Clause	Title & remarks/modifications	Statement
Annex A	Service-specific convergence sublayer (SSCS) – IEEE 802.2 convergence sublayer is not used in the present specification	N/R
Annex B	CCM* mode of operation	N
Annex C	Test vectors for cryptographic building blocks	N
Annex D	Protocol implementation conformance statement (PICS) – The protocol implementation conformance tables are given in G3 Annex A.	E
Annex E	Coexistence with other IEEE standards and proposed standards – This annex relates to wireless PHY standards and is not relevant for PLC technology	N/R
Annex F	IEEE 802.15.4 regulatory requirements – This annex relates to wireless PHY standards and is not relevant for PLC technology	N/R

A.3.4 Adaptation sublayer specification

A.3.4.1 Services and primitives

The Services and Primitives of the adaptation sublayer are described in G3 Annex F.

A.3.4.2 Information base attributes

A.3.4.2.1 General

Table A.21 lists the Information Base (IB) attributes of the adaptation sublayer.

Table A.21 – Adaptation sublayer IB attributes

Attribute	Identifier	Type	Read Only	Range	Description	Default
GMK	0x00	16 bytes	No		Write-only 16 byte GMK	All Zero
adpIPv6Address	0x01	IPv6 address	Yes	Any	Defines the IPv6 address obtained from adpShortAddress	FE80::::FFFF:00FF:FE00:FFFF
adpBroadcastLog TableEntryTTL	0x02	Unsigned Integer	No	0-3 600	Defines the time while an entry in the adpBroadcastLogTable remains active in the table (in seconds).	90
adpMaxBroadcast Wait	0x03	Unsigned Integer	No	0-3600	Maximum wait time in seconds for broadcast packets	90
adpMaxDiscovery PerHour	0x04	Unsigned Integer	No	0-200	Maximum number of discovery requests per hour	60
adpNumDiscovery Attempts	0x05	Unsigned Integer	No	0-15	Number of discovery attempts	6

Table A.21 – Adaptation sublayer IB attributes

Attribute	Identifier	Type	Read Only	Range	Description	Default
adpDiscovery AttemptsSpeed	0x06	Unsigned Integer	No	1-3600	Allows programming the maximum wait time between invocation of two consecutive network discovery primitive (in seconds).	60
adpPANConflict Wait	0x08	Unsigned Integer	No	0-3 600	Defines the time to wait between two consecutive CONFLICT frames for the same conflicting PAN ID (in seconds).	1 800
adpMaxPANConflict Count	0x09	Unsigned Integer	No	0-100	Defines the maximum number of CONFLICT frames sent by a device for the same PAN ID.	3
adpActiveScan Duration	0x0A	Unsigned Integer	No	0-60	Defines the time while an active scan shall last (in seconds).	5
adpBroadcastLog Table	0x0B	Set	Yes	–	Contains the broadcast log table, see A.3.4.2.2 and A.3.4.4.2.2.1.	Empty
adpRoutingTable	0x0C	Set	Yes	–	Contains the routing table, see Tables A.17 and A.18.	Empty
adpGroupTable	0x0E	Set	No	–	Contains the group addresses to which the device belongs.	Empty
adpToneMask	0x0F	70 bits	No	Any	Defines the Tone Mask to use during symbol formation	All bits set to 1
adpMaxHops	0x10	Unsigned Integer	No	0-0x0E	Defines the maximum number of hops to be used by the routing algorithm.	8
adpDeviceType	0x11	Unsigned Integer	No	0-2	Defines the type of the device connected to the modem: 0: Device, 1: Server, 2: Not_Device, Not_Server	2
adpNetTraversal Time	0x12	Unsigned Integer	No	Any	The Max duration between RREQ and the correspondent RREP (in seconds)	20

Table A.21 – Adaptation sublayer IB attributes

Attribute	Identifier	Type	Read Only	Range	Description	Default
adpRrtTtl	0x13	Unsigned Integer	No	0-3 600	The time to live of a route request table entry (in seconds)	90
adpKr	0x14	Unsigned Integer	No	0-31	A weight factor for ROBO to calculate link cost(1).	0
adpKm	0x15	Unsigned Integer	No	0-31	A weight factor for modulation to calculate link cost(1).	0
adpKc	0x16	Unsigned Integer	No	0-31	A weight factor for number of active tones to calculate link cost(1).	0
adpKq	0x17	Unsigned Integer	No	0-31	A weight factor for LQI to calculate route cost(1).	10 for Cenelec A 40 for FCC
adpKh	0x18	Unsigned Integer	No	0-31	A weight factor for hop to calculate link cost(1).	4 for Cenelec A 2 for FCC
adpRREQRetries	0x19	Unsigned Integer	No	Any	The number of RREQ retransmission in case of RREP reception time out.	0
adpRREQRERR Wait	0x1A	Unsigned Integer	No	Any	The number of seconds to wait between two consecutive RREQ\RRER generations.	30
adpWeakLQIValue	0x1B	Unsigned Integer	No	Any	The weak Link Value defines the threshold below which a direct neighbour is not taken into account during the commissioning procedure (compared to the LQI measured).	3 for Cenelec A 5 for FCC
adpKrt	0x1C	Unsigned Integer	No	0-31	A weight factor for number of active routes in routing table to calculate link cost(1).	0
adpSoftVersion	0x1D	Set	Yes	–	The software version	–
adpSnifferMode	0x1E	Unsigned Integer	No	0-1	Sniffer Mode activation/ deactivation	0
adpMaxJoinWait Time	0x21	Unsigned Integer	No	0-1023	Network joint timeout in seconds for LBD.	20

Table A.21 – Adaptation sublayer IB attributes

Attribute	Identifier	Type	Read Only	Range	Description	Default
adpPathDiscovery Time	0x22	Unsigned Integer	No	Any	Timeout for path discovery in msec.	5000
adpUseNewGMK Time	0x23	Unsigned Integer	No	All	The wait time in sec for a device to use new GMK after rekeying as described in clause A.4.5.4	3600
adpExpPrecGMK Time	0x24	Unsigned Integer	No	All	The time in sec to keep PrecGMK after switching to a new GMK as described in clause A.4.5.4	3600
(1) Link cost calculation is provided in G3 Annex B.						

A.3.4.2.2 Routing table and broadcast table entry

Table A.22 describes the routing table entry:

Table A.22 – Routing table entry

Size → 16 bits	3 bits	18 bits
Next Hop address	Status	Life Time (in seconds)

Table A.22b – Broadcast log table entry

Field Name	Size	Description
SrcAddr	2 bytes	The 16-bit source address of a broadcast packet. This is the address of the broadcast initiator.
SeqNumber	Integer, 1 byte	The sequence number contained in the BC0 header
TimeToLive	13 bits	The remaining time to live of this entry in the Broadcast Log Table, in seconds.

A.3.4.3 Data frame format, datagram transmission and addressing (based on RFC 4944)

A.3.4.3.1 Selections from [IETF RFC 4944]

The data frame format, the theory of operation for datagram transmission using the IEEE 802.15.4 MAC sublayer, and the addressing scheme are specified in [IETF RFC 4944]using the selections listed in Table A.23.

Table A.23 – Selections from [IETF RFC 4944]

Clause	Title & remarks/modifications	Statement
1.	Introduction	N
1.1.	Requirements Notation	N

Table A.23 – Selections from [IETF RFC 4944]

Clause	Title & remarks/modifications	Statement
1.2.	Terms Used	N
2.	IEEE 802.15.4 Mode for IP – Data frames shall be acknowledged – Only non beacon-enabled network are used	S
3.	Addressing Modes – IPv6 prefixes learning via router advertisements is not supported	S
4.	Maximum Transmission Unit	N
5.	LoWPAN Adaptation Layer and Frame Format – Extension: additional Command Frame header: see A.3.4.3.2.1. – When more than one LoWPAN header is used in the same packet, they SHALL appear in the following order: Mesh Addressing Header Broadcast Header Fragmentation Header Command Frame Header (see A.3.4.3.2.1)	E
5.1.	Dispatch Type and Header	N
5.2.	Mesh Addressing Type and Header – The value of the HopsLeft field shall not exceed adpMaxHops (see A.3.4.2.1).	S
5.3.	Fragmentation Type and Header	N
6.	Stateless Address Autoconfiguration – The Interface Identifier (see [IETF RFC 4291]) for an IEEE 802.15.4 interface SHALL be based on the EUI-64 identifier assigned to the device, the latest being itself based on a EUI-48. – Additional care shall be taken when choosing a PAN identifier, so that not to interfere with I/G and U/L bits of the interface identifier. If the PAN identifiers are chosen randomly, then shall be logically ANDed with 0xFCFF	S
7.	IPv6 Link Local Address	N
8.	Unicast Address Mapping	N
9.	Multicast Address Mapping	N
10.	Header Compression	N
10.1.	Encoding of IPv6 Header Fields	N
10.2.	Encoding of UDP Header Fields	N
10.3.	Non-Compressed Fields	N
10.3.1.	Non-Compressed IPv6 Fields	N
10.3.2.	Non-Compressed and Partially Compressed UDP Fields	N
11.	Frame Delivery in a Link-Layer Mesh	S
11.1.	LoWPAN Broadcast	N
12.	IANA Considerations	N
13.	Security Considerations	N
14.	Acknowledgements	N/R

Table A.23 – Selections from [IETF RFC 4944]

Clause	Title & remarks/modifications	Statement
15.	References	N/R
15.1.	Normative References	N
15.2.	Informative References	I
Appendix A.	Alternatives for Delivery of Frames in a Mesh	N/R

A.3.4.3.2 Extensions to RFC 4944

A.3.4.3.2.1 Command frame header

In addition of the LoWPAN header specified in [IETF RFC 4944], the present standard defines a new one: Command Frame header. This is used for the mesh routing procedure defines in A.3.4.4.

As shown in Figure A.5, the ADP sublayer command frames are identified using the ESC header type (see clause 5.1 of [IETF RFC 4944]), followed by an 8-bit dispatch field indicating the type of ADP command. This header shall be in the last position if more than one header is present in the 6LowPAN frame.

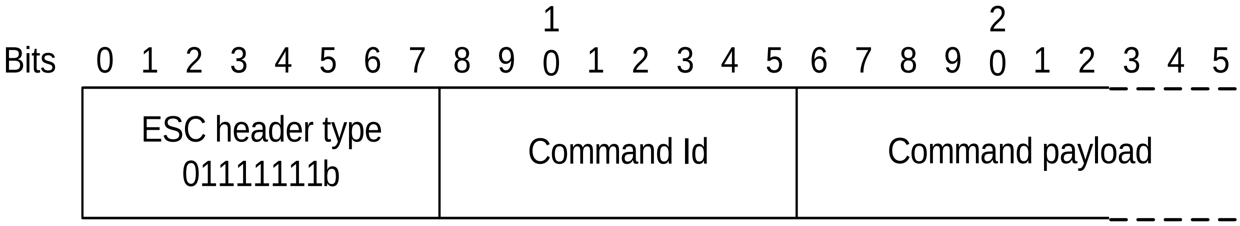


Figure A.5 – Command frame header format

The ADP sublayer command frames are specified in Table A.24:

Table A.24 – Command frame header identifier

Command	Command Id	Comments	Specified in …
Mesh routing message	0x01	Use for mesh routing protocol	Clause A.3.4.4
LoWPAN Bootstrapping Protocol message	0x02	Use for LoWPAN Bootstrap procedure	Clause A.3.4.5
Contention Free Access Command	0x03	Optional	Clause A.3.4.3.2.2

A.3.4.3.2.2 Contention free access command

Contention Free Access procedure is an optional feature of this standard and described in A.C.4.

The adaptation layer generates CFA (Contention Free Access) command if it receives an ADPD-DATA.request primitive with QualityOfService = 2 (See A.F.1.2).

Figure A.6 defines the format of the CFA command (see also Table A.25).

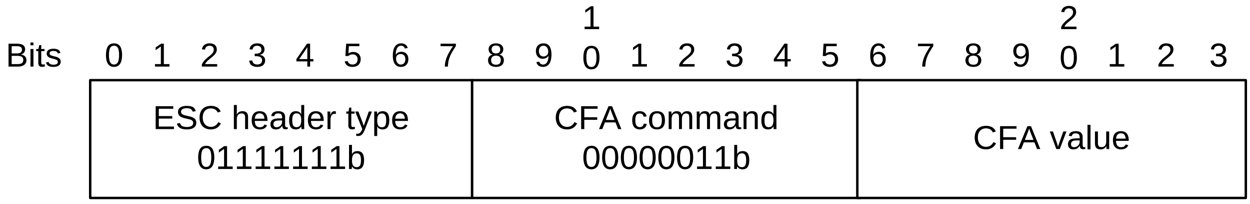


Figure A.6 – CFA command format

Table A.25 – CFA value field description

CFA value	Description
0	Request to allow a transmission during contention free slot
1	Request to stop a transmission during contention free slot
2	Response with SUCCESS
3	Response with FAIL

The network coordinator may always use a contention free slot for transmission if other devices are not allowed to use it the same time. Other devices shall ask the network coordinator for permission to use a contention free slot (CFS) for transmission by sending CFA command with request. The network coordinator may allow requested device to use a CFS for transmission by sending a confirmation response. After receiving a successful response from the network coordinator requested device can start a transmission during CFS. If the network coordinator denies a request the device shall not use a CFS for transmission. The requested device shall send a request to stop using CFS when it is done with contention free transmission.

Priority management can be performed using the "Normal" and "High" priority values for QOS parameter of MCPS-DATA.request primitive.

A.3.4.4 Mesh routing (based on G3 Annex H)

A.3.4.4.1 Selections from G3 Annex H

The mesh routing as described in G3 Annex H applies, with the selections specified in Table A.26.

Table A.26 – Selections from G3 Annex H

Clause	Title & remarks/modifications	Statement
1.	Introduction	N
2.	Requirements notation	N
3.	Overview – Routing is only permitted with 16-bit addresses, – LOAD uses the route cost described in G3 Annex B as a metric of routing.	S,E
4.	Terminology	N
5.	Data Structures	N

Table A.26 – Selections from G3 Annex H

Clause	Title & remarks/modifications	Statement
5.1	Routing Table Entry – The destination address shall be a 16-bit address, – The next hop address shall be a 16-bit address, – The routing table is stored in the IB under the attribute adpRoutingTable.	S, E
5.2	Route Request Table Entry – The originator address shall be a 16-bit address, – The reverse route address shall be a 16-bit address.	S
5.3	Message Format – For path discovery procedure, two messages have been added: Path Request (PREQ) and Path Reply (PREP). See A.3.4.4.2.4.	E
5.3.1	Route Request (RREQ) – The CT field shall be equal to 0x0F, to specify the use of the route cost described in G3 Annex B, – The D bit shall be set to 1, – The O bit shall be set to 1, – The link layer destination and originator address shall be 16-bit addresses.	S
5.3.2	Route Reply (RREP) – The CT field shall be equal to 0x0F, to specify the use of the route cost described in G3 Annex B, – The D bit shall be set to 1, – The O bit shall be set to 1, – The link layer destination and originator address shall be 16-bit addresses.	S
5.3.3	Route Error (RERR) – The D bit shall be set to 1, – The O bit shall be set to 1, – The unreachable address shall be 16-bit addresses.	S
6.	Operation	N
6.1	Generating Route Request	N
6.2	Processing and Forwarding Route Request	N
6.3	Generating Route Reply	N
6.4	Receiving and Forwarding Route Reply	N
6.5	Local Repair and RERR – If a link break occurs or a device fails during the delivery of data packets, the upstream node of the link break SHALL repair the route locally, and execute the repairing procedure described in the present clause.	S

Table A.26 – Selections from G3 Annex H

Clause	Title & remarks/modifications	Statement

7.	Configuration Parameters
	– The values of the configuration parameters shall be:
	NET_TRAVERSAL_TIME = adpNetTraversalTime
	RREQ_RETRIES = adpRREQRetries
	WEAK_LQI_VALUE = adpWeakLQIValue
	– Extension: the following parameters are added by the present standard:
	RREQ_RERR_WAIT = adpRREQRERRWait
	PATH_DISCOVERY_TIME = adpPathDiscoveryTime
		S, E

8.	IANA Consideration	N
9.	Security Considerations	N/R
10.	Acknowledgments	N/R
11.	References	N
11.1	Normative Reference	N
11.2	Informative Reference	I

A.3.4.4.2 Extensions to G3 Annex H

A.3.4.4.2.1 Unicast packet routing

The routing of unicast packet is performed using the following algorithm on reception of a MCPS-DATA.indication from the MAC layer:

IF (MAC destination address == address of device)

- IF (6LoWPAN destination address == 6LoWPAN address of device)
  - Generate an ADPD-DATA.indication primitive to indicate the arrival of a frame to the upper layer, with the following characteristics (see A.F.1.4):
  - DstAddrMode = 0x02
  - DstAddr = 6LoWPAN destination address
  - SrcAddr = The originator address in the 6LoWPAN mesh header
  - NsduLength = length of the payload
  - Nsdu = the payload
  - LinkQualityIndicator = msduLinkQuality (see A.F.3.2)
  - SecurityEnabled = (SecurityLevel != 0)
- ELSE IF (6LoWPAN destination address is in the neighbour table)
  - Forward the packet to the destination address, by invoking an MCPS-DATA.request primitive with the destination address set to the final destination address
- ELSE IF (6LoWPAN destination address is in the routing table and next hop in the neighbour table)
  - Forward the packet to the next hop found in the routing table, by invoking an MCPS-DATA.request primitive, and using the communication parameters to that device contained in the neighbour table.
- ELSE IF (6LoWPAN Destination Address not in routing table)
  - Perform a link repair as described in clause 6.5 of G3 Annex H

  - Queue the packet for a sending retry
- ELSE
  - Drop the frame
- ELSE IF (MAC Destination address == 0xFFFF)
  - This is a broadcast frame: execute algorithm described in A.3.4.4.2.2 of this document
- ELSE
  - Drop the frame

A.3.4.4.2.2 Multicast/Broadcast

A.3.4.4.2.2.1 Packet routing

The packet routing mechanism is based on clause 11.1 of [IETF RFC 4944]. This clause details more precisely the routing of broadcast and multicast packets.

As described in clause 11.1 of [IETF RFC 4944], each broadcast packet has a BC0 header containing a sequence number. Each time a node sends a broadcast packet, it shall increment this sequence number.

Each node shall have a broadcast log table. This table is used for routing broadcast packets, and each entry contains the parameters described in Table A.22b:

Each time a device receives a broadcast address with a HopsLft field of mesh header (see clause 5.2 of [IETF RFC 4944]) strictly greater than 0, it shall check if an entry already exists in the broadcast log table having the same SrcAddr and SeqNumber. If an entry exists, the received frame is silently discarded. Else, a new entry is added in the table, and the TimeToLive field is initialized with the value adpBroadcastLogTableEntryTTL (see A.3.4.2). When this value reaches 0, the entry is removed from the broadcast log table.

When a device receives a broadcast frame, so that it has to create an entry in the broadcast log table, it shall decrement its HopsLft field. If HopsLft is not zero, it triggers the transmission of the received broadcast frame.

This can be summarized by the following algorithm, executed upon reception of a frame whose destination address is 0xFFFF:

IF (final destination address = broadcast address) or (final destination address is found in adpGroupTable)

- IF ((SrcAddr, SeqNumber) exists in broadcast log table)
  - Discard frame
- ELSE
  - Create one entry (SrcAddr, SeqNumber, adpBroadcastLogTableEntryTTL) in broadcast log table, with the corresponding frame characteristics.
  - Generate an ADPD-DATA.indication primitive to upper layer with the following characteristics:
    - DstAddrMode = 0x02
    - DstAddr = Destination address in the 6LoWPAN mesh header (multicast or broadcast address)
    - SrcAddr = The originator address in the 6LoWPAN mesh header
    - NsduLength = length of the data
    - Nsdu = the data
    - LinkQualityIndicator = msduLinkQuality (see A.F.3.2)

  - SecurityEnabled = (SecurityLevel != 0)
- HopsLft=HopsLft -1
- If (HopsLft > 0), Trigger the frame transmission.

NOTE – In case of a multicast address, the broadcast address 0xFFFF is used at the MAC level as mentioned in clause 3 of [IETF RFC 4944]. Multicast frames are routed using the same algorithm as broadcast frames.

The broadcast log table is available in the Information Base with the attribute adpBroadcastLogTable (see A.3.4.2).

A.3.4.4.2.2.2 Groups

Each device can belong to one or more group of devices. The IB attribute adpGroupTable (see A.3.4.2) stores a list of 16-bit group addresses.

When the device receives a MAC broadcast message, and if the final destination address in the 6LoWPAN mesh header is equal to one of the 16-bit group addresses in adpGroupTable, then an ADPD-DATA.indication primitive is generated to the upper layer (as described in A.3.4.4.2.2.1).

Groups can be added or removed from the adpGroupTable using ADPM-SET.request primitive. The size of this table is implementation specific, and shall have at least one entry. The way groups are managed by upper layers is beyond the scope of this document.

A.3.4.4.2.3 Route discovery

A.3.4.4.2.3.1 Manual route discovery

A manual route discovery can be triggered by the upper layer, for maintenance or performance purposes. This is done through the invocation of the ADPM-ROUTE-DISCOVERY.request primitive. The adaptation sublayer then generates a RREQ frame and executes the algorithms as described in A.3.4.4.1.

After the algorithm completes, the adaptation sublayer generates an ADPM-ROUTE-DISCOVERY.confirm primitive with the corresponding status code, and eventually modify its routing table.

Only one route discovery procedure can be processed in the same time. All other ADPM-ROUTE-DISCOVERY.request will be ignored.

All devices shall handle RREQ, RREP and RERR frames as described in A.3.4.4.1 and modify their routing tables accordingly.

A.3.4.4.2.3.2 Automatic route discovery

If an ADPD.DATA.request primitive is invoked with its DiscoverRoute parameter set to TRUE, and if no entry is available in the routing table for the device designated by DstAddr, then the adaptation layer generates a RREQ and executes the algorithms described in A.3.4.4.1 in order to find a route to the destination. If the route discovery succeeds, then the data frame is send to the destination according to the newly discovered route. If the route discovery fails, then the adaptation layer shall generate an ADPD-DATA.confirm primitive with the status code ROUTE_ERROR.

If an ADPD.DATA.request primitive is invoked with its DiscoverRoute parameter set to FALSE, and if no entry is available in the routing table for the device designated by DstAddr, then the adaptation layer shall generate an ADPD-DATA.confirm primitive with the status code ROUTE_ERROR.

Route repairing procedures are described in A.3.4.4.1.

A.3.4.4.2.3.3 RREQ RERR generation frequency limit

A node shall wait RREQ_RERR_WAIT second between two successive RREQ/RERR generations to limit the number of broadcast packet in the network. The definition of the RREQ_RERR_WAIT parameter is given in A.3.4.4.1.

A.3.4.4.2.4 Path discovery

A.3.4.4.2.4.1 Operation

A path discovery can be triggered by the upper layers, for maintenance or performance purposes. This is done through the invocation of the ADPM-PATH-DISCOVERY.request primitive. The adaptation sublayer then generates a PREQ frame and executes the algorithms described in following subclauses.

After the algorithm completes (with the reception of a PREP frame), the adaptation sublayer generates an ADPM-PATH-DISCOVERY.confirm primitive to the upper layer.

Only one path discovery procedure can be processed in the same time. All other ADPM-PATH-DISCOVERY.request from the upper layer will be ignored.

Generating a path request (PREQ)

During the path discovery period, an originator, a node that requests a path discovery, generates a Path Request (PREQ) message (see A.3.4.4.2.4.2).

Once transmitted, the node waits for a Path Reply (PREP), else, and after PATH_DISCOVERY_TIME milliseconds, the node generate an ADPM-PATH-DISCOVERY.confirm to the upper layer with an NsduId field containing a Path Reply (PREP) with HOPS fields set to 0.

Processing and forwarding a path request (PREQ)

Upon receiving a Path Request (PREQ), an intermediate node tries to find entry of the same destination address in the routing table. If the entry is found, the node just forwards the PREQ to the next hop toward the destination. Else, the node just discards the PREQ.

Generating a path reply (PREP)

A final node, on receiving a Path Request (PREQ), generates a Path Reply (PREP) with the following information:

- Flag R set to 0,
- Hops field set to 1,
- Hops1 address set to its own address.

If an intermediate node cannot find a route to the destination of the PREQ, it generates a Path Reply (PREP) with R flag set to 1, the HOP to 1 and the Hops1 address to its own address then sends it to the source of the PREQ.

1 Processing and forwarding a path reply (PREP)

Upon receiving a Path Reply (PREP), an intermediate node tries to find entry of the same destination address in its own routing table. If the entry is found, the node just forwards the PREP to the next hop toward the destination with the following field updates:

- Set the Hop$_N$ address field in the PREP to its own address, with N = HOPS+1,
- Update the RC field with its own route cost and,
- Increment the HOPS field by one.

If there's no route to the destination, the node just discard the Path Reply message received.

A.3.4.4.2.4.2 Path request frame

The path request frame format and the detail of its related fields are described in Figure A.7 and Table A.27 respectively.

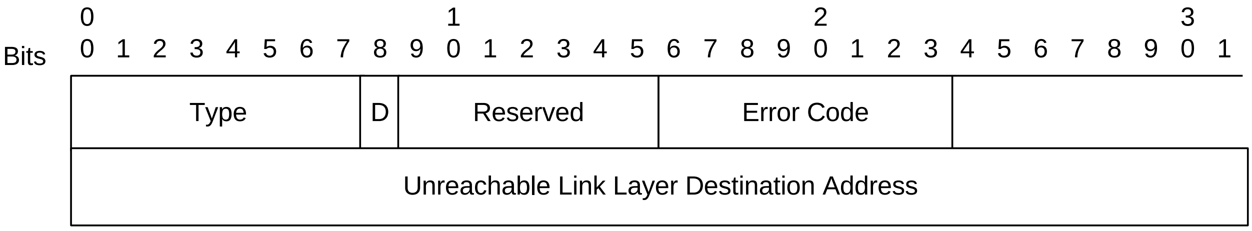


Figure A.7 – Path request (PREQ) message format

Table A.27 – Path request (PREQ) fields' definition

Field	Size, bits	Value	Definition
Type	8	4	Path Request (PREQ) message identifier
Destination Address	16	–	The 16 bit short link layer address of the destination for which a route is supplied
Originator Address	16	–	The 16 bit short link layer address of node which originated the packet.

A.3.4.4.2.4.3 Path reply frame

The path reply frame format and the detail of its related fields are described in Figure A.8 and Table A.28 respectively.

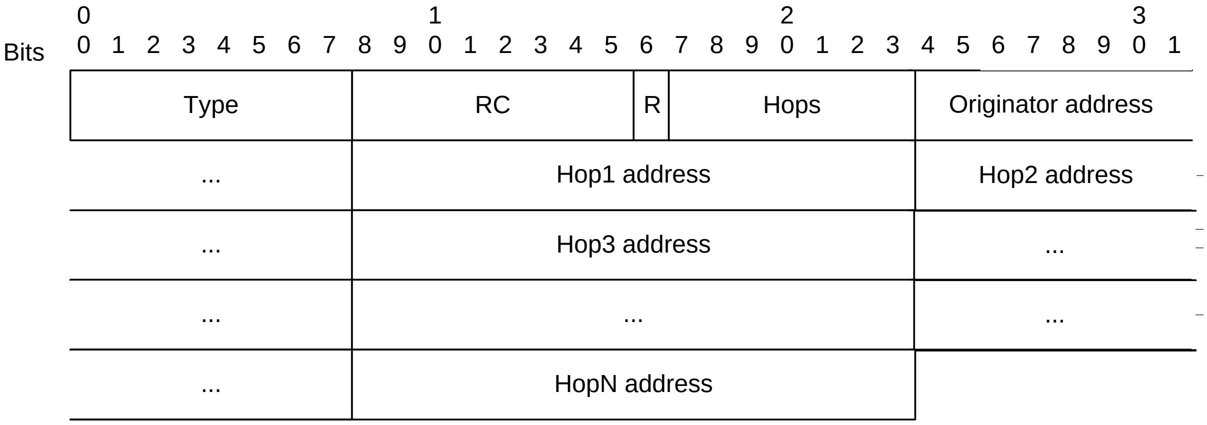


Figure A.8 – Path reply (PREP) message format

Table A.28 – Path reply (PREP) fields' definition

Field	Size (bits)	Value	Definition
Type	8	5	Path Reply (PREP) message identifier
RC	8		Route Cost – The accumulated link cost of the reverse route from the originator to the sender of the message.
R	1	0/1	Path discovery result: 1 Success of path discovery 0 Failure of path discovery
Hops	7	–	Number of hops of the route
Originator Address	16	–	The 16 bit short link layer address of node which originated the packet.
Hop_N	16	–	The 16 bit short link layer address of nodes constituting the path.

A.3.4.5 Commissioning of new devices (based on G3 Annex I)

A.3.4.5.1 Selections from G3 Annex I

The commissioning of new devices on an existing network as described in G3 Annex I applies, with the selections specified in Table A.29.

Table A.29 – Selections from G3 Annex I

Clause	Title & remarks/modifications	Statement
1.	Introduction	N
2.	Terminology	N
2.1.	Requirements notation	N
3.	Bootstrapping – Obtaining a 16-bit short address and security credentials are mandatory parts of the commissioning process.	S
3.1.	Resetting the device	N
3.2.	Scanning through channels – For getting the information of other devices within POS, the device SHALL perform an active scan.	S
3.3.	LoWPAN Bootstrapping Mechanism – 'LBA discovery phase' is described in A.3.4.5.2.2	E
3.3.1.	LoWPAN Bootstrapping Protocol message format	N
3.3.1.1	LBP message – Some enhancements and clarifications to LBP message format are given in A.3.4.5.2.1.	E
3.3.2.	LoWPAN Bootstrapping Information Base – PAN_type shall be Secured – Address_of_LBS shall be equal to the default address of the PAN coordinator that is 0x0000. – Short_Addr_Distribution_Mechanism shall be 0 for centralized address management.	S

Table A.29 – Selections from G3 Annex I

Clause	Title & remarks/modifications	Statement
3.3.3.	LBA discovering phase – Some enhancements and clarifications to 6LoWPAN bootstrapping procedure are given in A.3.4.5.2.2. – The LBD shall perform an active scan instead of broadcasting a LBA solicitation message.	E
3.3.4.	LoWPAN Bootstrapping Protocol (LBP)	S
3.3.5.	Bootstrapping in open 6LoWPAN	N/R
3.3.6.	1. The LBP messages from the LBD to the LBA are sent by invocation of the MCPS-DATA.request primitive with the following attributes: – SrcAddrMode = 0x03; – SrcAddr = Own EUI-64 address; – DstAddrMode = 0x02; – DstAddr = 16-bit short address of the LBA passed as an argument to the ADPM-NETWORK-JOIN.request primitive; – DstPANId = The PAN ID passed as an argument to the ADPM-NETWORK-JOIN.request primitive – msduLength = length of the LBP message – msdu = the LBP message itself – msduHandle = random number – QualityOfService = 0 – SecurityLevel = FALSE. All other parameters can be ignored. 2. The LBP messages from the LBA to the LBS are relayed by using the routing algorithm as described in clause A.3.4.4 with the DiscoverRoute parameter set to TRUE and the SecurityLevel set to a non-zero value. 3. The LBP messages in LBS are sent by invocation of the ADPM-LBP.request primitive which carries the following attributes: – DstAddrType = 0x02 – DstAddr = 16 bit LBA address – NsduLength = the length of the LBP message – Nsdu = the LBP message itself – NsduHandle = random number – NsduType = 0 – MaxHops = maximum number of hops – DiscoverRoute = TRUE – QualityOfService = 0 – SecurityEnabled = TRUE 4. The LBP messages from the LBS to the LBD are relayed to the LBA by using the routing algorithm as described in clause A.3.4.4 with the DiscoverRoute parameter set to TRUE and the SecurityLevel set to a non-zero value. 5. The LBP messages from the LBA to the LBD are sent by invocation of the MCPS-DATA.request primitive with the following attributes: – SrcAddrMode = 0x02; – SrcAddr = Own 16-bit short address;	S

Table A.29 – Selections from G3 Annex I

Clause	Title & remarks/modifications	Statement

	– DstAddrMode = 0x03;
	– DstAddr = The EUI-64 contained as a LBD in the LBP message;
	– DstPANId = LBA PAN_ID;
	– msduLength = length of the LBP message;
	– msdu = the LBP message itself;
	– msduHandle = random number;
	– QualityOfService = 0;
	– SecurityLevel = FALSE;	

3.3.7.	Role of Entities in LBP – If a LBD does not find any LBA during the LBA discovery phase, it shall still perform LBA discoveries as long as it is not commissioned. Note that LBA discovery is done using active scans rather that broadcasting LBA solicitation messages. – Only secured networks are used	S
3.4.	Assigning the short address Short addresses are assigned in a centralized fashion by the LBS	S
3.5.	Obtaining IPv6 address – The devices do not need to obtain an IPv6 address prefix, and the procedures described in this clause as well as in [IETF RFC 4862] shall be ignored. Only the IPv6 Link Local Address generated as stated in clause 7 of [IETF RFC 4944] is used for communication.	M

3.6.	Configuration Parameters
	– The values of the configuration parameters shall be:
	CHANNEL_LIST = 0xFFFF800 (not used)
	SCAN_DURATION = adpActiveScanDuration (see A.3.4.2.1)
	SUPERFRAME_ORDER = 15
	BEACON_ORDER = 15
	START_RETRY_TIME = 0 (not used)
	JOIN_RETRY_TIME = 0 (not used)
	ASSOCIATION_RETRY_TIME = 0 (not used)
		M

4.	IANA Consideration	N/R
5.	Security Considerations	N
6.	Contributors	N/R
7.	Acknowledgments	N/R
8.	References	N
8.1.	Normative References	N
8.2.	Informative References	I

A.3.4.5.2 Extensions to G3 Annex I

A.3.4.5.2.1 LoWPAN bootstrapping protocol (LBP) message format

A.3.4.5.2.1.1 General

LBP message format and the detail of its related fields are described in Figure A.9 and parameters (see A.2.4.5.2.1.3).

Table A.30 respectively.

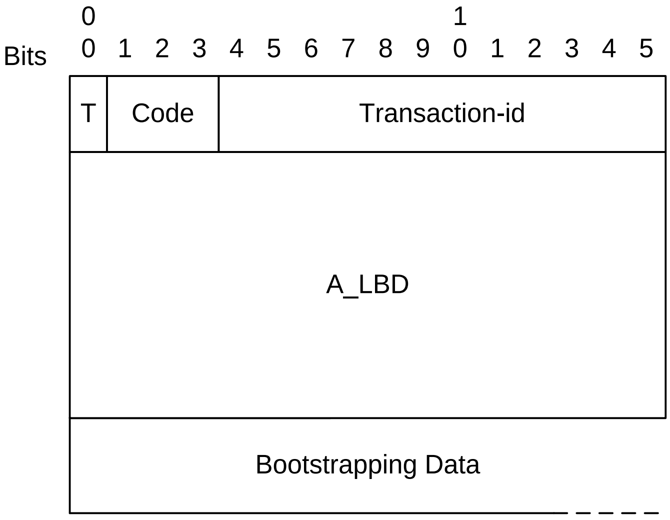


Figure A.9 – LBP message format

Where

T identifies the type of message (1-bit)

0 Message from LBD

1 Message to LBD

Code identifies the message code (3-bit) defined in Table A.30.

Transaction-id aids in matching Responses with Request (12-bit)

A_LBD The A_LBD field is 8 octets and indicates the EUI-64 address of the bootstrapping device (LBD).

Bootstrapping Data The Bootstrapping Data field is of variable length and contains additional information elements. Two types are defined:

Embedded EAP messages (see A.2.4.5.2.1.2),

Configuration parameters (see A.2.4.5.2.1.3).

Table A.30 – T and code fields in LBP message

T	Code	LBD message	Description
0	001	JOINING	The LBD requests joining a PAN and provides necessary authentication material
1	001	ACCEPTED	Authentication succeeded with delivery of Device Specific Information (DSI) to the LBD
1	010	CHALLANGE	Authentication in progress. PAN Specific Information (PSI) may be delivered to the LBD

Table A.30 – T and code fields in LBP message

T	Code	LBD message	Description
1	011	DECLINE	Authentication failed
0/1	100	KICK	KICK frame is used by a PAN coordinator to force a device to lose its MAC address, or by any device to inform the coordinator that it left the PAN. On reception of this frame, a device shall set its short address to the default value of 0xFFFF, disconnect itself from the network, and perform a reset of the MAC and adaptation layers. See A.3.4.5.2.2.7 for details about kicking procedure.
0	101	CONFLICT	CONFLICT frame is used by a device to inform the PAN coordinator that it has detected another PAN operating in the same POS. See A.3.5.2 for details about PAN ID conflict handling.

A.3.4.5.2.1.2 Embedded EAP messages

LBP messages embed Extended Authentication messages (EAP) as defined by [IETF RFC 3748]. Figure A.10 describes minor modification to fit the generic LBP information element format.

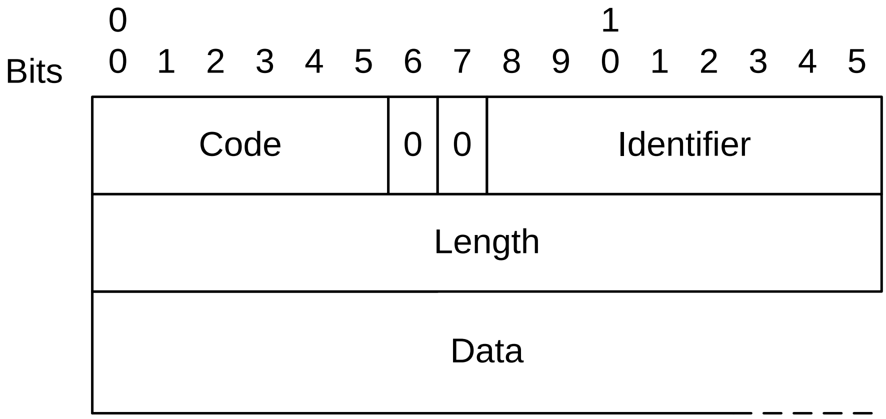


Figure A.10 – Embedded EAP message format (generic)

Where

Code identifies the Type of EAP packet (6-bit). EAP Codes are assigned as follows:

Request (sent to the peer = LBD)

Response (sent by the peer)

Success (sent to the peer)

Failure (sent to the peer)

The Code field is slightly different from a regular EAP Code field as specified in [IETF RFC 3748]. The conversion appears straightforward in both directions. The proper conversion shall apply when the EAP message is propagated over another protocol (i.e., RADIUS) and in case of integrity protection covering the EAP header.

Identifier aids in matching Responses with Requests (8-bit).

Length The Length field is two octets and indicates the length, in octets, of the EAP packet including the Code, Identifier, Length, and Data fields. A message with the Length field set to a value larger than the number of received octets shall be silently discarded.

Data The Data field is zero or more octets. The format of the Data field is determined by the Code field. Refer to [IETF RFC 3748] for more details on:

Specific format for Request/Response messages and the introduction of the Type field (Identity, Nak, etc.),

Specific format for Success/Failure messages with an empty Data field.

A.3.4.5.2.1.3 Configuration parameters

Configuration parameter format and the detail of its related fields are described in Figure A.11:

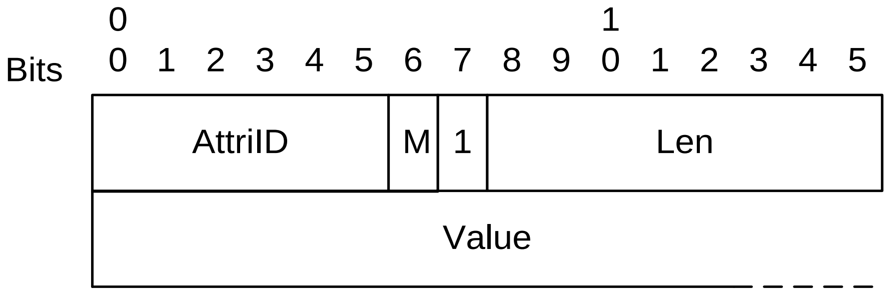


Figure A.11 – Configuration parameter format

Where

Attr-ID represents the ID of the Attribute in LoWPAN Information Base (LIB) (6-bit)

M identifies the type of the Attribute (1-bit):

- Device Specific Information (DSI)
- PAN Specific Information (PSI)

Len indicates the length, in octets, of the Value field (8-bit)

Value is zero or more octets and contains the value of the Attribute. Its format is defined by Attr-ID.

A.3.4.5.2.2 6LoWPAN bootstrapping procedures

A.3.4.5.2.2.1 Overview

This clause proposes some enhancements and clarifications to 6LoWPAN bootstrapping procedure. This procedure is executed when the ADPM-NETWORK-JOIN.request primitive is invoked by the upper layer.

Figure A.12 provides an overview of the messages exchanged between devices during the Bootstrapping procedure.

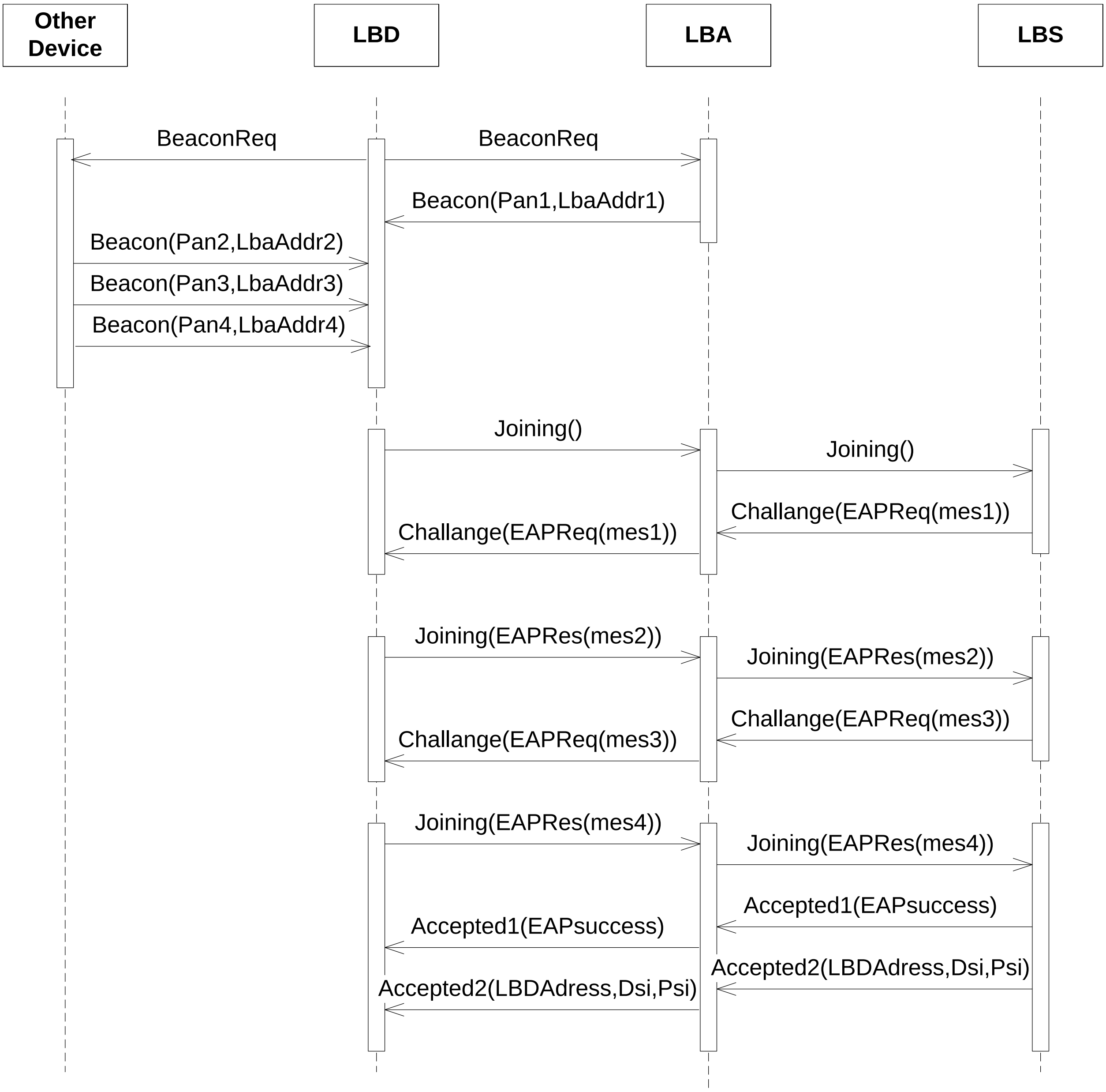


Figure A.12 – Bootstrapping protocol messages sequence chart

Figure A.13 summarizes the forwarded messages involved during a nominal association procedure on a PAN between different protocol layers of the devices, when a single LBP protocol message needs to be exchanged between the LBD and the LBS.

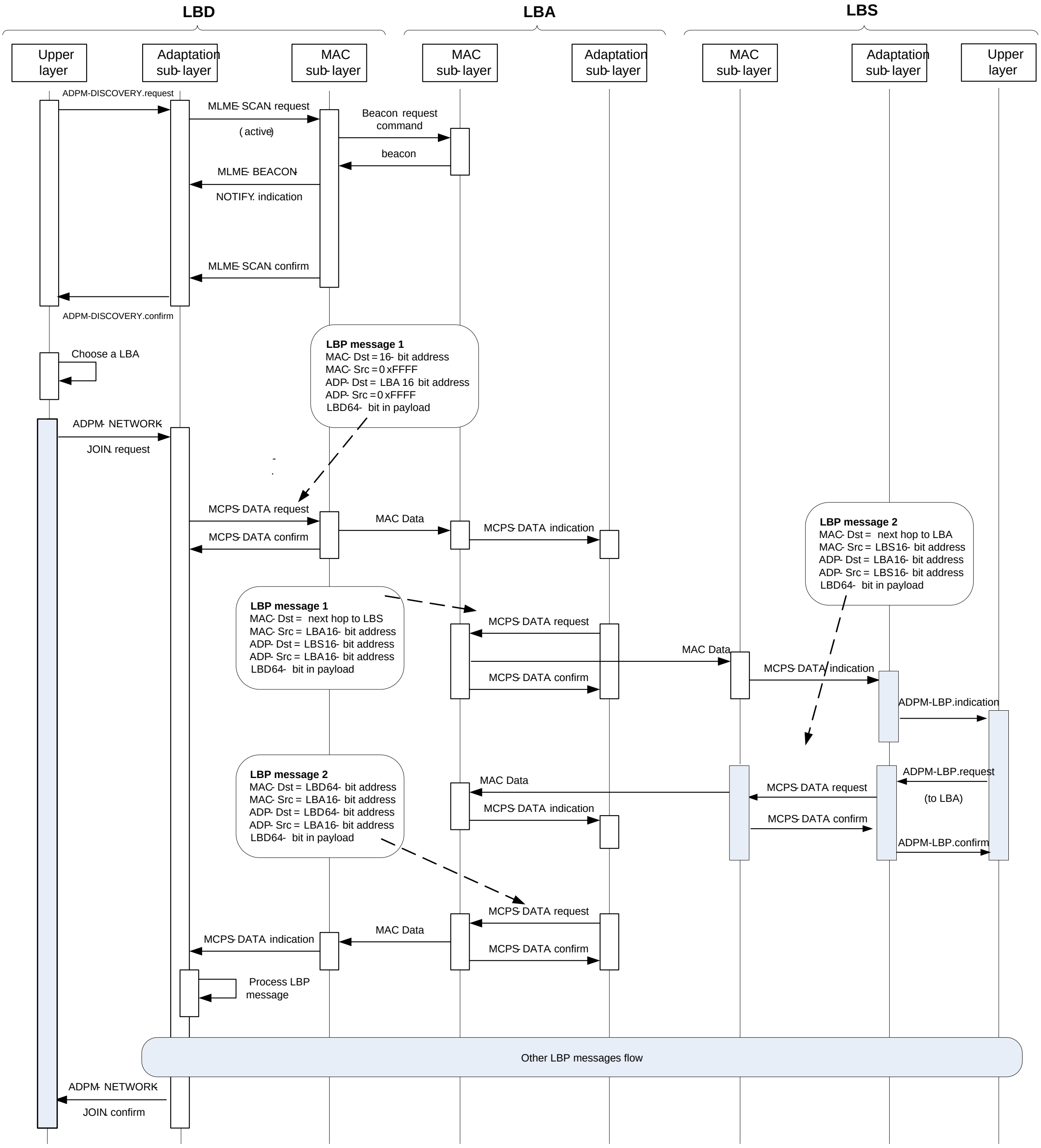


Figure A.13 – Bootstrapping protocol messages forwarding

A.3.4.5.2.2.2 Discovering phase

At the beginning of the Bootstrapping procedure, an End Device (aka LoWPAN Bootstrapping Device or LBD) shall launch an "active channel scan" (cf. [IEEE 802.15.4] clause 7.5.2.1.2).

The higher layer can start an active scan by invoking the ADPM-DISCOVERY.request primitive, and specifying the duration of the scan. The adaptation layer then invokes the MLME-SCAN.request primitive of the MAC layer with the following parameters:

- ScanType = 0x01;
- ScanChannels = all bits to 0 (not used);
- ScanDuration = Duration;
- ChannelPage = 0 (not used);

- SecurityLevel = 0;
- KeyIdMode = Ignored;
- KeySource = Ignored;
- KeyIndex = Ignored.

The LBD sends a 1-hop broadcast Beacon.request frame and any Full Feature Device in the Neighbourhood shall reply by sending a Beacon frame with its PAN identifier, short address and capabilities.

Upon receiving each Beacon frame, the MAC layer in LBD issues a MLME-BEACON-NOTIFY.indication primitive with the PANDescriptor parameters corresponding to the beacon. At the end of scan duration, the adaptation layer generates an ADPM-DISCOVERY.confirm primitive which contains the PANDescriptorList. If multiple beacons with the same PAN Identifier are received, only one beacon per discovered PAN is selected to be included in PANDescriptor.

The choice of the beacon is based on the following criteria:

- Association permit, rejected if negative;
- Minimum value of route cost to coordinator;
- Maximum value of beacon link quality;
- Short address, according to a round robin algorithm.
- After finishing the scan procedure, the device can join the network following the procedure described in A.3.4.5.2.2.6.

A device shall not perform more than adpMaxDiscoveryPerHour network discovery procedures per hour.

A.3.4.5.2.2.3 Access control phase

Once the discovery phase is finished, the LBD send an LBP JOINING frame to the LBA. This frame includes a field that carries the EUI-64 address of the joining LBD.

This frame, as any other frame during the initial part of the Bootstrapping process, is transmitted between the LBD and the LBA without any additional security at the MAC layer.

When received by the LBA, this frame is relayed by the LBA to the LBS. The LBA is supposed fully bootstrapped with the full capability to directly transmit any message to the LBS in a secure way.

The LBP protocol has been designed to fit two different authentication architectures:

– The authentication function is directly supported by the LBS, and in this case all the authentication material (access lists, credentials, etc.) shall be loaded in the LBS or,

– The authentication function is supported by a remote (and usually centralized) AAA server, and in this case, LBS is only in charge of forwarding the EAP messages to the AAA server over a standard AAA protocol (i.e., RADIUS, RFC 2865).

The following procedure description is only based on the first architecture but extension to the second one appears straightforward.

So, when received by the LBS, the EUI-64 address may be compared with an Access Control list (white list or black list) with the following possibilities:

– This address does not fit the Access Control list and the LBS send back an LBP DECLINE message, embedding an EAP Failure message or,

– This address fit the Access Control List (or the Access Control is not implemented) and the LBS send back an LBP CHALLENGE message, embedding an EAP Request message. This latter message also carries the first authentication message.

- In the present version of this standard, the EAP identity phase is skipped as proposed by [IETF RFC 3748] to directly move to the authentication phase by sending the first message of the selected EAP method.
- The EAP identity phase could be reintroduced later when the need of roaming features arise.

In both cases, these messages are relayed by the LBA to the LBD.

A.3.4.5.2.2.4 Authentication and key distribution phase

The Authentication phase is fully dependant of the EAP method in place. The EAP protocol is very flexible and support various EAP methods (EAP-MD5, EAP-AKA, EAP-TLS, etc.). Each method is characterized by its credentials (shared secret, certificate, SIM cards, etc.) and by its signature and encryption algorithms.

Methods are ordinary based on two round-trip exchanges:

- The first one for mutual authentication and initial exchange of ciphering material; and,
- The second one for mutual control of session keys derivation.

At the end, the LBD shall be equipped with two sets of session keys:

- Transient EAP Key (TEK) for the end-to-end security of EAP messages. These TEKs are generated as described in [IETF RFC 4764].
- Group session keys for a basic PAN security. These keys are shared by all the authenticated nodes in the PAN. Every MAC data frame with SecurityEnabled field set to 1, except those involved in the initial phases of the bootstrapping procedure, is securely transmitted with encryption and decryption at every hop. These Group keys may be refreshed periodically or when a node is detached from the PAN.

Other keys may be derived for additional security services provided at the Application level.

Refer to A.4.5 for further details on the proposed EAP method.

A.3.4.5.2.2.5 Authorization and initial configuration phase

Upon completion of Authentication and Key Distribution process, LBS shall send back an LBP DECLINE message embedding an EAP Failure message if the authentication is failed. This message is relayed by the LBA to the LBD to inform LBD that LBS did not accept the join request of LBD.

If LBS accepts the join request of LBD, it selects a 16-bit short Address, globally defined and fully routable in the PAN and sends back an LBP ACCEPTED message, embedding an EAP Success message. At receipt of this message, the LBD activates the GMK key. A second LBP ACCEPTED message is sent by LBS embedding the global 16-bit short address and optionally other Device Specific and PAN Specific parameters. These messages are relayed by the LBA to the LBD. At this stage, the LBD owns a 16-bit short address and a session key allowing the secure transmission of the messages within the PAN.

At reception of the LBP message, the LBD may set-up an optimized route to the LBS with the help of the LOAD protocol (see A.3.4.4). The path used by device during association phase may be stored in routing tables as an initial route between LBD and LBS without invoking the LOAD protocol.

A.3.4.5.2.2.6 Joining a PAN for any node except coordinator

The network joining procedure is performed by a device which is not a PAN coordinator and does not have a short address (default short address of a device upon reset is 0xffff which means no short address). During this procedure, the device is authenticated, associated to the network and receives GMK and a short address assigned by the coordinator. After a successful Discovery phase as described in clause 7.4.5.2.2.2, this procedure may be triggered by invocation of the ADPM-NETWORK-JOIN.request primitive with PANID and LBAAddress of one of the discovered devices as listed in PANDescriptor of ADPM-DISCOVERY.confirm. The selection criteria of PAN and LBAAddress is implementation specific.

If the join is successful, upper layer is informed by a successful ADPM-NETWORK-JOIN.confirm which includes the assigned short address and PAN ID of the network. In case of failure, this procedure may be repeated after repeating Discovery phase.

If the join procedure is not complete within *adpMaxJoinWaitTime*, a fail confirmation shall be sent to upper layer.

The upper layer in LBS receives the join request of a device as well as subsequent authentication messages as ADPM-LBP.indication primitives with embedded LBP messages. It also sends the authentication messages as well as acceptance and short address embedded in LBP messages to LBD by invoking ADPM-LBP.request. The processing of received LBP messages and construction of LBP messages to be send to LBD is performed in upper layer of LBS. This includes the processing and construction of EAP messages as described in clause A.4.5.

In LBD, all the LBP messages are processed internally and the upper layer is not aware of the message exchanges during authentication procedure. Upon completion or timeout, the upper layer of LBD receives an ADPM-NETWORK-JOIN.confirm with success or failure status.

A.3.4.5.2.2.7 Leaving a PAN – Removal of a device by the PAN coordinator

The PAN coordinator may instruct a device to remove itself from the network invoking the ADPM-LBP.request primitive, using a KICK frame. This frame is a standard LBP message frame with its T field set to 1 and its Code field set to 100b. The bootstrapping data in that message shall be empty.

When a device receives this message, it shall check if the A_LBD field of the LBP message is its own address. If not, the message is silently discarded. Else the device shall perform the following steps:

- Acknowledge the frame if necessary;
- Set its 16-bit short address to 0xFFFF;
- Generate a ADPM-NETWORK-LEAVE.indication containing the 64-bit address of the device;
- Invoke a MLME-RESET.request primitive with the SetDefaultPIB parameter set to TRUE;
- Invoke its ADPM-RESET.request primitive to reset itself.

Figure A.14 describes the messages exchanged during removal of a device from the PAN by the coordinator.

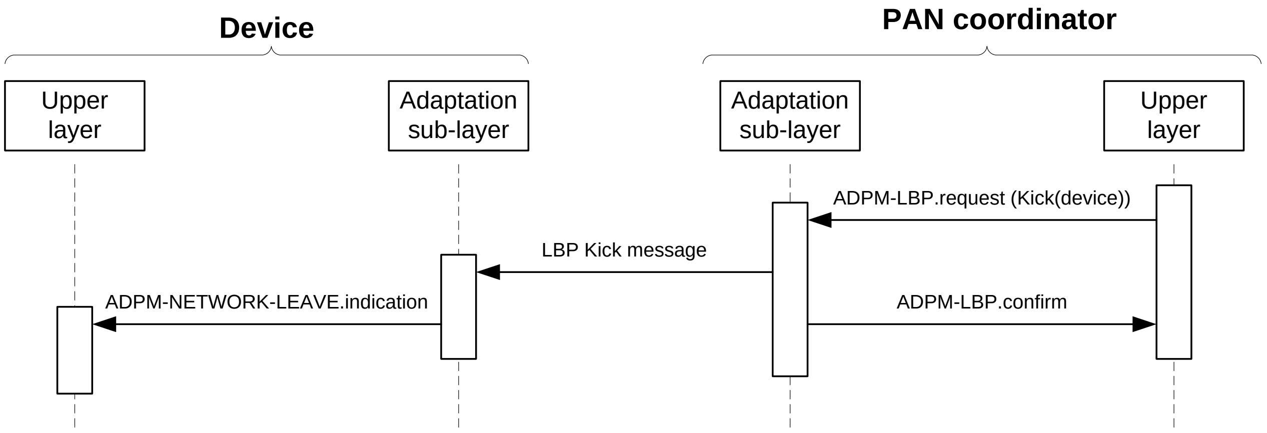


Figure A.14 – Message sequence chart during removal of a device by the coordinator

Upon completion of this procedure, the device shall restart the joining network procedure described in A.3.4.5.2.2.

A.3.4.5.2.2.8 Leaving a PAN – Removal of a device by itself

A device may also call the ADPM-NETWORK-LEAVE.request primitive with ExtendedAddress parameter set to NULL to remove itself from the network, and notify the PAN coordinator about this removal.

If ADPM-NETWORK-LEAVE.request primitive is invoked by a device which is the PAN coordinator, or if the ExtendedAddress parameter is not NULL, then the adaptation sublayer shall issue an ADPM-NETWORK-LEAVE.confirm primitive with the status INVALID_REQUEST.

If ADPM-NETWORK-LEAVE.request primitive is invoked by a device which is not the PAN coordinator and the ExtendedAddress parameter is NULL, then the adaptation sublayer shall:

– Send a KICK frame to the PAN coordinator using an ADPD-DATA.request primitive using a standard LBP message with KICK frame as described in Table 56 with T field set to 0 and its Code field set to 100b. The bootstrapping data in that message should be empty.

– Set its 16-bit short address to 0xFFFF;

– Invoke a MLME-RESET.request primitive with the SetDefaultPIB parameter set to TRUE;

– Invoke its ADPM-RESET.request primitive to reset itself.

Figure A.15 describes the messages exchanged during the removal of a device initiated by the device itself.

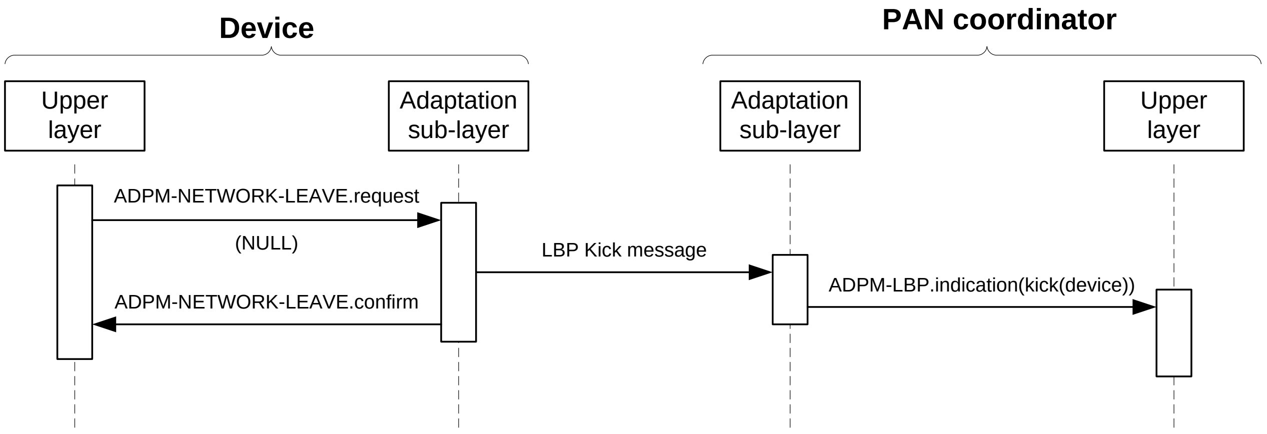


Figure A.15 – Message sequence chart during removal of a device by itself

On the PAN coordinator side, an ADPM-LBP.indication containing the KICK message is generated to inform the upper layers. This message contains the 64-bit address of the device which removed itself from the PAN.

A.3.4.6 Sniffer mode

This mode is used to support monitoring the transmitted packets on the power line. Once activated, the modem will process all packets regardless of their destination address. The sniffer modem shall generate an ADPD-DATA.indication for any received packet. The modem activated in sniffer mode shall not forward packets. If a sniffer modem receives a fragment, it shall add an IPv6 fragment header to the packet so the upper layer can detect it. The fragment offset field and the Identification field shall be set to the offset of the LOWPAN header and the Datagram_Tag respectively.

A.3.5 Functional description

A.3.5.1 Network formation

The network formation can only be performed by the PAN coordinator. Any device other than the PAN coordinator shall not attempt to perform a network formation.

Prior to the network formation, the PAN coordinator shall perform an active scan as described in A.3.4.5.2.2.2. If the PANDescriptorList given by the ADPM-DISCOVERY.confirm primitive is empty, then the PAN coordinator can start a new network. If the PANDescriptorList is not empty, the PAN coordinator may inform the rest of the system that a PAN is already operating in the POS of the device, and may start a new network afterwards. The procedures and decisions associated with this behaviour are implementation specific.

After the network discovery, the PAN coordinator shall set its PAN ID to the predefined value stored in it. This value can be obtained remotely from a configuration server, or locally computed. The way how this PAN ID is chosen and set in the coordinator is implementation specific.

NOTE – The PAN identifier shall be logically ANDed with 0xFCFF, as described in 0 of the present document (clause 6 of [IETF RFC 4944]).

Once the PAN identifier has been determined, the adaptation sublayer shall invoke the MLME-START.request with the following parameters:

- PANId = the PAN identifier computed;
- LogicalChannel = 0 (not used);
- ChannelPage = 0 (not used);
- StartTime = 0 (not used);
- BeaconOrder = 15 (beaconless network);
- SuperframeOrder = 15 (not used);
- PANCoordinator = TRUE;
- BatteryLifeExtension = FALSE (not used);
- CoordRealignment = FALSE;
- CoordRealignSecurityLevel, CoordRealignKeyIdMode, CoordRealignKeySource and CoordRealignKeyIndex: not used, shall be set to 0;
- BeaconSecurityLevel = 0;
- BeaconKeyIdMode, BeaconKeySource, BeaconKeyIndex: not used, shall be set to 0.

The MAC sublayer then generates a MLME-START.confirm primitive with the corresponding status code, which is forwarded to the upper layers through the generation of an ADPM-NETWORK-START.confirm.

A.3.5.2 PAN ID conflict detection and handling

At any time, when a device is associated to a PAN, its MAC sublayer shall analyse the destination and source PAN Identifier in the MAC header of any frame it receives.

If a frame containing a destination or source PAN Identifier is received and does not match the PAN Identifier of the device or the 0xFFFF PAN ID, the frame should be dropped and it shall generate a MLME-SYNC-LOSS.indication primitive with the following characteristics:

- LossReason = PAN_ID_CONFLICT;
- PANId = The conflicting PAN ID;
- LogicalChannel = 0 (not used);
- ChannelPage = 0 (not used);
- SecurityLevel = 0 (not used);
- KeyIdMode, KeySource and KeyIndex can be ignored.

If the adaptation sublayer receives a MLME-SYNC-LOSS.indication primitive with another LossReason than PAN_ID_CONFLICT, it shall ignore it.

In response, the adaptation layer shall generate a CONFLICT frame to its PAN coordinator. This frame is a standard LBP message frame with its Code field set to 101b. The bootstrapping data in that message shall contain the PAN Id of the detected PAN using the format defined in clause 3.3.1 of G3 Annex I and described in Figure A.16:

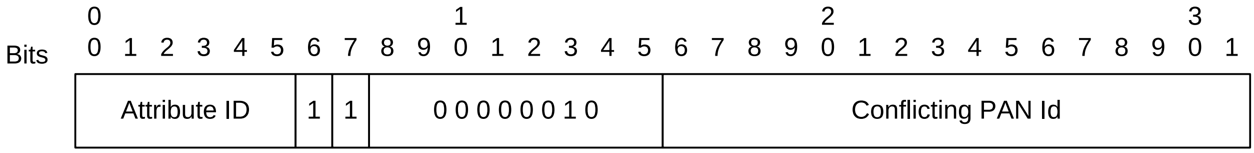


Figure A.16 – CONFLICT message format

This frame is sent to the PAN coordinator with short address of 0x0000 using an ADPD-DATA.request primitive which carries the following attributes:

– NsduLength = the length of the frame;

– Nsdu = the frame;

– NsduHandle = a random number;

– DiscoverRoute = TRUE;

– QualityOfService = 0;

– SecurityEnabled = TRUE.

This shall be translated to MCPS-DATA.request with:

– DstAddrMode = 0x02;

– DstAddr = the short address of coordinator 0x0000;

A device shall wait adpPANConflictWait seconds between two consecutive sending of a CONFLICT frame for the same conflicting PAN Id, and the total number of CONFLICT frames sent for a given conflicting PAN Id shall not exceed adpMaxPANConflictCount. When this value is reached, the device shall stop sending CONFLICT frames for this conflicting PAN Id.

When the PAN coordinator receives this frame, it shall generate an ADPM-NETWORK-STATUS.indication primitive to the upper layer, with:

- The Status field sets to PAN_ID_CONFLICT; and

- The AdditionalInformation field sets to the conflicting PAN Id.

A.4 Security

A.4.1 Access control and authentication

An End Device (ED) may not access to the network without a preliminary Identification (with comparison to white or black lists) and Authentication. Identification and Authentication are based on two parameters that personalized every ED:

- A EUI-48 MAC address as defined in IEEE 802. This address may be easily converted into a EUI-64 as requested by [IEEE 802.15.4] and related documents;
- A 128-bit shared secret (aka Pre-Shared Key or PSK) used as a credential during the authentication process. It is shared by the ED itself (aka peer) and an authentication server. The mutual authentication is based on a proof the other party knows the PSK. It is of highest importance, the PSK remains secret.

The Identification and Authentication processes are activated when an ED restarts and may also be launched at any time according to the security policy in place. The related material is carried by the 6LoWPAN Bootstrapping Protocol (LBP) (see A.3.4.5) that embeds the Extensible Authentication Protocol (EAP) (see A.3.4.5.2.1.2).

As shown in Figure A.17, LBP and EAP have been designed to be relayed by intermediates nodes. Then during the Bootstrapping phase, when an ED (aka LBD) that have not yet acquired a routable 16-bit address, is a 1-hop distance of the PAN Coordinator (aka LBS) they can directly communicate. Otherwise, they shall use an intermediate node (aka LBA) located `at 1-hop distance of the LBD.

Moreover, two different authentication architectures shall be considered:

- The authentication server function is directly supported by the LBS, and in this case all the authentication material (access lists, credentials, etc.) shall be loaded in the LBS.
- The authentication server function is supported by a remote (and usually centralized) AAA server, and in this case, the LBS is only in charge of forwarding the EAP messages to the AAA server over a standard AAA protocol (i.e., RADIUS RFC 2865).

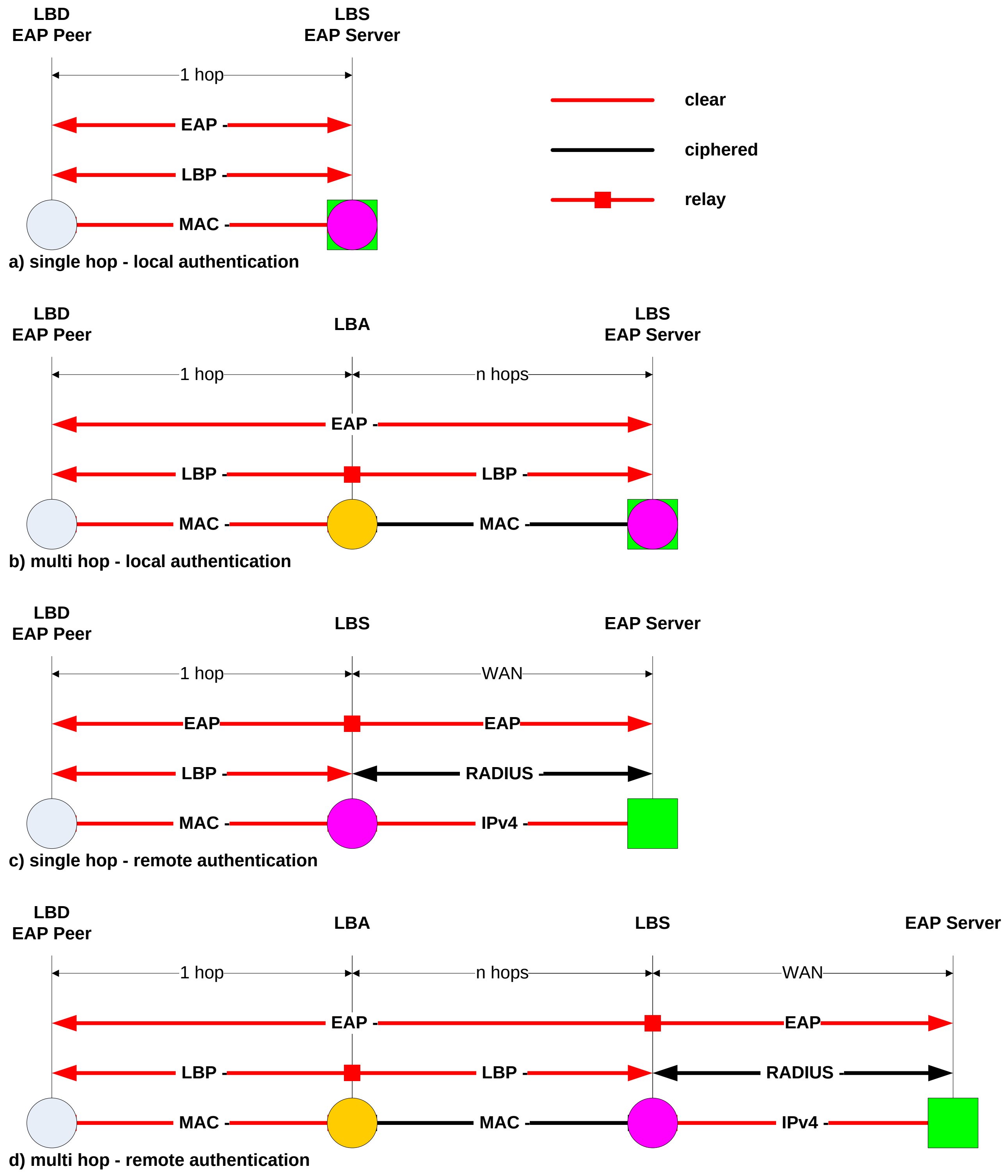


Figure A.17 – LBP and EAP Relaying Capabilities

The Authentication process is fully dependant of the EAP method in place. The EAP protocol is very flexible and support various EAP methods (EAP-MD5, EAP-AKA, EAP-TLS, etc.). Each method is characterized by its credentials (shared secret, certificate, SIM cards, etc.) and by its signature and encryption algorithms.

The method adopted for the OFDM CPL network is EAP-PSK (see A.4.5), the main design goals of which are:

- Simplicity: it is entirely based on a single credential (a 128-bit Pre-Shared Key) and a single cryptographic algorithm (AES-128);

- Security: it appears very conservative in its design following well-known and improved cryptographic schemes.
- Extensibility: in the OFDM CPL case, it is easily extended to support Group Key distribution (see A.4.5.2).

A.4.2 Confidentiality and integrity

As shown by Figure A.18, confidentiality and integrity services are ensured at different levels:

- At MAC level: as defined in [IEEE 802.15.4], a CCM* type of ciphering is delivered to every frame transmitted between nodes in the network. It's a universal Low Layer Confidentiality and Integrity service (with anti-replay capabilities). The MAC frames are encrypted and decrypted at every hop. The only exceptions are some well-controlled frames in the early stages of the Bootstrapping process. To fairly support this service, all the nodes in the network receive the same Group session key (GMK). This GMK is individually and securely distributed to every node by using the EAP-PSK Secure Channel.

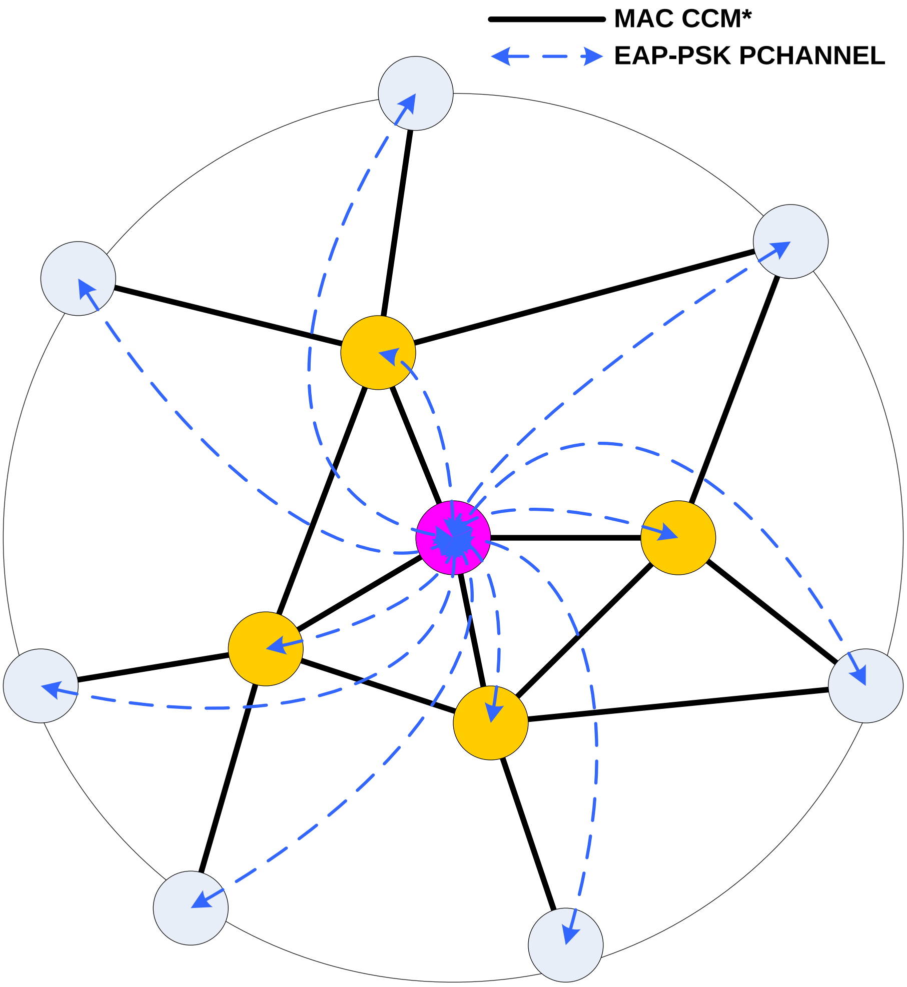


Figure A.18 – Confidentiality and Security

- At the EAP-PSK level: as defined in [IETF RFC 4764], EAP-PSK provides Confidentiality and Integrity (and Replay Protection) services, also known as Protected Channel (PCHANNEL) to the messages exchanged over EAP between the EAP server and any peer.

A.4.3 Anti-Replay and DoS prevention

It is always difficult to prevent DoS attacks, and especially those targeting the Physical level, but by nature their impact is limited to a small area.

The CCM* ciphering mode is generalized at MAC layer. It prevents unauthenticated devices accessing the network and having malicious actions on routing, provisioning and any other Low Layer processes. The only exception is the well-controlled Bootstrapping process.

Moreover, an anti-replay mechanism is specified at the MAC sublayer.

A.4.4 Authentication and key distribution protocol – Selections from [IETF RFC 3748]

Authentication and key distribution are supported by the Extensible Authentication Protocol (EAP) as given in [IETF RFC 3748] together with the selections listed in Table A.31.

Table A.31 – Selections from [IETF RFC 3748]

Clause	Title & remarks/modifications	Statement
1.	Introduction	N
2.	Extensible Authentication Protocol (EAP) – Initial Identity Request (allow roaming and EAP method negotiation) is let for further study and shall be bypassed.	S
2.1	Support for Sequences	N
2.2	EAP Multiplexing Model – Only one EAP method is defined (cf. A.3.4.5)	S
2.3	Pass-Through Behaviour – Over LBP, the Code field is slightly different from a regular EAP Code field as specified in [IETF RFC 3748]. The conversion appears straightforward in both directions. The proper conversion shall apply when the EAP message is propagated over another protocol (i.e., RADIUS) and in case of integrity protection covering the EAP header	S
2.4	Peer-to-Peer Operation	N
3.	Lower Layer Behaviour	N
3.1	Lower Layer Requirements – LBP and underlying protocols provide: – Reliable transport – Error detection (CRC) – No Lower Layer security when bootstrapping – MTU size greater than 1 020 octets (by fragmentation) – No duplication – Ordering guaranties	S
3.2	EAP Usage Within PPP	N/R
3.3	EAP Usage Within IEEE 802	N/R
3.4	Lower Layer Indications	N
4.	EAP Packet Format – Over LBP, the Code field is slightly different from a regular EAP Code field.	S
4.1	Request and Response – Over LBP, the Code field is slightly different from a regular EAP Code field.	S
4.2	Success and Failure – Over LBP, the Code field is slightly different from a regular EAP Code field.	S

Table A.31 – Selections from [IETF RFC 3748]

Clause	Title & remarks/modifications	Statement
4.3	Retransmission Behaviour	N
5.	Initial EAP Request/Response Types – For the Type field, the only available values are 3 (Nak – in Response only) and the value assigned to the EAP method (see A.4.5). Other values are left for further study	S
5.1.	Identity	N/R
5.2.	Notification	N/R
5.3.	Nak	N
5.4.	MD5-Challenge	N/R
5.5.	One-Time Password (OTP)	N/R
5.6.	Generic Token Card (GTC)	N/R
5.7.	Expanded Types	N/R
5.8.	Experimental	N/R
6.	IANA Considerations	N
7.	Security Considerations	N
8.	Acknowledgements	I
9.	References	N
Appendix A.	Changes from [IETF RFC 2284]	I

A.4.5 EAP method

The EAP protocol is very flexible and support various EAP methods (EAP-MD5, EAP-AKA, EAP-TLS, etc.). Each method is characterized by its credentials (shared secret, certificate, SIM cards, etc.) and by its signature and encryption algorithms.

For the OFDM CPL case, the recommended method is Pre-Shared Key EAP Method (EAP-PSK) as given in [IETF RFC 4764] together with the selections listed in Table A.32.

Table A.32 – Selections from [IETF RFC 4764]

Clause	Title & remarks/modifications	Statement
1.	Introduction	N
2.	Protocol Overview	N
3.	Cryptographic Design of EAP-PSK	N
4.	EAP-PSK Message Flows – EAP-PSK extension capabilities are used for Group Key distribution in full compliance to [IETF RFC 4764]. See A.4.5.2	N
5.	EAP-PSK Message Format – EAP-PSK extension capabilities are used for Group Key distribution in full compliance to [IETF RFC 4764]. See A.4.5.2	N
6.	Rules of Operation for EAP-PSK Protected Channel	N
7.	IANA Considerations	N
8.	Security Considerations	N
9.	Security Claims	I

Table A.32 – Selections from [IETF RFC 4764]

Clause	Title & remarks/modifications	Statement
10.	Acknowledgements	I
11.	References	N
Appendix A.	Generation of the PSK from a Password – Discouraged	N/R

A.4.5.1 Overview of EAP-PSK

EAP-PSK, according to the EAP specification supports the following key hierarchy:

Pre-Shared Key (PSK)	PSK is the long-term 128-bit credential shared by the EAP server and the peer
Authentication Key (AK)	A 128-bit key derived from the PSK that the EAP peer and server use to mutually authenticate
Key-Derivation Key (KDK)	A 128-bit key derived from the PSK that the EAP peer and server use to derive session keys (such as TEK, MSK and EMSK)
Transient EAP Key (TEK)	A session key that is used to establish a protected channel between the EAP peer and server during the EAP authentication. EAP-PSK uses a 128-bit TEK in conjunction with AES-128 in EAX mode of operation as a cipher suite
Master Session Key (MSK)	A session key derived between the EAP peer and server. EAP-PSK generates a 512-bit MSK that may be used to provide security at the Application level
Extended Master Session Key (EMSK)	A session key derived between the EAP peer and server. EAP-PSK generates a 512-bit EMSK. It is not used in OFDM CPL and shall not be generated.

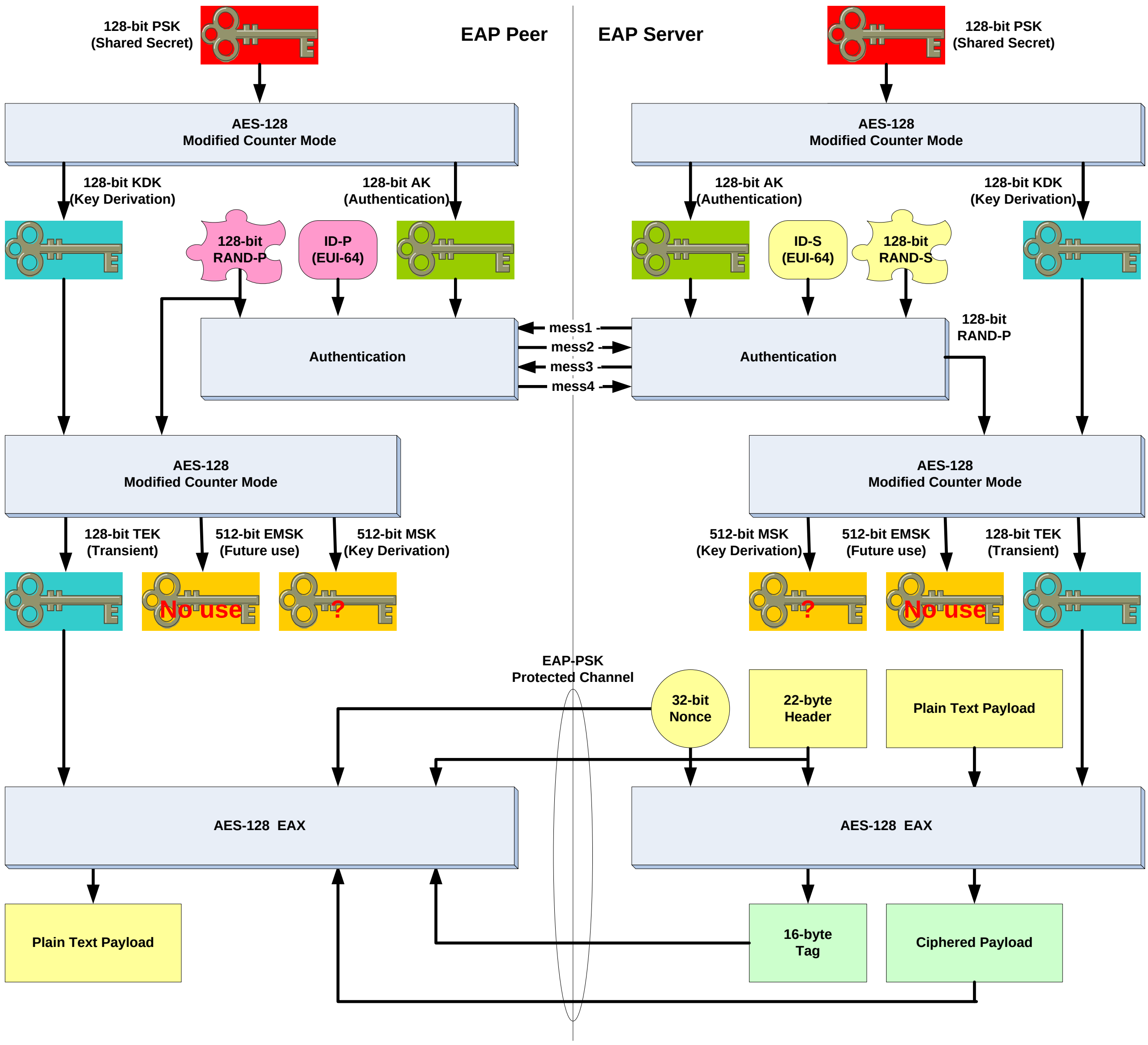


Figure A.19 – EAP-PSK key hierarchy overview

A.4.5.2 Group key distribution

The 128-bit Group Master Key (GMK) is generated by the EAP Server. Then it is securely and individually delivered to the EAP peers via the EAP-PSK Protected Channel (PCHANNEL).

GMK is assumed being random. GMK generation is considered as purely implementation dependant.

GMK is distributed to the peer in two circumstances:

- During the Bootstrapping process, carried as a regular extension to EAP-PSK message 3 of the Figure A.19.
- During the Re-keying process, carried as a regular extension to EAP-PSK message 5 of the Figure A.19. The GMK lifetime is rather long (several 10s years) due to the 4 byte counter included in the nonce. Nevertheless it's of good policy to timely re-key the network or when a node is leaving it.

A.4.5.3 GMK field format

The GMK field in message 3 or 5 of the Figure A.19 is defined in compliance with the generic extension field (EXT) (see [IETF RFC 4764], clause 5.3.) described in Figure A.20.

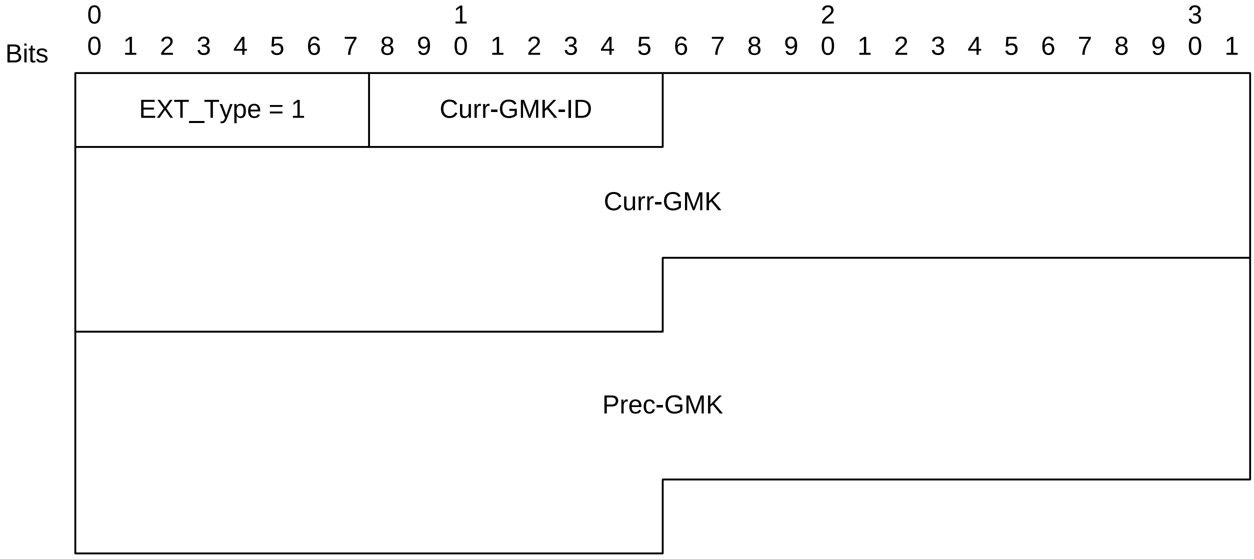


Figure A.20 – GMK field format for messages 3 and 5

Where

EXT_Type The EXT_TYPE field is one octet and indicates the type of the Extension

1 GMK

Curr-GMK-ID The Curr-GMK-ID field is one octet and represents the Key Identifier of the current GMK.

Curr-GMK The Curr-GMK is 16 octets and contains the value of the current GMK.

Prec-GMK The Prec-GMK is 16 octets and contains the value of the preceding GMK.

A.4.5.4 Peer side procedure

Once a peer receives GMK field embedded in a message 3 (in case of Bootstrapping) or a message 5 (in case of Re-keying), it shall install Curr-GMK in a table at Curr-GMK-ID index and Prec-GMK at its corresponding index location (the Curr-GMK-ID of the last re-keying procedure) and shall process and respond to the message according to [IETF RFC 4764].

Both keys are immediately available to decrypt a received packet using the key identified by the Key Identifier contained in the MAC header of the received frame.

In case of Bootstrapping, the peer keeps sending the frames in clear text up to the reception of an EAP Success message. Then, it starts sending ciphered frames using the Curr-GMK.

In case of Re-keying, the peer keeps sending messages according to the previously assigned policy until reception of an EAP Success message. Then, it starts sending frames using the current GMK.

In case the peer did not receive the EAP Success message after a *adpUseNewGMKTime*, it may start using Curr-GMK to send frames as soon as it receives a packet with the Key Identifier of Curr-GMK which indicates the server had already invoked switching to the Curr-GMK in the network.

After switching to the Curr-GMK, a peer may keep receiving some messages encrypted with the Prec-GMK during a transient period. The Prec-GMK may be deleted after *adpExpPrecGMKTime* delay which shall be long enough to make sure that all devices in the network have switched to Curr-GMK.

A.4.5.5 Server side procedure

The Bootstrapping procedure is defined in A.3.4.5.2.2.

In case of re-keying, the EAP server generates a new GMK. Then it transmits a LBP challenge message, embedding an EAP Request message that contains the newly generated GMK as Curr-GMK and the previous GMK as Prec-GMK, to every formerly associated peer. The Curr-GMK-ID shall be chosen to be different from Key Identifier associated with Prec-GMK.

If the EAP server does not receive the EAP response to the LBP challenge message from a peer, it may retry the re-keying procedure described above. The PAN coordinator may remove the peer device using the procedure described in A.3.4.5.2.2.7 if EAP server failed to receive the response.

Once the EAP server has received the EAP response to the LBP challenge message from all peers, it may start sending EAP Success messages to the peers to invoke switching to the new GMK.

G3 Annex A

(normative)

Protocol implementation conformance statement

(This annex forms an integral part of this Recommendation.)

A.A.1 Overview

Compliance with clauses of [IEEE 802.15.4] shall be consistent with the extensions and selections defined in this annex.

The first part of this annex entirely takes as reference the protocol implementation conformance statement of [IEEE 802.15.4], Annex D.

The second part of this annex gives similar tables to ensure that all items related to the physical layer of G3-PLC have been taken into account.

A.A.2 PICS proforma tables

A.A.2.1 Functional device types (from Annex D.7.1 of [IEEE 802.15.4])

Table A.A.1 – PICS – Functional device types (from annex D.7.1 of [IEEE 802.15.4])

Item number	Support			Comments
	N/A	Yes	No	
FD1		X		
FD2			X	
FD3		X		
FD4		X		
FD5		X		

A.A.2.2 PHY functions (from Annex D.7.2.1 of [IEEE 802.15.4])

Table A.A.2 – PICS – PHY functions (from Annex D.7.2.1 of [IEEE 802.15.4])

Item number	Support			Comments
	N/A	Yes	No	
PLF1		X		
PLF2		X		
PLF3	X			Radio specific requirement
PLF4	X			Radio specific requirement
PLF5	X			Radio specific requirement
PLF6		X		
PLF7	X			Radio specific requirement
PLF8		X		

Item number	Support			Comments
	N/A	Yes	No	
PLF8.1	X			Radio specific requirement
PLF8.2		X		
PLF8.3	X			Radio specific requirement

A.A.2.3 PHY packet (from annex D.7.2.2 of [IEEE 802.15.4])

Table A.A.3 – PICS – PHY packet (from Annex D.7.2.2 of [IEEE 802.15.4])

Item number	Support			Comments
	N/A	Yes	No	
PLP1		X		

A.A.2.4 Radio frequency (from Annex D.7.2.3 of [IEEE 802.15.4])

Table A.A.4 – PICS – Radio frequency (from Annex D.7.2.3 of [IEEE 802.15.4])

Item number	Support			Comments
	N/A	Yes	No	
RF1	X			Radio specific requirement
RF1.1	X			Radio specific requirement
RF1.2	X			Radio specific requirement
RF1.3	X			Radio specific requirement
RF1.4	X			Radio specific requirement
RF2	X			Radio specific requirement

A.A.2.5 MAC sublayer functions (from Annex D.7.3.1 of [IEEE 802.15.4])

Table A.A.5 – PICS – MAC sublayer functions (from Annex D.7.3.1 of [IEEE 802.15.4])

Item number	Support			Comments
	N/A	Yes	No	
MLF1		X		
MLF1.1			X	Indirect transmission is not supported
MLF2		X		
MLF2.1		X		
MLF2.2		X		
MLF2.3		X		
MLF3		X		
MLF3.1		X		
MLF3.2		X		
MLF4		X		
MLF5			X	

Table A.A.5 – PICS – MAC sublayer functions (from Annex D.7.3.1 of [IEEE 802.15.4])

Item number	Support			Comments
	N/A	Yes	No	
MLF5.1			X	
MLF5.2			X	
MLF6		X		
MLF7		X		
MLF8			X	Performed by 6LoWPAN
MLF9		X		
MLF9.1		X		
MLF9.2		X		
MLF9.2.1		X		
MLF9.2.2		X		
MLF10.1	X			Radio specific requirement
MLF10.2		X		
MLF10.3			X	Not necessary for non beacon-enabled networks
MLF10.4			X	
MLF11			X	
MLF12			X	
MLF13			X	

A.A.2.6 MAC frames (from Annex D.7.3.2 of IEEE 802.15.4)

Table A.A.6 – PICS – MAC frames (from Annex D.7.3.2 of [IEEE 802.15.4])

Item number	Support						Comments
	Transmitter			Receiver			
	N/A	Yes	No	N/A	Yes	No	
MF1		X			X		
MF2		X			X		
MF3		X			X		Acknowledgement frames are described in [ITU-T G.9955]/Annex A and G3 Annex E of the present document
MF4		X			X		
MF4.1			X			X	Association performed by 6LoWPAN
MF4.2			X			X	Association performed by 6LoWPAN
MF4.3			X			X	Association performed by 6LoWPAN

MF4.4			X			X	No transaction support
MF4.5			X			X	Performed by 6LoWPAN
MF4.6			X			X	
MF4.7		X			X		
MF4.8			X			X	
MF4.9			X			X	

G3 Annex B

(informative)

Routing cost

(This annex forms an integral part of this Recommendation.)

This part describes the characteristics a routing cost used in the LOAD routing algorithm (described in G3 Annex H and in A.3.4.4) shall have.

A route cost is defined as the sum of all the link costs on the route. As described in G3 Annex H, a route cost is an integer value between 0 and 255, lower values meaning better routes. While the link cost computation algorithm is implementation dependant, the following formula may be used:

MOD_Kr = 1 for ROBO, 0 for other modulations

MOD_Km = 3 for ROBO, 2 for DBPSK, 1 for DQPSK and 0 for D8PSK

Link Cost = AdpKr* MOD_Kr

+ AdpKm* MOD_Km

+ AdpKc *(Maximum Number of Tones – number of active tones)/Maximum Number of Tones

+ AdpKq * (Maximum LQI – LQI)/Maximum LQI

+ AdpKh * 1

+ AdpKrt * number of active routes/Maximum number of active routes

If we note P a route which goes through devices $\{D_0, D_1, ..., D_{N-1}\}$, where N is the number of hops on the route ($0 < N \le 8$), and $C\{[D_i, D_j]\}$ the link cost between devices D_i and D_j, the route cost $RC(P)$ of P can then be defined as:

$$RC(P) = \sum_{i=0}^{N-1} C\{[D_i, D_{i+1}]\}$$

The link cost may take into account PHY transmission parameters, number of hops, etc. The link cost computation algorithm is implementation dependant.

G3 Annex C

(normative)

Channel access

(This annex forms an integral part of this Recommendation.)

A.C.1 Overview

The channel access is accomplished by using the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) mechanism with a random backoff time. The random backoff mechanism spreads the time over which stations attempt to transmit, thereby reducing the probability of collision. Each time a device wishes to transmit data frames, it shall wait for a random period. If the channel is found to be idle, following the random backoff, the device shall transmit its data. If the channel is found to be busy, following the random backoff, the device shall wait for another random period before trying to access the channel again.

A Carrier sense is a fundamental part of the distributed access procedure. Physical Carrier Sense (PCS) is provided by the PHY as described in clause A.7.2.8 of [ITU-T G.9955]. In the latter case, PCS shall stay high long enough to be detected and Virtual Carrier Sense (VCS) to be asserted by the MAC. A virtual carrier sense mechanism is provided by the MAC by tracking the expected duration of channel occupancy. Virtual carrier sense is set by the length of received packet or upon collision. In these cases, virtual carrier sense tracks the expected duration of the Busy state of the medium. The medium shall also be considered Busy when the station is transmitting.

A VCS timer is maintained by all stations to improve reliability of channel access. The VCS timer is set based on received long (data) or short (ACK) frames. The VCS timer is also set upon collision or when the station powers up. Stations use this information to compute the expected Busy condition of the medium or the expected duration of the Contention State and store this information in the VCS timer.

A Collision occurs in each of the following circumstances:

– The transmitting station receives a something other than ACK or NACK response when a response is expected.

– The transmitting station shall infer a Collision from the absence of any response to a transmission when a response is expected. Note that the absence of a response could also be the result of a bad channel. Since there is no way to distinguish between the two causes a Collision is inferred.

A.C.2 Interframe (IFS) spacing

A time intervals between frames on the medium constitute the Interframe Space and are necessary due to propagation and processing time. As shown in Figure A.C.1, three interframe space values are defined. Contention Interframe Space (CIFS) occurs after the end of the previous transmission. The second defined interval is the Response Interframe Space (RIFS).

RIFS is the time between the end of a transmission and the start of its associated response. If no response is expected, the CIFS is in effect.

An Extended Interframe Space (EIFS) is defined for conditions when the station does not have complete knowledge of the state of the medium. This can occur when the station initially attaches to the network, when errors in the received frames make them impossible to decode unambiguously. If a packet is received and correctly decoded before the expiration of the EIFS, then the EIFS is cancelled. The EIFS is significantly longer than the other interframe spaces, providing protection from Collision for an ongoing frame transmission or segment burst when any of these conditions occur. The EIFS is calculated as follows:

$$aEIFS = aSymbolTime * (N_{FCH} + aMaxFrameSize + aCIFS + aRIFS) + aAckTime$$

where N_{FCH} is the number of symbols in the Frame Control Header (FCH) – see [ITU-T G.9955].

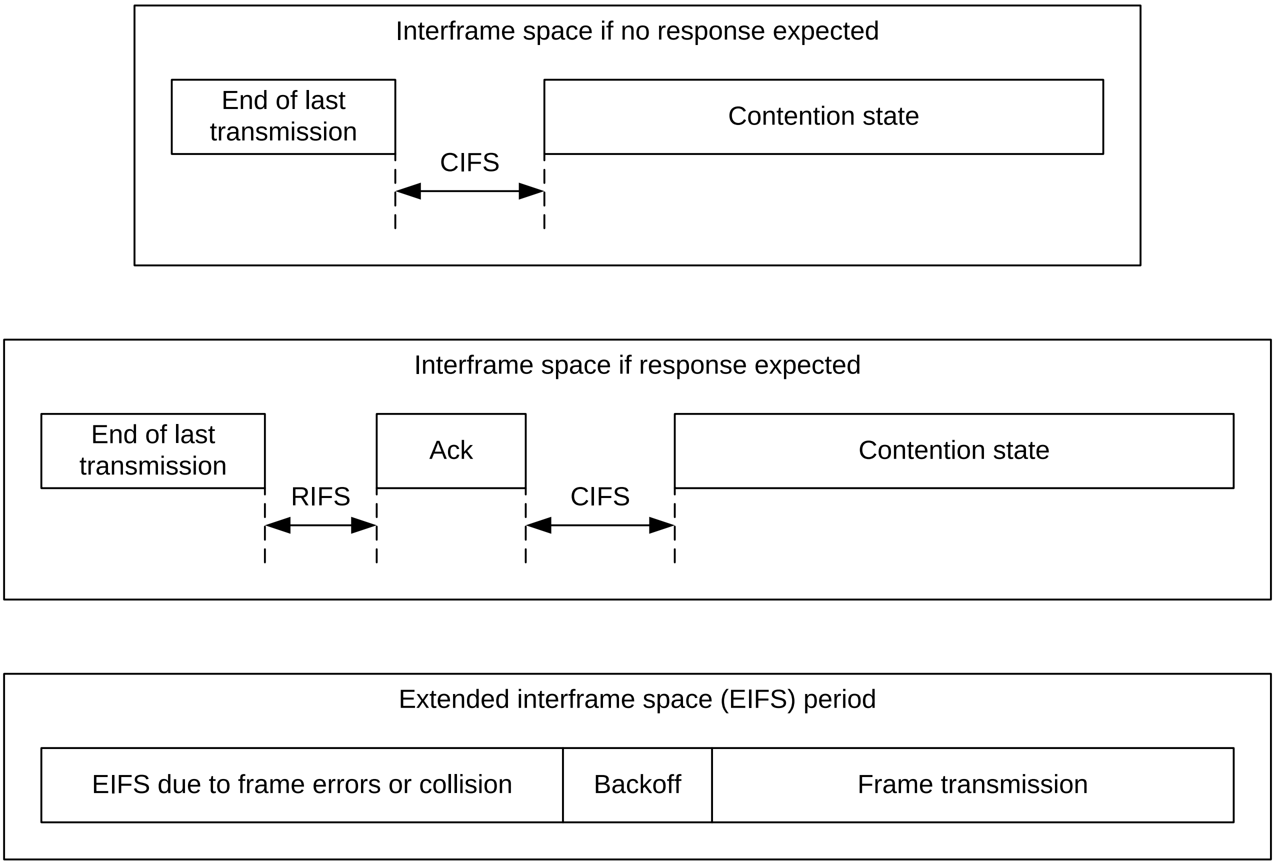

Interframe space if no response expected
End of last transmission
Contention state
CIFS
Interframe space if response expected
End of last transmission
Ack
Contention state
RIFS
CIFS
Extended interframe space (EIFS) period
EIFS due to frame errors or collision
Backoff
Frame transmission

Figure A.C.1 – IFS

A.C.3 CSMA-CA

The present specification supports only an unslotted version of the CSMA-CA algorithm for non-beacon PAN described in [IEEE 802.15.4].

The random backoff mechanism spreads the time over which stations attempt to transmit, thereby reducing the probability of collision, using a truncated binary exponential backoff mechanism.

The CSMA-CA algorithm shall be used before the transmission of data or MAC command frames

The algorithm is implemented using units of time called backoff periods, where one backoff period shall be equal to u*nitBackoffPeriod* symbols.

Each device shall maintain two variables for each transmission attempt: ***N*B** and ***B*E**. ***NB*** is the number of times the CSMA-CA algorithm has been used as backoff while attempting the current transmission; this value shall be initialized to 0 before each new transmission attempt.

BE is the backoff exponent, which is related to how many backoff periods a device shall wait before attempting to assess a channel. ***BE*** shall be initialized to the value of ***minB*E**.

Note that if ***minBE*** is set to 0, collision avoidance will be disabled during the first iteration of this algorithm. Figure A.C.2 illustrates the steps of the CSMA-CA algorithm. The MAC sublayer shall first initialize ***NB****,* and ***BE*** [step (1)] and then proceed directly to step (2).

The MAC sublayer shall delay for a random number of complete backoff periods in the range 0 to $2^{BE} -1$ [step (2)] and then request that the PHY perform a PCS (Physical Carrier Sense) [step (3)].

$$\text{Backoff Time} = \text{Random}(2^{BE} -1) \times \text{aSlotTime}$$

If the channel is assessed to be busy [step (4)], the MAC sublayer shall increment both ***NB*** and ***BE*** by one, ensuring that ***BE*** shall be no more than ***maxB*****E**.

NOTE – For high priority packets ***maxB*****E** shall be equal to ***minBE***.

If the value of ***NB*** is less than or equal to ***maxCSMABackoff*****s**, the CSMA-CA algorithm shall return to step (2).

If the value of ***NB*** is greater than ***maxCSMABackoff*****s**, the CSMA-CA algorithm shall terminate with a Channel Access Failure status.

If the channel is assessed to be idle [step (5)], the MAC sublayer shall begin transmission of the frame immediately.

In order to improve the fairness, ***BE*** shall be reduced to ***minBE*** if *mod*(***NB, macCSMAFairnessLimit)****=0* where *mod* is modulo operation. For example if ***macCSMAFairnessLimit*** = 15 and ***maxCSMABackoffs*** = 50 as soon as the number of backoffs reaches 15, 30 and 45, the ***BE*** should be reduced to **minBE**.

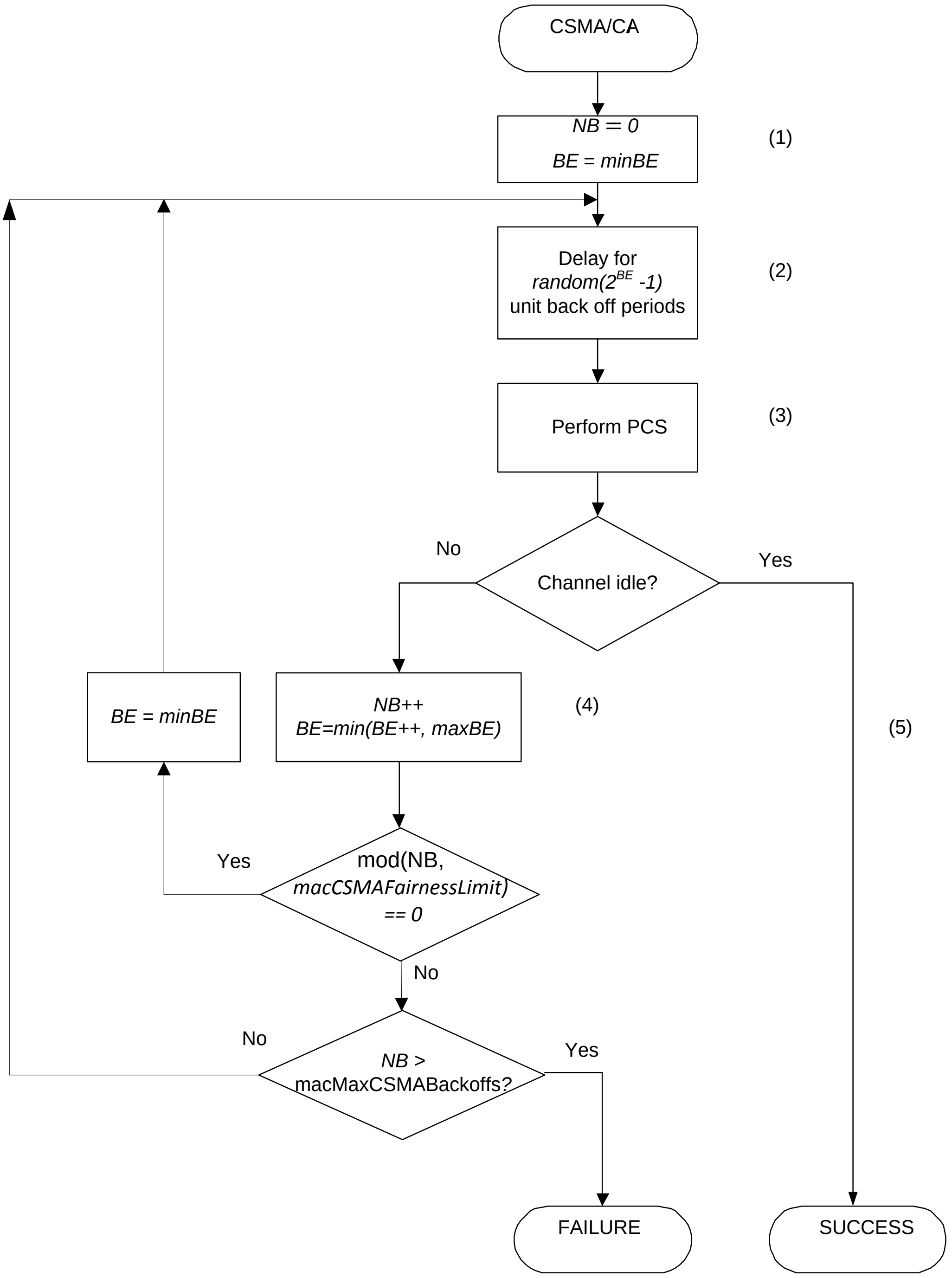


Figure A.C.2 – CSMA/CA algorithm

A.C.4 Priority

Prioritized access to the channel can be beneficial for real time application or control application when urgent message shall be delivered as soon as possible. Only two levels of priority (High and Normal) will be used to minimize complexity. Priority resolution is implemented by using two contention time windows during contention state as shown in Figure A.C.3:

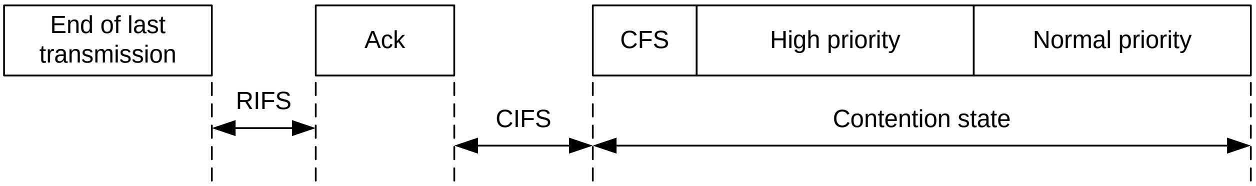


Figure A.C.3 – Priority Contention Windows

First slot of contention window is called Contention Free Slot (CFS). The contention free slot shall be used for transmission of subsequent segments of a MAC packet without backoff procedure to prevent possible interruption from other nodes and to simplify the MAC packet reassembly procedure on a receiver. In this case, only the first segment is sent using either normal or high priority contention window and the rest are sent using the contention free slot.

The high and normal priority stations will compete for channel during HPCW and NPCW correspondingly. Since HPCW is located before NPCW high priority stations will get access to the channel before station with normal priority. Duration of HPCW and NPCW are calculated as follow:

HPCW time = macHighPriorityWindowSize * aSlotTime;

NPCW time = (2^{maxBE} * *aSlotTime)* – HPCW time;

CFS time = *aSlotTime*

A.C.5 ARQ

ARQ (Automatic Repeat reQuest) is implemented based on acknowledged and unacknowledged retransmission. The MAC sublayer uses a response type as part of its ARQ mechanism. ACK is a traditional positive acknowledgment that when received allows the transmitter to assume successful delivery of the frame. The negative acknowledgment (NACK) is used to inform a packet originator that the receiver received the packet but it was corrupted.

A successful reception and validation of a data can be confirmed with an acknowledgment. If the receiving device is unable to handle the received data frame for any reason, the message is not acknowledged.

If the originator does not receive an acknowledgment after waiting period, it assumes that the transmission was unsuccessful and retries the frame transmission. If an acknowledgment is still not received after several retries, the originator can choose either to terminate the transaction or to try again. When the acknowledgment is not requested, the originator assumes the transmission was successful. Also if acknowledgment is not requested the originator can retransmit the same packets few times to increase probability of data delivery. The receiver shall be able distinguish and discard redundant copies using the Sequence Number and Segment Count. The retransmitted packet will have the same Sequence Number and Segment Count as original.

The acknowledgment cannot be requested for broadcast or multicast transmission. On transmit side ARQ shall configure the number of retransmissions (cf. macMaxFrameRetries from 7.4.2 of [IEEE 802.15.4]) as shown in Figure A.C.4.

On receive side ARQ generates acknowledgement for PLC packet with correct FCS (CRC16) if packet corresponds to this address as shown in Figure A.C.5.

The received packet FCS (16 bit) will be sent back to the packet originator as a part of an acknowledgement (Frame Control Header).

All nodes will detect ACK during response time but only one station expecting ACK will accept it as acknowledgement and use 16 bit of FCS from ACK for identification.

MAC acknowledgement is described in details in G3 Annex E.

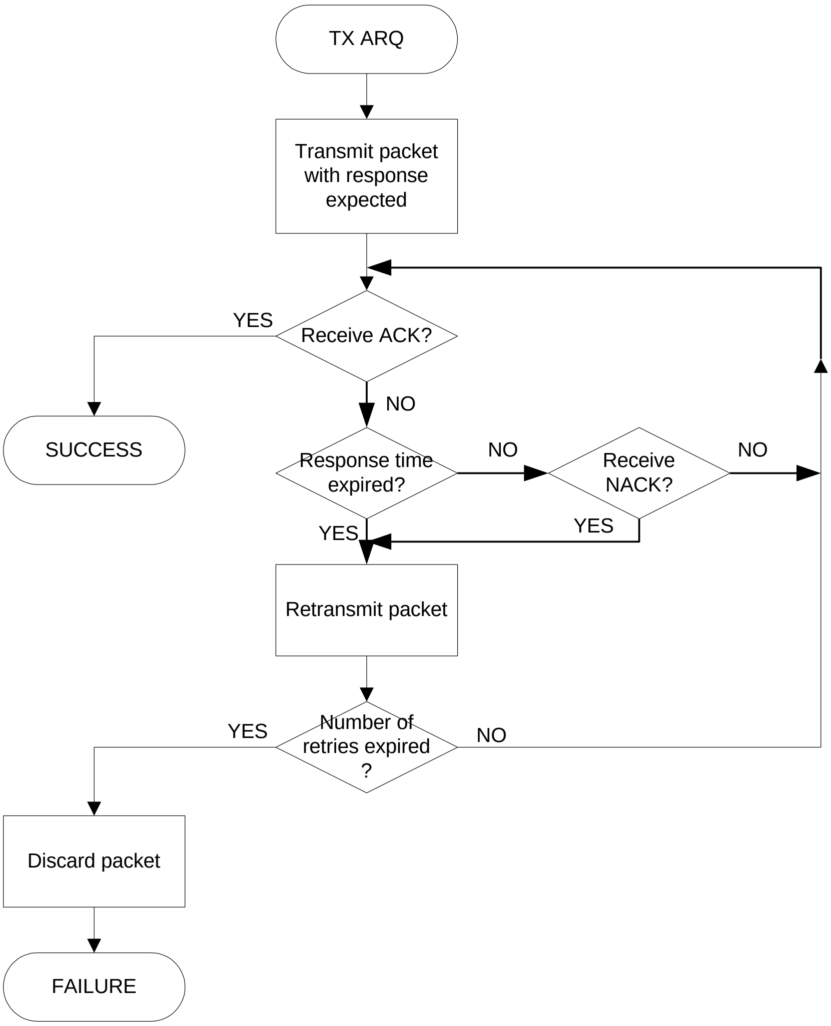


Figure A.C.4 – Transmit ARQ

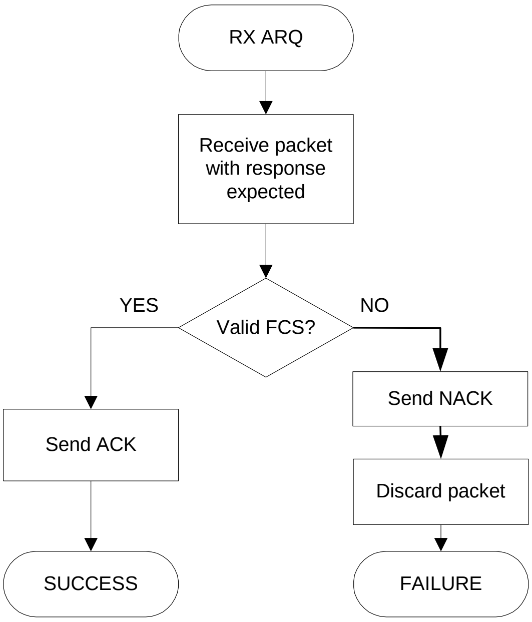


Figure A.C.5 – Receive ARQ

A.C.6 Segmentation and reassembly overview

The G3 PHY layer supports different types of modulation and tone map. Number of data bytes of the PHY payload can change dynamically, based on channel condition. This requires implementing MAC payload fragmentation on MAC sublayer. If the size of MAC payload plus the MAC header is too large to fit within one PSDU, it must be partitioned into smaller segments that each can fit within a PSDU. This process of partitioning MAC frame into PSDUs is called segmentation and the reverse process is called reassembly. The segmentation may require the addition of padding bytes to the last segment to fit last PHY frame. The acknowledgement and retransmission occur independently for resulting MAC segment. All forms of addressing (unicast and broadcast) are subject to the segmentation.

The Segment Control field definition are shown in Table A.5:

For a packet that requires setting the TMR bit and segmentation, the TMR bit shall be set in the last segment only.

Last Segment Flag (LSF) shall be set to 1 to indicate the last segment of MAC packet.

Segment Count (SC) shall be set to 0 for the first segment and incremented for each following segment.

Segment length (SL) specifies the length of the MAC payload in bytes for current segment excluding MAC header, byte padding and FCS. When security is activated, the MAC payload is constituted of the ciphered payload and the MIC-32.

If segmentation is required to transmit a MAC packet, each resulting MAC frame shall be created as follow:

– MAC header (MHR) and FCS (MFR) are presented in each segment.

– The first and the following segments have the same value of the Sequence Number assigned for the MAC packet. Only Segment Count is incremented for following segments.

– If data encryption is required it must be done before packet segmentation. On receiver side data decryption is done after packet reassembly.

- All segments except the last one shall set the Contention Control (CC) bit to inform the receiver that next PHY frame will be sent in contention free slot. The last segment clears the Contention Control bit to allow the normal contention access to the channel.

The Segment Control fields (see Table A.5): **SL**, **SC** and **LSF** are used to keep track of segments of fragmented MAC packet and assembly whole packet on receiver side.

G3 Annex D

(normative)

Modified MAC sublayer data primitives

(This annex forms an integral part of this Recommendation.)

A.D.1 MCPS-DATA.request

The semantic of the MCPS-DATA.request primitive is as follows:

MCPS-DATA.request(
SrcAddrMode,
DstAddrMode,
DstPANId,
DstAddr,
msduLength,
msdu,
msduHandle,
TxOptions,
SecurityLevel,
KeyIdMode,
KeySource,
KeyIndex,
QualityOfService
)

Table A.D.1 specifies the parameters for the MCPS-DATA.request primitive.

Table A.D.1 – MCPS-DATA.request parameters

Name	Type	Valid range	Description
SrcAddrMode	Integer	0x00–0x03	The source addressing mode for this primitive and subsequent MPDU. This value can take one of the following values: 0x00 = no address (addressing fields omitted, see 7.2.1.1.8 of [IEEE 802.15.4]). 0x01 = reserved. 0x02 = 16-bit short address. 0x03 = 64-bit extended address.

Table A.D.1 – MCPS-DATA.request parameters

Name	Type	Valid range	Description
DstAddrMode	Integer	0x00–0x03	The destination addressing mode for this primitive and subsequent MPDU. This value can take one of the following values: 0x00 = no address (addressing fields omitted, see 7.2.1.1.6 of [IEEE 802.15.4]). 0x01 = reserved. 0x02 = 16-bit short address. 0x03 = 64-bit extended address.
DstPANId	Integer	0x0000–0xffff	The 16-bit PAN identifier of the entity to which the MSDU is being transferred. NOTE – PAN identifier value is logically AND with 0xFCFF
DstAddr	Device address	As specified by the DstAddrMode parameter	The individual device address of the entity to which the MSDU is being transferred.
msduLength	Integer	≤aMaxMACPayloadSize	The number of octets contained in the MSDU to be transmitted by the MAC sublayer entity.
Msdu	Set of octets	–	The set of octets forming the MSDU to be transmitted by the MAC sublayer entity.
msduHandle	Integer	0x00–0xff	The handle associated with the MSDU to be transmitted by the MAC sublayer entity.
TxOptions	Bitmap	3-bit field	The 3 bits (b0, b1, b2) indicate the transmission options for this MSDU. For b0: 1 = acknowledged transmission, 0 = unacknowledged transmission. For b1: 1 = GTS transmission, 0 = CAP transmission for a beacon-enabled PAN. For b2: 1 = indirect transmission, 0 = direct transmission. Indirect transmission is not supported and bit b2 should be always set to 0. For a non beacon-enabled PAN, bit b1 shall be set to 0.

Table A.D.1 – MCPS-DATA.request parameters

Name	Type	Valid range	Description
QualityOf Service	Integer	0x00–0x02	The QOS (Quality of Service) parameter of the MSDU to be transmitted by the MAC sublayer entity. This value can take one of the following values: 0 = Normal priority, 1 = High priority, 2 = Contention free (optional).
SecurityLevel	Integer	0x00 and 0x05	The security level to be used as described in A.3.3.6.
KeyIdMode	Integer	0x01	The mode used to identify the key to be used (see A.3.3.6).This parameter is ignored if the SecurityLevel parameter is set to 0x00.
KeySource	Set of 0 octets	As specified by	Not used
KeyIndex	Integer	0x00 – 0x01	The index of the key to be used (see A.3.3.6).

A.D.2 MCPS-DATA.indication

The semantic of the MCPS-DATA.indication primitive is as follows:

MCPS-DATA.indication (

SrcAddrMode,

SrcPANId,

SrcAddr,

DstAddrMode,

DstPANId,

DstAddr,

msduLength,

msdu,

msduLinkQuality,

DSN,

Timestamp,

SecurityLevel,

KeyIdMode,

KeySource,

KeyIndex,

QualityOfService

)

The table below specifies the parameters for the MCPS-DATA.indication primitive.

Table A.D.2 – MCPS-DATA.indication parameters

Name	Type	Valid range	Description
SrcAddrMode	Integer	0x00 – 0x03	The source addressing mode for this primitive and subsequent MPDU. This value can take one of the following values: 0x00 = no address (addressing fields omitted, see clause 7.2.1.1.8 of [IEEE 802.15.4]). 0x01 = reserved. 0x02 = 16-bit short address. 0x03 = 64-bit extended address.
SrcPANId	Integer	0x0000 – 0xFFFF	The 16-bit PAN identifier of the device from which the frame was received. NOTE – PAN identifier value is logically AND with 0xFCFF
SrcAddr	Device address	As specified by the SrcAddrMode parameter	The address of the device which sent the message.
DstAddrMode	Integer	0x00 – 0x03	The destination addressing mode for this primitive and subsequent MPDU. This value can take one of the following values: 0x00 = no address (addressing fields omitted, see clause 7.2.1.1.6 of [IEEE 802.15.4]). 0x01 = reserved. 0x02 = 16-bit short address. 0x03 = 64-bit extended address.
DstPANId	Integer	0x0000–0xffff	The 16-bit PAN identifier of the entity to which the MSDU is being transferred. NOTE – PAN identifier value is logically AND with 0xFCFF
DstAddr	Device address	As specified by the DstAddrMode parameter	The individual device address of the entity to which the MSDU is being transferred.
msduLength	Integer	≤aMaxMACPayloadSize	The number of octets contained in the MSDU to be indicated to the upper layer
msdu	Set of octets	–	The set of octets forming the MSDU received by the MAC sublayer entity.
msduLinkQuality	Integer	0x00 – 0xFF	The LQI value measured during reception of the message.
DSN	Integer	0x00 – 0xFF	The DSN of the received frame.
Timestamp	Integer	0x00000000 – 0xFFFFFFFF	The absolute time in milliseconds at which the frame was received and constructed, decrypted (assuming encryption was valid) (32 bit value).
SecurityLevel	Integer	0x00 and 0x05	The security level to be used as described in A.3.3.6.

Table A.D.2 – MCPS-DATA.indication parameters

Name	Type	Valid range	Description
KeyIdMode	Integer	0x01	The mode used to identify the key used (see Table 96 in clause 7.6.2.2.2 of [IEEE 802.15.4]). This parameter is ignored if the SecurityLevel parameter is set to 0x00.
KeySource	Set of 0 octets	As specified by the KeyIdMode parameter	Not used
KeyIndex	Integer	0x00 – 0x01	The index of the key to be used (see A.3.3.6).

QualityOfServ ice	Integer	0x00–0x02	The QOS (Quality of Service) parameter of the MSDU received by the MAC sublayer entity.
			This value can take one of the following values:
			0 = Normal priority;
			1 = High priority;
			2 = Contention free (optional);

G3 Annex E

(normative)

MAC acknowledgement

(This annex forms an integral part of this Recommendation.)

The present specification does not use IEEE 802.15.4-2006 MAC acknowledgment frame but specifies a positive and negative acknowledgments using Frame Control Header (clause A.5.5 of [ITU-T G.9955]).

The Frame Control Header contains an information used by all stations in the network for channel access, as well as PHY receiver information used by the destination. For this reason, Frame Control Header has specific physical layer encoding and modulation as defined in [ITU-T G.9955]/Annex A.

Only Frame Control Header will be used as positive (ACK) or negative (NACK) acknowledgement.

The packet originator may request an acknowledgment by setting Delimiter Type field of Frame Control Header (clause A.5.5 of [ITU-T G.9955]).

The receiver will send ACK to the originator if it is requested and the MAC frame was decoded correctly by PHY.

If the MAC frame is received without error as determined by FCS, the receiver shall send an ACK to the originator only if an acknowledgement is requested and the destination address and PANID of the frame match the receiver's device address and PANID.

If the MAC frame is received with errors as determined by FCS, the receiver may send a NACK to the originator only if an acknowledgment is requested and the destination address and PANID of the frame match the receiver's device address and PANID.

However, if the receiver can determine that the error is caused by collision, it may avoid sending a NACK (no response) to invoke a collision state on transmitting station. The transmitting station shall infer a collision from the absence of any response to a transmission when a response is expected. In this case the transmitting station shall attempt a retransmission after EIFS interval.

If a valid NACK is received the transmitting station shall attempt a retransmission using CSMA random backoff.

The receiver will send NACK to the originator if it is requested and the received MAC frame is corrupted and cannot be recovered by PHY.

ACK and NACK frames contain the 16-bit CRC (MAC FCS field) received in the MAC frame for which the ACK or NACK response is being sent. These 16 bits are used as ACK or NACK identifier and are located in FCH as follow TM[7:0] = FCS[15:8] and PDC[7:0] = FCS[7:0] (see clause A.5.2 of [ITU-T G.9955]). The transmitter shall extract FCS field from received ACK/NACK and compare it with FCS of transmitted packet to determine validity of the response. If it matches, the ACK/NACK response is accepted otherwise it will be ignored and treated as a collision.

G3 Annex F

(normative)

Adaptation sublayer service primitives

(This annex forms an integral part of this Recommendation.)

A.F.1 ADP data service

A.F.1.1 Overview

The ADPD is used to transport application layer PDU to other devices on the network, and supports the following primitives:

- ADPD-DATA.request;
- ADPD-DATA.confirm;
- ADPD-DATA.indication.

A.F.1.2 ADPD-DATA.request

A.F.1.2.1 Semantics of the service primitive

This primitive requests the transfer of an application PDU to another device, or multiple devices. The semantics of this primitive are as follows:

```
ADPD-DATA.request        (
                         NsduLength,
                         Nsdu,
                         NsduHandle,
                         DiscoverRoute,
                         QualityOfService,
                         SecurityEnabled
                         )
```

Table A.F.1 – Parameters of the ADPD-DATA.request primitive

Name	Type	Valid range	Description
NsduLength	Integer	0 – 1 280	The size of the NSDU, in bytes
Nsdu	Set of octets	–	The NSDU to send
NsduHandle	Integer	0x00 – 0xFF	The handle of the NSDU to transmit. This parameter is used to identify in the ADPD-DATA.confirm primitive which request is concerned. It can be randomly chosen by the application layer.
DiscoverRoute	Boolean	TRUE or FALSE	If TRUE, a route discovery procedure will be performed prior to sending the frame if a route to the destination is not available in the routing table.

Table A.F.1 – Parameters of the ADPD-DATA.request primitive

Name	Type	Valid range	Description
			If FALSE, no route discovery is performed.
QualityOfService	Integer	0x00 – 0x02	The requested quality of service (QoS) of the frame to send. Allowed values are: 0x00 = normal priority 0x01 = high priority 0x02 = contention free access (optional)
SecurityEnabled	Boolean	TRUE or FALSE	If TRUE, the frame shall be sent encrypted.

A.F.1.2.2 When generated

This primitive is generated by the upper layer to request the sending of a given NSDU.

A.F.1.2.3 Effect on receipt

If this primitive is received when the device has not joined a network, the adaptation sublayer will issue an ADPD-DATA.confirm primitive with the status INVALID_REQUEST. Else, the ADPD constructs a 6LoWPAN frame with the following characteristics depending on transmission mode:

Case of unicast frame:

- The mesh addressing header is present as described in clause 5.2 of [IETF RFC 4944], where:
  - V shall be set to 1, to specify that the originator address is a 16-bit network address.
  - F shall be set to 1, to specify that the originator address is a 16-bit network address.
  - HopsLft = MaxHops.
  - Originator address = The 16-bit network address of the sending device, available in the NIB.
  - Final destination address = 16-bit destination address of the device designated by the IPv6 address "DstAddr".
  - The broadcast header is not present.
- If necessary, the fragmentation header shall be present to transport NPDU which do not fit in an entire IEEE 802.15.4 frame. In that case, clause 5.3 of [IETF RFC 4944] applies,
  - LOWPAN_HC1 compressed IPv6 header is present with the following parameters:
    - IPv6 Source Address mode = PC-IC (bits 0 and 1 set to 1).
    - IPv6 Destination Address mode = PC-IC (bits 2 and 3 set to 1).
    - Bit 4 = 1 (no Traffic Class and Flow Label).
    - Bits 5 and 6 = value of NsduType.

Case of multicast frame:

- The mesh addressing header is present as described in part 5.2 of [IETF RFC 4944], where
  - V shall be set to 1, to specify that the originator address is a 16-bit network address.
  - F shall be set to 1, to specify that the originator address is a 16-bit network address.
  - HopsLft = MaxHops.
  - Originator address = The 16-bit network address of the sending device, available in the NIB.

  - Final destination address = 0xFFFF.
  - The broadcast header is present with the following values:
    - Sequence Number = previous Sequence Number + 1
- If necessary, the fragmentation header shall be present to transport NPDU which do not fit in an entire IEEE 802.15.4 frame. In that case, clause 5.3 of [IETF RFC 4944] applies,
  - LOWPAN_HC1 compressed IPv6 header is present with the following parameters:
    - IPv6 Source Address mode = PC-IC (bits 0 and 1 set to 1).
    - IPv6 Destination Address mode = PC-IC (bits 2 and 3 set to 1).
    - Bit 4 = 1 (no Traffic Class and Flow Label).
    - Bits 5 and 6 = value of NsduType.

Once the frame is constructed, it is routed according to the procedures described in A.3.4.4 (modified clause 6 of G3 Annex H) if the destination address is a unicast address. If the frame is to be transmitted, the MCPS-Data.request primitive is invoked, with the following parameters in case of a unicast sending:

- SrcAddrMode = 0x02, for 16-bit address;
- DstAddrMode = 0x02, for 16-bit address;
- SrcPANId = DstPANId = the value of macPANId obtained from the MAC PIB;
- SrcAddr = the value of macShortAddr obtained from the MAC PIB;
- DstAddr = the 16-bit address of the next hop determined by the routing procedure;
- msduLength = the length of the frame, or fragment in case of fragmentation, in bytes;
- msdu = the frame itself;
- msduHandle = NsduHandle.
- TxOptions:
  - b0 = 1 if unicast transmission, 0 otherwise;
  - b1 = 0;
  - b2 = 0.
- SecurityLevel:
  - 0 if SecurityEnabled = FALSE;
  - 5 if SecurityEnabled = TRUE.
- KeyIdMode, KeySource: Ignored.
- KeyIndex: Ignored if SecurityLevel=0; Else depends on security policy.

In case of a broadcast (or multicast) frame, the MCPS-Data.request primitive is invoked with the following parameters:

- SrcAddrMode = 0x02, for 16-bit address;
- DstAddrMode = 0x02, for 16-bit address;
- SrcPANId = DstPANId = the value of macPANId obtained from the MAC PIB;
- SrcAddr = the value of macShortAddr obtained from the MAC PIB;
- DstAddr = 0xFFFF;
- msduLength = the length of the frame, or fragment in case of fragmentation, in bytes;
- msdu = the frame itself;
- msduHandle = NsduHandle;
- TxOptions:

  - b0 = 1 if unicast transmission, 0 otherwise;
  - b1 = 0;
  - b2 = 0.
- SecurityLevel
  - 0 if SecurityEnabled = FALSE;
  - 5 if SecurityEnabled = TRUE.
- KeyIdMode, KeySource: Ignored.
- KeyIndex: Ignored if SecurityLevel=0; Else depends on security policy.

If security processing fails for that frame, it shall be discarded and an ADPD-DATA.confirm primitive shall be generated with the status code returned by the security processing suite.

If the DiscoverRoute parameter is set to TRUE then, the route discovery procedure shall be initiated prior to sending the frame in case the final destination address is not available in the routing table. For a complete description of this procedure, see A.3.4.4.

A.F.1.3 ADPD-DATA.confirm

A.F.1.3.1 Semantics of the service primitive

This primitive reports the result of a previous ADPD-DATA.request primitive.

The semantics of this primitive are as follows:

```
ADPD-DATA.confirm          (
                           Status,
                           NsduHandle
                           )
```

Table A.F.2 – Parameters of the ADPD-DATA.confirm primitive

Name	Type	Valid Range	Description
Status	Enum	SUCCESS, INVALID_IPV6_FRAME, INVALID_REQUEST, NO_KEY, BAD_CCM_OUTPUT, ROUTE_ERROR, BT_TABLE_FULL, FRAME_NOT_BUFFERED or any status values returned from security suite or the MCPS-DATA.confirm primitive	The status code of a previous ADPD-DATA.request identified by its NsduHandle
NsduHandle	Integer	0x00 – 0xFF	The handle of the NSDU confirmed by this primitive

A.F.1.3.2 When generated

This primitive is generated in response to an ADPD-DATA.request primitive. The Status parameter indicates if the request succeeded or the reason of failure.

A.F.1.3.3 Effect on receipt

On reception of this primitive, the upper layer is notified of the status of a previous ADPD-DATA.request primitive.

A.F.1.4 ADPD-DATA.indication

A.F.1.4.1 Semantics of the service primitive

This primitive is used to transfer received data from the adaptation sublayer to the upper layer. The semantics of this primitive are as follows:

```
ADPD-DATA.indication      (
                          NsduLength,
                          Nsdu,
                          LinkQualityIndicator,
                          SecurityEnabled
                          )
```

Table A.F.3 – Parameters of the ADPD-DATA.indication primitive

Name	Type	Valid Range	Description
NsduLength	Integer	0-1280	The size of the NSDU, in bytes
Nsdu	Set of octets	–	The received NSDU
LinkQualityIndicator	Integer	0x00-0xFF	The value of the link quality during reception of the frame
SecurityEnabled	Boolean	TRUE or FALSE	TRUE if the received frame was encrypted.

A.F.1.4.2 When generated

This primitive is generated by the adaptation sublayer when a valid data frame whose final destination is the current station has been received.

A.F.1.4.3 Effect on receipt

On generation of this primitive, the upper layer is notified of the arrival of a data frame.

A.F.2 ADP management service

A.F.2.1 Overview

The ADPM allows the transport of command frames used for network maintenance. The list of primitives supported by the ADPM is:

- ADPM-DISCOVERY.request;
- ADPM-DISCOVERY.confirm;
- ADPM-NETWORK-START.request;
- ADPM-NETWORK-START.confirm;
- ADPM-NETWORK-JOIN.request;
- ADPM-NETWORK-JOIN.confirm;
- ADPM-NETWORK-JOIN.indication;
- ADPM-NETWORK-LEAVE.request;

- ADPM-NETWORK-LEAVE.indication;
- ADPM-NETWORK-LEAVE.confirm;
- ADPM-RESET.request;
- ADPM-RESET.confirm;
- ADPM-GET.request;
- ADPM-GET.confirm;
- ADPM-SET.request;
- ADPM-SET.confirm;
- ADPM-NETWORK-STATUS.indication;
- ADPM-ROUTE-DISCOVERY.request;
- ADPM-ROUTE-DISCOVERY.confirm;
- ADPM-PATH-DISCOVERY.request;
- ADPM-PATH-DISCOVERY.confirm.

A.F.2.2 ADPM-DISCOVERY.request

A.F.2.2.1 Semantics of the service primitive

This primitive allows the upper layer to request the ADPM to scan for networks operating in its POS.

The semantics of this primitive are as follows:

```
ADPM-DISCOVERY.request          (
                                Duration,
                                )
```

Table A.F.4 – Parameters of the ADPM-DISCOVERY.request primitive

Name	Type	Valid Range	Description
Duration	Integer	0x00-0xFF	The number of seconds the active scan shall last

A.F.2.2.2 When generated

This primitive is generated by the next upper layer to get informed of the current networks operating in the POS of the device.

A.F.2.2.3 Effect on receipt

On receipt of this primitive, the ADP layer will initiate an active scan by invoking the MLME-SCAN.request with the following parameters:

- ScanType = 0x01 for active scan;
- ScanChannels = all bits set to 0 (not used);
- ScanDuration = Duration;
- ChannelPage = 0 (not used);
- SecurityLevel = 0;
- KeyIdMode, KeySource and KeyIndex: Ignored.

Upon receiving each Beacon frame, the MAC layer in LBD issues a MLME-BEACON-NOTIFY.indication primitive with the PANDescriptor parameters corresponding to the beacon. At the end of scan duration, the adaptation layer generates an ADPM-DISCOVERY.confirm primitive which contains the PANDescriptorList according to the procedure described in 7.4.5.2.2.2.

A.F.2.3 ADPM-DISCOVERY.confirm

A.F.2.3.1 Semantics of the service primitive

This primitive is generated by the ADP layer upon completion of a previous ADPM-DISCOVERY.request.

The semantics of this primitive are as follows:

```
ADPM-DISCOVERY.confirm          (
                                Status,
                                PANCount,
                                PANDescriptor
                                )
```

Table A.F.5 – Parameters of the ADPM-DISCOVERY.confirm primitive

Name	Type	Valid Range	Description
Status	Enum	FAILED, SUCCESS, NO_BEACON	SUCCESS if at least one MLME-BEACON-NOTIFY.indication is received, NO_BEACON if no MLME-BEACON-NOTIFY.indication is received In all other cases, FAILED.
PANCount	Integer	0x00-0xFF	The number of networks operating in the POS of the device
PANDescriptor	List of PAN descriptors	This list contains the PAN descriptors as described in Table A.F.6. Number of PAN descriptors is specified by PANCount.	The PAN operating in the POS of the device

Table A.F.6 – PAN descriptor structure specification

Name	Type	Valid Range	Description
PANId	Integer	0x0000 – 0xFFFF. PAN identifier must be logically ANDed with 0xFCFF.	The 16-bit PAN identifier.
LinkQuality	Integer	0x00-0xFF	The 8-bit Link Quality of LBA. It is used by the associating device to select LBA and PAN.
LBAAddress	Integer	0x0000-0xFFFF	The 16 bit short address of a device in this PAN to be used as the LBA by the associating device.

Table A.F.6 – PAN descriptor structure specification

Name	Type	Valid Range	Description
RC_COORD	Integer	0x00-0xFF	The estimated route cost from LBA to the coordinator. It is used by the associating device to select LBA and PAN.

A.F.2.3.2 When generated

This primitive is generated by the ADP layer for the upper layer on completion of an ADPM-DISCOVERY.request primitive.

A.F.2.3.3 Effect on receipt

On receipt of this primitive, the upper layer is notified of the completion of the network scan, and obtains a list of found operating networks.

A.F.2.4 ADPM-NETWORK-START.request

A.F.2.4.1 Semantics of the service primitive

This primitive allows the upper layer to request the starting of a new network. It shall only be invoked by device designated as the PAN coordinator during the factory process.

The semantics of this primitive are as follows:

```
ADPM-NETWORK-START.request          (
                                    PANId
                                    )
```

Table A.F.7 – Parameters of the ADPM-NETWORK-START.request primitive

Name	Type	Valid Range	Description
PANId	Integer	0x0000 – 0xFFFF	The PANId of the network to create, determined at the application level NOTE – PANId value must be logically AND with 0xFCFF.

A.F.2.4.2 When generated

This primitive is generated by the upper layer of the PAN coordinator to start a new network.

A.F.2.4.3 Effect on receipt

On receipt of this primitive by a device which is not a PAN coordinator, it shall issue an ADPM-NETWORK-START.confirm primitive with the status INVALID_REQUEST.

Prior to invoking this primitive, the upper layer of the PAN coordinator shall perform an ADPM-DISCOVERY.request to make sure no other network is currently operating. In case another network is operating, the upper layer may invoke the ADPM-NETWORK-START.request.

On receipt of this primitive by a device which is the PAN coordinator, and if no network has already be formed, the ADP layer shall perform the steps described in A.3.5.1.

On receipt of the MLME-START.confirm primitive, the ADP layer shall issue an ADPM-NETWORK-START.confirm primitive with the appropriate status code.

A.F.2.5 ADPM-NETWORK-START.confirm

A.F.2.5.1 Semantics of the service primitive

This primitive reports the status of a ADPM-NETWORK-START.request.

The semantics of this primitive are as follows:

ADPM-NETWORK-START.confirm (
Status
)

Table A.F.8 – Parameters of the ADPM-NETWORK-START.confirm primitive

Name	Type	Valid Range	Description
Status	Enum	SUCCESS, INVALID_REQUEST, STARTUP_FAILURE or any status value returned from the MLME-START.confirm primitive	The result of the attempt to create the network

A.F.2.5.2 When generated

This primitive is generated by the ADP layer in response to an ADPM-NETWORK-START.request primitive, and indicates if the network formation was successful or not, and an eventual reason for failure.

A.F.2.5.3 Effect on receipt

On receipt of this primitive, the next higher layer is notified about the status of its previous ADPM-NETWORK-START.request.

A.F.2.6 ADPM-NETWORK-JOIN.request

A.F.2.6.1 Semantics of the service primitive

This primitive allows the next upper layer to join an existing network.

The semantics of this primitive are as follows:

ADPM-NETWORK-JOIN.request (
PANId,
LBAAddress
)

Table A.F.9 – Parameters of the ADPM-NETWORK-JOIN.request primitive

Name	Type	Valid Range	Description
PANId	Integer	0x0000 – 0xFFFF	The 16-bit PAN identifier of the network to join.
LBAAddress	16-bit address	0x0000 – 0xFFFF	The 16-bit short address of the device acting as a LoWPAN Bootstrap Agent as defined in G3 Annex I.

A.F.2.6.2 When generated

The upper layer invokes this primitive when it wishes to join an existing PAN using the MAC association procedure.

A.F.2.6.3 Effect on receipt

On receipt of this primitive by a device which is already joined, the adaptation sublayer generates an ADPM-NETWORK-JOIN.confirm with the status INVALID_REQUEST.

On receipt of this primitive by a device which is not already joined, the adaptation sublayer initiates the MAC association procedure ("bootstrap") described in A.3.4.5.2.2.

On completion, an MLME-SET.request is invoked to set the 16-bit short address of the device which was obtained during the "bootstrapping" phase. Then, an ADPM-NETWORK-JOIN.confirm primitive is generated with a status of SUCCESS.

A.F.2.7 ADPM-NETWORK-JOIN.confirm

A.F.2.7.1 Semantics of the service primitive

This primitive is generated by the ADP layer to indicate the completion status of a previous ADPM-NETWORK-JOIN.request.

The semantics of this primitive are as follows:

```
ADPM-NETWORK-JOIN.confirm   (
                            Status,
                            NetworkAddress,
                            PANId
                            )
```

Table A.F.10 – Parameters of the ADPM-NETWORK-JOIN.confirm primitive

Name	Type	Valid Range	Description
Status	Status	SUCCESS, INVALID_REQUEST, NOT_PERMITTED.	The result of the attempt to join the network.
NetworkAddress	Integer	0x0001 –0x7FFF, 0xFFFF	The 16-bit network address that was allocated to the device. If the allocation fails, this address is equal to 0xFFFF.
PANId	Integer	0x0000 – 0xFFFF	The 16-bit address of the PAN of which the device is now a member. NOTE – PANId value is logically AND with 0xFCFF.

A.F.2.7.2 When generated

This primitive is generated in response to an ADPM-NETWORK-JOIN.request primitive, and allows the upper layer to obtain information on the status of its request.

The status NOT_PERMITTED is given if the device was unable to authenticate itself to the PAN coordinator.

A.F.2.7.3 Effect on receipt

On receipt of this primitive the upper layer is informed on the status of its request.

A.F.2.8 ADPM-NETWORK-LEAVE.request

This primitive allows a non coordinator device to remove itself from the network as described in A.3.4.5.2.2.8. The removal of a device by the coordinator is performed using ADPM-LBP.request according to the procedure described in A.3.4.5.2.2.7.

A.F.2.8.1 Semantics of the service primitive

The semantics of this primitive are as follows:

```
ADPM-NETWORK-LEAVE.request          (
                                    ExtendedAddress
                                    )
```

Table A.F.11 – Parameters of the ADPM-NETWORK-LEAVE.request primitive

Name	Type	Valid Range	Description
ExtendedAddress	64-bit address	Any	If NULL, the device removes itself from the network.

A.F.2.8.2 When generated

The next higher layer of a non coordinator device generates this primitive to request to leave the network.

A.F.2.8.3 Effect on receipt

On receipt of this primitive by a device which is not associated to any network, the adaptation sublayer shall issue an ADPM-NETWORK-LEAVE.confirm primitive with the status INVALID_REQUEST.

On receipt of this primitive by a device which is associated to any network, the following steps shall be performed:

– If the device is a coordinator or if ExtendedAddress != NULL,
  - Issue ADPM-NETWORK-LEAVE.confirm with INVALID_REQUEST

– Else
  - The device removes itself from the network, using the procedure described in A.3.4.5.2.2.8.
  - Issue ADPM-NETWORK-LEAVE.confirm with SUCCESS

A.F.2.9 ADPM-NETWORK-LEAVE.indication

A.F.2.9.1 Semantics of the service primitive

This primitive is generated by the ADP layer of a non coordinator device to inform the upper layer that it has been unregistered from the network by the coordinator. The semantics of this primitive are as follows:

```
ADPM-NETWORK-LEAVE.indication  (
                                    ExtendedAddress,
                                    )
```

Table A.F.12 – Parameters of the ADPM-NETWORK-LEAVE.indication primitive

Name	Type	Valid Range	Description
ExtendedAddress	64-bit address	Any	The 64-bit network address of the device removed from the network.

A.F.2.9.2 When generated

This primitive is generated by the adaptation sublayer of a device when it has been removed from the network by the PAN coordinator or by the adaptation sublayer of the PAN coordinator when a device has decided to leave the network.

A.F.2.9.3 Effect on receipt

On receipt of this primitive, the upper layer of the device is notified that it is no more a part of the PAN.

A.F.2.10 ADPM-NETWORK-LEAVE.confirm

A.F.2.10.1 Semantics of the service primitive

This primitive allows the upper layer to be informed on the status of its previous ADPM-NETWORK-LEAVE.request.

The semantics of this primitive are as follows:

```
ADPM-NETWORK-LEAVE.confirm      (
                                Status,
                                ExtendedAddress
                                )
```

Table A.F.13 – Parameters of the ADPM-NETWORK-LEAVE.confirm primitive

Name	Type	Valid Range	Description
Status	Enum	SUCCESS, INVALID_REQUEST, UNKNOWN_DEVICE or any status returned by the MCPS-DATA.confirm primitive	The status of the request.
ExtendedAddress	64-bit address	Any	The 64-bit network address of the device removed from the network.

A.F.2.10.2 When generated

This primitive is generated on completion of a device removal. If it is successful, the SUCCESS code is given. Else, an error status is given as explained in A.3.4.5.2.2.8.

A.F.2.10.3 Effect on receipt

On receipt, the upper layer is notified of the result of its request.

A.F.2.11 ADPM-RESET.request

A.F.2.11.1 Semantics of the service primitive

This primitive allows the upper layer to request that the ADP layer performs a reset.

The semantics of this primitive are as follows:

```
ADPM-RESET.request           (
                            )
```

This primitive has no parameter.

A.F.2.11.2 When generated

This primitive allows a reset of the adaptation sublayer, and allows resetting the MIB attributes.

A.F.2.11.3 Effect on receipt

On receipt of this primitive, the following steps are performed:

– The adaptation sublayer issues a MLME-RESET.request primitive with the SetDefaultPIB parameter set to TRUE, and waits for the MLME-RESET.confirm primitive.

– The adaptation sublayer clears all of its internal variables, and flushes its routing and neighbour tables.

– The adaptation sublayer issues an ADPM-RESET.confirm primitive with the status SUCCESS, or DISABLE_TRX_FAILURE if the MAC reset operation failed.

A.F.2.12 ADPM-RESET.confirm

A.F.2.12.1 Semantics of the service primitive

This primitive allows the upper layer to be notified of the completion of an ADPM-RESET.request primitive.

The semantics of this primitive are as follows:

```
ADPM-RESET.confirm        (
                          Status
                          )
```

Table A.F.14 – Parameters of the ADPM-RESET.confirm primitive

Name	Type	Valid Range	Description
Status	Enum	Any status value returned from the MLMERESET.confirm primitive	The status of the request

A.F.2.12.2 When generated

This primitive is generated by the ADP layer when a previous ADPM-RESET.request primitive has completed.

A.F.2.12.3 Effect on receipt

The upper layer is notified of the completion of the command.

A.F.2.13 ADPM-GET.request

A.F.2.13.1 Semantics of the service primitive

This primitive allows the upper layer to get the value of an attribute from the information base.

The semantics of this primitive are as follows:

```
ADPM-GET.request (
                          AttributeId,
```

AttributeIndex

)

Table A.F.15 – Parameters of the ADPM-GET.request primitive

Name	Type	Valid Range	Description
AttributeId	Integer	See clause A.3.4.2	The identifier of the IB attribute to read
AttributeIndex	Integer	Depends on attribute, see A.3.4.2	The index within the table of the specified IB attribute to read. This parameter is valid only for IB attributes that are tables.

A.F.2.13.2 When generated

This primitive is generated by the upper layer to read the value of an attribute from the IB.

A.F.2.13.3 Effect on receipt

On receipt of this primitive, the adaptation sublayer attempts to retrieve the selected attribute in the information base. If the attribute is not found, the adaptation layer generates an ADPM-GET.confirm primitive with the status UNSUPPORTED_ATTRIBUTE. If the attribute is found (and is a table), but the AttributeIndex is out of range, the adaptation layer generates an ADPM-GET.confirm primitive with the status INVALID_INDEX.

Else, the adaptation sublayer generates an ADPM-GET.confirm primitive with the status SUCCESS, and the value read from the IB in the AttributeValue parameter.

A.F.2.14 ADPM-GET.confirm

A.F.2.14.1 Semantics of the service primitive

This primitive allows the upper layer to be informed of the status of a previously issued ADPM-GET.request primitive.

The semantics of this primitive are as follows:

ADPM-GET.confirm (

Status,

AttributeId,

AttributeIndex,

AttributeValue

)

Table A.F.16 – Parameters of the ADPM-GET.confirm primitive

Name	Type	Valid Range	Description
Status	Enum	SUCCESS, UNSUPPORTED_ATTRIBUTE or INVALID_INDEX	The status of the reading.
AttributeId	Integer	See A.3.4.2	The identifier of the IB attribute read.
AttributeIndex	Integer	Depends on attribute, see A.3.4.2	The index within the table of the specified IB attribute read. This parameter is valid only for IB attributes that are tables.
AttributeValue	Various	Attribute specific	The value of the attribute read from the IB.

A.F.2.14.2 When generated

This primitive is generated by the adaptation sublayer in response to an ADPM-GET.request primitive.

A.F.2.14.3 Effect on receipt

On reception of this primitive, the upper layer is informed on the status of its request, and eventually gets the desired value.

A.F.2.15 ADPM-SET.request

A.F.2.15.1 Semantics of the service primitive

This primitive allows the upper layer to set the value of an attribute in the information base.

The semantics of this primitive are as follows:

```
ADPM-SET.request (
                  AttributeId,
                  AttributeIndex,
                  AttributeValue
                  )
```

Table A.F.17 – Parameters of the ADPM-SET.request primitive

Name	Type	Valid Range	Description
AttributeId	Integer	See A.3.4.2	The identifier of the IB attribute to write
AttributeIndex	Integer	Depends on attribute, see A.3.4.2	The index within the table of the specified IB attribute to write. This parameter is valid only for IB attributes that are tables.
AttributeValue	Various	Depends on attribute	The value to write

A.F.2.15.2 When generated

This primitive is generated by the upper layer to write the value of an attribute in the IB.

A.F.2.15.3 Effect on receipt

On receipt of this primitive, the adaptation sublayer attempts to write the selected attribute in the information base. If the attribute is not found, the adaptation layer generates an ADPM-SET.confirm primitive with the status UNSUPPORTED_ATTRIBUTE. If the attribute is found (and is a table), but the AttributeIndex is out of range, the adaptation layer generates an ADPM-SET.confirm primitive with the status INVALID_INDEX. If the attribute is found but is read only, the adaptation layer generates an ADPM-SET.confirm primitive with the status READ_ONLY. If the attribute is found, can be written, but the AttributeValue is out of range, the adaptation layer generates an ADPM-SET.confirm primitive with the status INVALID_PARAMETER. Else, the adaptation layer generates an ADPM-SET.confirm primitive with the status SUCCESS.

A.F.2.16 ADPM-SET.confirm

A.F.2.16.1 Semantics of the service primitive

This primitive allows the upper layer to be informed about a previous ADPM-SET.request primitive.

The semantics of this primitive are as follows:

```
ADPM-SET.confirm  (
                    Status,
                    AttributeId,
                    AttributeIndex
                    )
```

Table A.F.18 – Parameters of the ADPM-SET.confirm primitive

Name	Type	Valid Range	Description
Status	Enum	SUCCESS, UNSUPPORTED_ATTRIBUTE, READ_ONLY, INVALID_PARAMETER or INVALID_INDEX	The status of the writing
AttributeId	Integer	See A.3.4.2	The identifier of the IB attribute written
AttributeIndex	Integer	Depends on attribute, see A.3.4.2	The index within the table of the specified IB attribute written. This parameter is valid only for IB attributes that are tables.

A.F.2.16.2 When generated

This primitive is generated by the adaptation layer in response to an ADPM-SET.request primitive.

A.F.2.16.3 Effect on receipt

On reception of this primitive, the upper layer is informed on the status of its request.

A.F.2.17 ADPM-NETWORK-STATUS.indication

A.F.2.17.1 Semantics of the service primitive

This primitive allows the next higher layer of a PAN coordinator or a coordinator to be notified when a particular event occurs on the PAN.

The semantics of this primitive are as follows:

```
ADPM-NETWORK-STATUS.indication    (
                                  Status,
                                  AdditionalInformation
                                  )
```

Table A.F.19 – Parameters of the ADPM-NETWORK-STATUS.indication primitive

Name	Type	Valid Range	Description
Status	Enum	PAN_ID_CONFLICT, or any status code returned by MLME-COMM-STATUS.indication	The status or event to notify.
AdditionalInformation	String	Any string	The eventual additional information to the status or event.

A.F.2.17.2 When generated

This primitive is generated when the adaptation sublayer of a PAN coordinator has received a LBP message from a device on the network indicating that a PAN Id conflict is occurring. See A.3.5.2 for complete description of the PAN ID conflict handling mechanism.

In that case, this primitive is never generated by the adaptation sublayer of a device which is not a PAN coordinator.

This primitive is also generated if the underlying MAC layer (of a PAN coordinator or a coordinator) generates a MLME-COMM-STATUS.indication.

A.F.2.17.3 Effect on receipt

On reception, the upper layer of a PAN coordinator is informed that a PAN Id conflict was detected, or that a MAC event occurred.

A.F.2.18 ADPM-ROUTE-DISCOVERY.request

A.F.2.18.1 Semantics of the service primitive

This primitive allows the upper layer to initiate a route discovery.

The semantics of this primitive are as follows:

```
ADPM-ROUTE-DISCOVERY.request      (
                                  DstAddr,
                                  MaxHops
                                  )
```

Table A.F.20 – Parameters of the ADPM-ROUTE-DISCOVERY.request primitive

Name	Type	Valid Range	Description
DstAddr	Short address	0x00-0x7FFF	The Short unicast destination address of the route discovery.
MaxHops	Integer	0x01-0x0E	This parameter indicates the maximum number of hops allowed for the route discovery.

A.F.2.18.2 When generated

This primitive is generated by the upper layer of a device to obtain a route to another device.

A.F.2.18.3 Effect on receipt

An ADPM-ROUTE-DISCOVERY.confirm with the status INVALID_REQUEST is generated if the DstAddr is not a unicast IPv6 address, or if the MaxHops value is out of range.

On receipt of this primitive, the device will initiate a route discovery procedure as described in A.3.4.4.2.3.

A.F.2.19 ADPM-ROUTE-DISCOVERY.confirm

A.F.2.19.1 Semantics of the service primitive

This primitive allows the upper layer to be informed of the completion of a route discovery.

The semantics of this primitive are as follows:

```
ADPM-ROUTE-DISCOVERY.confirm        (
                                    Status
                                    )
```

Table A.F.21 – Parameters of the ADPM-ROUTE-DISCOVERY.confirm primitive

Name	Type	Valid Range	Description
Status	Status	SUCCESS, INVALID_REQUEST, ROUTE_ERROR	The status of the route discovery.

A.F.2.19.2 When generated

This primitive is generated by the adaptation layer on completion of a route discovery as described in A.3.4.4.2.3, and in G3 Annex H.

A.F.2.19.3 Effect on receipt

On reception of this primitive, the upper layer is informed on the completion of the route discovery. If the Status value is SUCCESS, the routing table has been correctly updated with a brand new route to the desired destination, and the device may begin sending frames to that destination.

A.F.2.20 ADPM-PATH-DISCOVERY.request

A.F.2.20.1 Semantics of the service primitive

This primitive allows the upper layer to initiate a path discovery.

The semantics of this primitive are as follows:

```
ADPM-PATH-DISCOVERY.request   (
                                    DstAddr
                                    )
```

Table A.F.22 – Parameters of the ADPM-PATH-DISCOVERY.request primitive

Name	Type	Valid Range	Description
DstAddr	short address	0 – 0x7FFF	The short unicast destination address of the path discovery.

A.F.2.20.2 When generated

This primitive is generated by the upper layer of a device to obtain the path to another device.

A.F.2.20.3 Effect on receipt

An ADPM-PATH-DISCOVERY.confirm with the status INVALID_REQUEST is generated if the DstAddr is not in the routing table or after the failure of the procedure.

On receipt of this primitive, the device will initiate a path discovery procedure as described in A.3.4.4.2.4.

A.F.2.21 ADPM-PATH-DISCOVERY.confirm

A.F.2.21.1 Semantics of the service primitive

This primitive allows the upper layer to be informed of the completion of a path discovery.

The semantics of this primitive are as follows:

```
ADPM-ROUTE-DISCOVERY.confirm        (
                                    DstAddr,
                                    NSDU
                                    )
```

Table A.F.23 – Parameters of the ADPM-PATH-DISCOVERY.confirm primitive

Name	Type	Valid Range	Description
DstAddr	Short address	0 – 0x7FFF	The Short unicast destination address of the path discovery.
NsduId	integer	N.C	The buffer containing addresses of nodes constituting the path.

A.F.2.21.2 When generated

This primitive is generated by the adaptation layer on completion of a path discovery as described in clause A.3.4.4.2.4 of the present document.

A.F.2.21.3 Effect on receipt

On reception of this primitive, the upper layer is informed on the completion of the path discovery.

A.F.2.22 ADPM-LBP.request

A.F.2.22.1 Semantics of the service primitive

This primitive allows the upper layer of client to send the LBP message to server modem.

The semantics of this primitive are as follows:

```
ADPM-LBP.request(
                DstAddrType,
                DstAddr,
                NsduLength,
                Nsdu,
                NsduHandle,
                NsduType,
```

MaxHops,
DiscoveryRoute,
QualityOfService,
SecurityEnable
)

Table A.F.24 – Parameters of the ADPM-LBP.request primitive

Name	Type	Valid Range	Description
DstAddrType	Integer	0x02 – 0x03	The type of destination address contained in the DstAddr parameter. The allowed values are: 0x02 = 2 Bytes address (LBA address) 0x03 = 8 Bytes address (LBD address
DstAddr	Set of octets	–	16 bits address of LBA or 64 bits (extended address of LBD)
NsduLength	Integer	0 – 1 280	The size of the NSDU, in bytes
Nsdu	Set of octets	–	The NSDU to send
NsduHandle	Integer	0x00 – 0xFF	The handle of the NSDU to transmit. This parameter is used to identify in the ADPM-LBP.confirm primitive which request is concerned. It can be randomly chosen by the application layer.
NsduType	Integer	0x00 – 0x03	The type of data contained in the NSDU. 0x00 = any data 0x01 = UDP 0x02 = ICMP 0x03 = TCP
MaxHops	Integer	0x01 – 0x0E	The number of times the frame will be repeated by network routers.
DiscoveryRoute	Boolean	TRUE – FALSE	If TRUE, a route discovery procedure will be performed prior to sending the frame if a route to the destination is not available in the routing table. If FALSE, no route discovery is performed.
QualityOfService	Integer	0x00 – 0x01	The requested quality of service (QoS) of the frame to send. Allowed values are: 0x00 = standard priority 0x01 = high priority
SecurityEnabled	Boolean	TRUE – FALSE	If TRUE, this parameter enables the ADP layer security for processing the frame.

A.F.2.22.2 When generated

This primitive is generated by the LBS to perform the authentication, re-keying and leave procedure.

A.F.2.22.3 Effect on receipt

On reception of this primitive, the modem sends the coming frame to the destination.

A.F.2.23 ADPM-LBP.confirm

A.F.2.23.1 Semantics of the service primitive

This primitive reports the result of a previous ADPM-LBP.request primitive.

The semantics of this primitive are as follows:

```
ADPM-LBP.confirm  (
                   Status,
                               NsduHandle,
                   )
```

Table A.F.25 – Parameters of the ADPM-LBP.confirm primitive

Name	Type	Valid Range	Description
Status	Enum	SUCCESS, INVALID_REQUEST, NO_KEY, BAD_CCM_OUTPUT, ROUTE_ERROR, BT_TABLE_FULL, FRAME_NOT_BUFFERED or any status values returned from security suite or the MCPS-DATA.confirm primitive	The status code of a previous ADPM-LBP.request identified by its NsduHandle.
NsduHandel	Integer	0x00 – 0xFF	The handle of the NSDU confirmed by this primitive.

A.F.2.23.2 When generated

This primitive is generated in response to a ADPM-LBP.request primitive, the Status parameter indicating if the request succeeded, or the reason of failure.

A.F.2.23.3 Effect on receipt

On reception of this primitive, the upper layer is notified of the status of a previous ADPM-LBP.request primitive.

A.F.2.24 ADPM-LBP.indication

A.F.2.24.1 Semantics of the service primitive

This primitive is used to transfer received LBP frame from the ADP layer to the upper layer.

The semantics of this primitive are as follows:

```
     ADPM-LBP.indication      (DstAddr,
                          SrcAddr,
                          NsduLength,
                          Nsdu,
                          NsduType,
                          LinkQualityIndicator,
                          SecurityEnabled
```

)

Table A.F.26 – Parameters of the ADPM-LBP.indication primitive

Name	Type	Valid Range	Description
DstAddr	Integer	0x0000 – 0xFFFF	16 bits final destination address
SrcAddr	Integer	0x0000 – 0xFFFF	16 bits original source address
NsduLength	Integer	0 – 1 280	The size of the NSDU, in bytes
Nsdu	Set of octets	–	The NSDU to send
NsduType	Integer	0x00 – 0x03	The type of data contained in the NSDU. 0x00 = any data 0x01 = UDP 0x02 = ICMP 0x03 = TCP
LinkQualityIndicator	Integer	0x00 – 0xFF	The value of the link quality during reception of the frame
SecurityEnabled	Boolean	TRUE – FALSE	If TRUE, this parameter enables the adaptation sublayer security for processing the frame.

A.F.2.24.2 When generated

This primitive is generated by the ADP layer of client modem when a valid LBP frame whose final destination is the current station has been received.

A.F.2.24.3 Effect on receipt

On generation of this primitive, the upper layer is notified of the arrival of a LBP frame.

A.F.2.25 ADPM-BUFFER.indication

A.F.2.25.1 Semantics of the service primitive

This primitive allows the next higher layer to be notified when the modem reach his limit capability to perform coming frame.

The semantics of this primitive are as follows:

```
ADPM-BUFFER.indication  (
                        BufferReady
                        )
```

Table A.F.27 – Parameters of the ADPM-BUFFER.indication primitive

Name	Type	Valid Range	Description
BufferReady	Boolean	TRUE – FALSE	TRUE: modem is ready to receipt more data frame FALSE: modem is not ready, stop sending data frame

A.F.2.25.2 When generated

This primitive is generated when the adaptation layer of a modem has reached his limit to perform more Data frame.

A.F.2.25.3 Effect on receipt

On reception, the upper layer shall stop the data flow if BufferReady is equal to FALSE and open it if BufferReady is TRUE.

A.F.3 Behaviour to MAC indications

A.F.3.1 Overview

This clause describes the behaviour of the adaptation layer in response to an unsolicited indication from the MAC layer.

A.F.3.2 MCPS-DATA.indication

On reception of this indication, the adaptation layer shall execute the routing algorithm as described in A.3.4.4.

A.F.3.3 MLME-ASSOCIATE.indication

Nothing shall be done upon reception of this primitive by the adaptation layer.

A.F.3.4 MLME-DISASSOCIATE.indication

Nothing shall be done upon reception of this primitive by the adaptation layer.

A.F.3.5 MLME-BEACON-NOTIFY.indication

When a MLME-BEACON-NOTIFY.indication is received, and if an ADPM-DISCOVERY.request is currently operating, the adaptation layer shall add the PANId to the PANDescriptorList which will be forwarded to the upper layer in the ADPM-DISCOVERY.confirm primitive.

A.F.3.6 MLME-GTS.indication

Nothing shall be done upon reception of this primitive by the adaptation layer.

A.F.3.7 MLME-ORPHAN.indication

Nothing shall be done upon reception of this primitive by the adaptation layer.

A.F.3.8 MLME-COMM-STATUS.indication

On reception of this primitive, the adaptation layer shall generate an ADPM-NETWORK-STATUS.indication primitive, with the Status parameter equal to that of the MLME-COMM-STATUS.indication primitive, and the AdditionalInformation parameter equal to the concatenation of the SrcAddr and DstAddr, separated by a ":".

A.F.3.9 MLME-SYNC-LOSS.indication

The adaptation layer shall respond to the reception of this primitive as described in A.3.5.2.

G3 Annex G

(normative)

Device starting sequence of messages

(This annex forms an integral part of this Recommendation.)

Each device shall start with an adpDeviceType attribute of Not_Device, Not_Server (see Table A.21) and then the following procedure is performed:

a) Reset the equipment by sending the ADPM-RESET.request.

b) Set the type of the device to Device or Server mode and optionally set the PIB parameters to configure it.

c) If the equipment is a device it shall perform the following steps:

- Discovery procedure by invoking the ADPM-DISCOVERY.request;
- If there is a device or a server in its POS, it shall then invoke the ADPM-NETWORK-JOIN.request to perform the bootstrapping procedure.

d) Else (the equipment is a server) it shall perform the following steps:

- Discovery procedure by invoking the ADPM -DISCOVERY.request.
- If there is no device in the server's POS, it shall invoke the ADPM-NETWORK-START to start a network; else it should inform the rest of the system that a PAN is already operating in the POS of the device, and may start a new network afterwards as described in A.3.5.1. The procedures and decisions associated with this behaviour are implementation specific.

Equipment cannot send or receive data packets unless it has joined the network.

G3 Annex H

(normative)

6LoWPAN Ad Hoc On-Demand Distance Vector Routing (LOAD)

(This annex forms an integral part of this Recommendation.)

NOTE 1 – This annex is copied from IETF draft-daniel-6lowpan-load-adhoc-routing-03: 6LoWPAN Ad Hoc On-Demand Distance Vector Routing (LOAD). Edited by K. Kim, S. Daniel, G. Montenegro, S. Yoo, N. Kushalnager. June 19, 2007.

NOTE 2 – In this annex, the term "TBD" refers to items left for further study.

Network Working Group

Internet-Draft

Intended status: Standards Track

K. Kim, Ed.

picosNet Corp/Ajou Univ.

S. Daniel Park, Ed.

SAMSUNG Electronics

G. Montenegro

Microsoft Corporation

S. Yoo

Ajou University

N. Kushalnagar

Intel Corp

June 19, 2007

Status of this Memo

This Internet-Draft will expire on December 21, 2007.

Abstract

6LoWPAN Ad Hoc On-Demand Distance Vector Routing (LOAD) is intended for use by IEEE 802.15.4 devices in a 6LoWPAN. It is a simplified on-demand routing protocol based on AODV.

A.H.1 Introduction

The IEEE 802.15.4 standard [IEEE 802.15.4] targets low power personal area networks. The "IPv6 over IEEE 802.15.4" document [I-D.montenegro-lowpan-ipv6-over-802.15.4] defines basic functionality required to carry IPv6 packets over IEEE 802.15.4 networks (including an adaptation layer, header compression, etc.).

Likewise, the functionality required for packet delivery in IEEE 802.15.4 meshes is defined, as mesh topologies are expected to be common in LoWPAN networks. However, neither the IEEE 802.15.4 standard nor the "IPv6 over IEEE 802.15.4" specification provide any information as to how such a mesh topology could be obtained and maintained.

The 6LoWPAN Ad hoc Routing Protocol (LOAD) is a simplified on-demand routing protocol based on AODV [RFC 3561] for 6LoWPAN. Besides the main AODV specification [RFC 3561], several efforts aim at simplifications of the protocol, as in the AODVjr proposal [AODVjr] or the TinyAODV implementation [TinyAODV]. Similarly, DyMO allows for minimalist implementation leaving non-essential functionality as optional [I-D.ietf-manet-dymo]. LOAD enables multihop routing between IEEE 802.15.4 devices to establish and maintain routing routes in 6LoWPAN.

This document defines the message formats, the data structures, and the operations of LOAD.

A.H.2 Requirements notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC 2119].

A.H.3 Overview

This clause describes the distinctive features of LOAD compared to AODV. LOAD is defined to be operating on top of the adaptation layer instead of the transport layer. That is, it creates a mesh network topology underneath and unbeknownst to IPv6 network layer. IPv6 sees a 6LoWPAN as a single link. This is similar to how other technologies regularly create complex structures underneath IP (e.g., ethernet spanning tree bridges, token ring source routing, ATM, etc.). LOAD control packets use the encapsulation defined in [I-D.montenegro-lowpan-ipv6-over-802.15.4]. All LOAD control packets shall use the prot_type value TBD (suggested value of 4).

LOAD assumes the use of either one of the two different addresses for routing: the EUI-64 address and the 16 bit short address of the 6LoWPAN device.

LOAD makes use of broadcast in its route discovery. It does so in order to propagate the Route Request (RREQ) messages. In this specification, such broadcast packets are obtained by setting the PAN id to the broadcast PAN (0xffff) and by setting the destination address to the broadcast short address (0xffff).

LOAD does not use the destination sequence number in order to reduce the size of the control messages and simplify the route discovery process. For ensuring loop freedom, only the destination of a route SHOULD generate a RREP in reply. The intermediate nodes SHOULD not respond with a RREP. By the same reason, LOAD does not use the "Gratuitous RREP".

LOAD MAY use the local repair for a link break during a data delivery. In a local repair, only the destination generates a RREP in reply because of no use of the destination sequence number.

If a local repair fails, LOAD MAY generate a Route Error (RERR) message toward the originator of the data delivery to notify that the destination is no longer reachable by way of the broken link. The format of RERR is simplified to include only one unreachable destination while the RERR of AODV MAY include multiple ones.

LOAD does not use the "precursor list" of AODV to simplify the routing table structure. Notice that AODV uses the precursors for forwarding RERR messages in the event of detection of the loss of the next hop link. In LOAD, RERR is forwarded only to the originator of the failed data delivery, thus no requiring to use the precursor list.

LOAD MAY use the route cost, which is the accumulated link cost from the originator to the destination, as a metric of routing. For this, LOAD utilizes the Link Quality Indicator (LQI) of the 6LoWPAN PHY layer in the routing decision in addition to the hop distance. There are many ways to include LQI in the routing metric. The approach taken by LOAD avoids a route which contains weak links whose LQI is below certain threshold value (i.e., WEAK_LQI_VALUE).

LOAD SHOULD utilize the acknowledged transmission option at the 6LoWPAN MAC layer for keeping track of the connectivity of a route. LOAD uses neither the passive acknowledgements nor the HELLO messages of AODV.

The basic operations of LOAD are route discovery, managing data structures and maintaining local connections. For these operations, LOAD maintains the following two tables: the routing table and the route request table. The routing table stores route information such as destination, next hop node, and status. The route request table stores the temporary route information used in the route discovery process.

There are two different types of 6LoWPAN devices: the reduced function device(RFD) and the full function device (FFD). LOAD SHOULD utilize only FFD for mesh routing. Thus, A FFD SHOULD implement the operations of LOAD and maintain the data structures of LOAD.

A.H.4 Terminology

This clause defines the terminology of LOAD that is not defined in [RFC 3753] and [RFC 3561].

Destination — A node to which data packets are to be transmitted. Same as "destination node".

forward route — A route set up to send data packets from the originator to its destination.

link cost — The link Quality (LQ) between a node and its neighbour node.

link quality indicator (LQI) — A mechanism to measure the Link Quality (LQ) in IEEE 802.15.4 PHY layer [IEEE 802.15.4]. It measures LQ by receiving the signal energy level. A high LQ value implies the good quality of communication (i.e., low link cost).

weak link — A link of which the LQI falls below WEAK_LQI_VALUE.

originator — A node that initiates a route discovery process. Same as "originating node".

route cost — An accumulated link cost as a LOAD control message (RREQ or RREP) passes through the nodes on the route.

reverse route — A route set up to forward a RREP back to the originator from the destination. Same as "reverse route" in [RFC 3561].

A.H.5 Data Structures

A FFD in 6LoWPAN SHOULD maintain a routing table and a route request table. This clause describes the tables and the message formats.

A.H.5.1 Routing Table Entry

The routing table of LOAD includes the following fields:

destination address — The 16 bit short or EUI-64 link layer address of the final destination of a route

next hop address — The 16 bit short or EUI-64 link layer addresses of the next hop node to the destination.

Status — The status of a route. It includes the following states: VALID, INVALID, ROUTE_DISCOVERY, etc.

life time — The valid time in milliseconds before the expiration or the deletion of a route.

A.H.5.2 Route Request Table Entry

Route request table is used for discovering routes. It stores the following route request information until a route is discovered.

route request ID — A sequence number uniquely identifying the particular RREQ when taken in conjunction with the originator.

originator address — The 16 bit short or EUI-64 link layer address of the node which originates a RREQ.

reverse route address — The 16 bit short or EUI-64 link layer address of the next hop node on the reverse route to the originator.

forward route cost — The accumulated link cost along the forward route from the originator to the current node through which a RREQ is forwarded.

reverse route cost — The accumulated link cost along the reverse route from the final destination to the current node through which a RREP is forwarded.

valid time — The time of the expiration or deletion of a route in milliseconds.

A.H.5.3 Message Format

A.H.5.3.1 Route Request (RREQ)

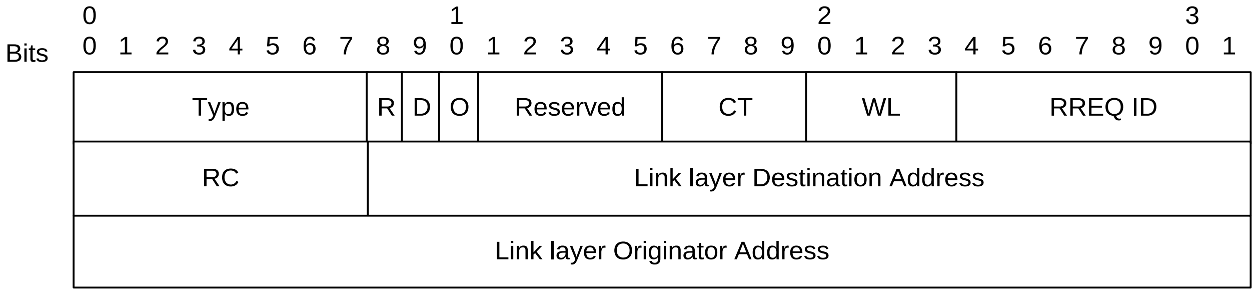


Figure – A.H.1 – RREQ message format

The RREQ message format is shown in Figure A.H.1 and contains the following fields:

Type — 1 for indicating a RREQ message.

CT — Type of route cost. The followings are the current route cost types known:

0 : Hop count while avoiding weak links

1-0xf: TBD

WL The total number of weak links on the routing path from the originator to the sender of the RREQ.

R 1 Local Repair.

D 1 for the 16 bit address of the destination,

0 for the EUI-64 address of the destination.

O 1 for the 16 bit address of the originator,

0 for the EUI-64 address of the originator.

RC(Route cost) The accumulated link cost of the reverse route from the originator to the sender of the RREQ. The type of link cost is specified by CT.

RREQ ID A sequence number uniquely identifying the particular RREQ when taken in conjunction with the originator.

Reserved 0; ignored on reception.

Link layer Destination Address The 16 bit short or EUI-64 link layer address of the destination for which a route is supplied.

Link layer Originator Address The 16 bit short or EUI-64 link layer address of the node which originated the Route Request.

A.H.5.3.2 Route Reply (RREP)

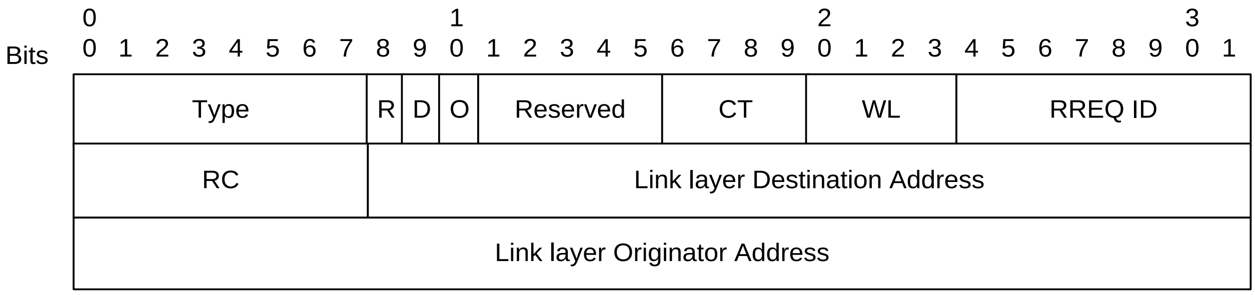


Figure A.H.2 – RREP message format

The RREP message format is shown in Figure A.H.2 and contains the following fields:

Type 2 for indicating a RREP message.

CT Type of route cost. The followings are the current route cost types known:

0 : Hop count while avoiding weak links

1-0xf: TBD

WL The total number of weak links on the routing path from the originator of the RREP to the sender of the RREP.

R 1 Local Repair.

D 1 for the 16 bit address of the destination,

0 for the EUI-64 address of the destination.

O 1 for the 16 bit address of the originator,

0 for the EUI-64 address of the originator.

Reserved 0; ignored on reception.

RC(Route cost) The accumulated link cost of the route from the originator of the RREP to the sender of the RREP. The type of link cost is specified by CT.

RREQ ID A sequence number uniquely identifying the particular RREQ when taken in conjunction with the originator.

Link layer Destination Address The 16 bit short or EUI-64 link layer address of the destination for which a route is supplied.

Link layer Originator Address The 16 bit short or EUI-64 link layer address of the node which originated the Route Request.

A.H.5.3.3 Route Error (RERR)

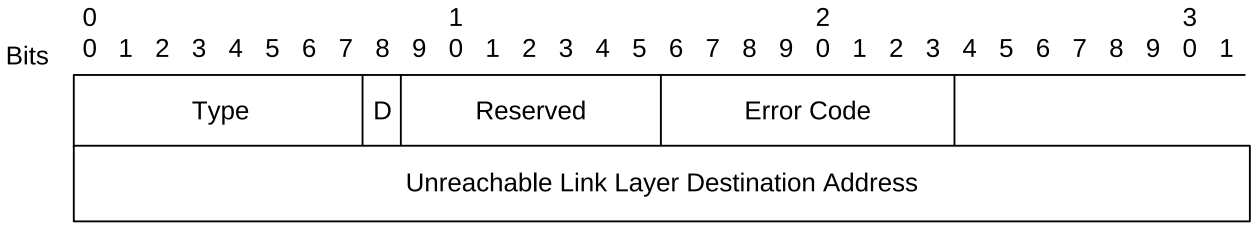


Figure A.H.3 – RERR message format

The RERR message format is shown in Figure A.H.3 and contains the following fields:

Type 3 for indicating a RERR message.

D 1 for the 16 bit address of the destination,

0 for the EUI-64 address of the destination.

Reserved 0; ignored on reception.

Error Code Numeric value for describing error.

0x00 = No available route

0x01 = Low battery

0x02 = routing cost not supported

0x03 – 0xff = reserved (TBD)

Unreachable Link Layer Destination Address The 16 bit short or EUI-64 link layer address of the final destination that has become unreachable due to a link break.

A.H.6 Operation

A.H.6.1 Generating Route Request

The basic operations of LOAD include route discovery, managing data structures and maintaining local connections. A node maintains the following two tables for routing: the routing table and the routing request table.

During the discovery period, an originator, a node that requests a route discovery, generates a Route Request (RREQ) message with the RREQ ID which was incremented by one from the previous RREQ ID value.

A node SHOULD NOT originate more than RREQ_RATELIMIT RREQs per second. After broadcasting a RREQ, a node waits for a RREP. If a route is not discovered within

NET_TRAVERSAL_TIME milliseconds, the node MAY try again the discovery process a maximum of RREQ_RETRIES times.

A.H.6.2 Processing and Forwarding Route Request

Upon receiving a RREQ, an intermediate FFD node tries to find the entry of the same originator address and RREQ ID pair in the route request table. If the entry is found, the node just discards the RREQ. Otherwise, the node creates a reverse route to the originator in the routing table and a RREQ entry in the route request table. It then checks whether the link through which the RREQ is received is a weak link or not. If the link is a weak link, the node adds 1 to the WL field of the RREQ. Then, the node forwards the RREQ.

A.H.6.3 Generating Route Reply

When the destination receives a RREQ, it tries to find the entry of the same originator address and RREQ ID pair in the route request table. If the entry is found, the destination compares the route cost of the RREQ with the forward route cost of the entry. If the cost of the RREQ is better than(i.e., less than) that of the entry, the destination updates the reverse route to the originator in the routing table and generates a RREP in reply. If the cost of the RREQ is not less than that of the entry, the destination just discards the RREQ.

If the CT field of the RREQ is 0 (i.e., hop count while avoiding weak links), the route cost becomes a tuple of (WL, RC) and is ordered lexicographically. That is, the route cost (WL,RC) is said to be better (or smaller) than or equal to (WL',RC') if the following condition holds.

(WL,RC) <= (WL',RC') if and only if WL < WL', or WL == WL' and RC <= RC'

A.H.6.4 Receiving and Forwarding Route Reply

Upon receiving a RREP, an intermediate node checks whether the link through which the RREP is received is a weak link or not. If the link is a weak link, the node add 1 to the WL field of the RREP.

The node then checks whether it has a route entry for the destination of the RREP (i.e., the originator of the corresponding RREQ). If it does not have the route entry, it just discards the RREP. Otherwise, it also checks for the existence of the corresponding RREQ entry (which has the same RREQ ID and originator address pair as that of the RREP) in the route request table. If there is no such entry, then it just discards the RREP.

If there is such an entry and the entry has worse reverse route cost (i.e., higher value) than the route cost of the RREP, the node updates the entry with the information of the RREP and forwards it to the previous hop node toward the destination of the RREP. If the entry has better reverse route cost (i.e., lower value) than that of the RREP, the node just discards the RREP.

If the CT field of the RREP is 0 (i.e., hop count while avoiding weak links), the route cost becomes a tuple of (WL,RC) and is ordered lexicographically.

During the delivery of the RREP to the originator, the route cost value of the RREP is accumulated on the reverse route from the destination to the originator.

A.H.6.5 Local Repair and Route Error (RERR) Messages

If a link break occurs or a device fails during the delivery of data packets, the upstream node of the link break MAY repair the route locally. To repair a route, the node disseminates a RREQ with the originator address set to its own address and the destination address set to the data packet's destination address. In this case, the 'R flag' of the RREQ is set to 1. The data packet is buffered during the route discovery period. If the destination node receives the RREQ for a route repair, it responds with a RREP of which the 'R flag' is also set to 1.

If the repairing node cannot receive a RREP from the final destination until the end of the route discovery period, it unicasts a RERR with an error code that indicates the reason of the repair failure to the originator. A repairing node SHOULD NOT generate more than RERR_RATELIMIT RERRs per second. Then, the buffered data packet is discarded. If the originator that sends a data packet receives the RERR, it MAY try to reinitiate route discovery.

When the repairing node receives a RREP from the destination during the route discovery period, it updates the routing table entry information from the RREP. Then the node transmits the buffered data packet to the destination through the new route.

A.H.7 Configuration Parameters

This clause describes the default values for some important parameters associated with LOAD operations.

Parameter Name	Value
--------------------	-------------
NET_TRAVERSAL_TIME	TBD
RREQ_RETRIES	3
RREQ_RATELIMIT	2
RERR_RATELIMIT	2
WEAK_LQI_VALUE	8

A.H.8 IANA Consideration

This document needs an additional IANA registry for the prot_type value that indicates the LOAD format.

A.H.9 Security Considerations

The security considerations of the [RFC 3561] are applicable to this document. As described in the charter of the 6lowpan, e.g., LOAD will also try to reuse existing security considerations related to Ad hoc routing protocols. Further considerations will be studied in the next version.

A.H.10 Acknowledgments

Thanks to the authors of RFC 3753 and RFC 3561, as parts of this document are patterned after theirs. Thanks to Nandakishore Kushalnagar, Byeong-Hee Roh, Myung-ho Jung, Dae-hong Son, and Minho Lee for their useful discussions and supports for writing this document.

A.H.11 References

A.H.11.1 Normative Reference

…………………………

…………………………

A.H.11.2 Informative Reference

…………………………

…………………………

G3 Annex I

(normative)

Commissioning in 6LoWPAN

(This annex forms an integral part of this Recommendation.)

NOTE 1 – This annex is copied from IETF draft-6lowpan-commissioning-02: Commissioning in 6LoWPAN. Edited by K. Kim, S. Shams, S. Yoo, S. Park, G. Mulligan. July 15, 2008.

NOTE – In this annex, the term "TBD" refers to items left for further study.

Network Working Group

Internet-Draft

Intended status: Standards Track

Expires: January 16, 2009

K. Kim, Ed.

S. ShamsAjou University

picosNet Corp/Ajou Univ.

S. Park, Ed.

S. Yoo

SAMSUNG Electronics

G. Mulligan

July 15, 2008

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on January 15, 2009.

Copyright Notice

Abstract

The commissioning process defines the startup procedure executed by any 6LoWPAN device. This document defines the startup procedure that should be followed by a 6LoWPAN device in any open or secured network.

A.I.1 Introduction

6LoWPAN is a low-power wireless personal area network(LoWPAN) which is comprised of the IEEE 802.15.4-2006 standard [IEEE 802.15.4] devices. One of the design goal for 6LoWPAN architecture is to ensure minimum human intervention during provisioning a sensor device in a PAN. However, a 6LoWPAN device requires a set of pre-deployed information, called LoWPAN Information Base(LIB), to find the right PAN, to successfully join with the PAN, and to establish communication within the PAN. A device needs specific procedure, what we named as a Bootstrapping protocol for 6LoWPAN device, to collect those information from LoWPAN Bootstrapping Server (LBS) and to start communication in a PAN. This procedure needs to be well defined for interoperability of devices from different vendors. This procedure involves extracting LIB, security credentials, becoming part of existing network, obtaining 16-bit short address, and IP settings.

A.I.2 Terminology

Active Scan An active scan is used by a device to locate all coordinators transmitting beacon frames within its personal operating space, which is provided by IEEE 802.15.4. It requests other devices to transmit the beacon frame.

Association An IEEE 802.15.4 device can be assigned a dynamic 16 bit short address during an association operation with a neighbour device (or router) which is also called as the parent device. After getting the short address, a device can communicate with its parent or child by using only the assigned short address.

Coordinator A full-function device (FFD) which is the principal controller of a 6LoWPAN. It is also called as PAN coordinator. It MAY initiate the synchronization of the entire 6LoWPAN by transmitting beacons.

ED Scan An ED scan allows a device to obtain a measure of the peak energy in each requested channel, which is provided by IEEE 802.15.4.

Full Function Device (FFD) A device implementing the complete protocol set of IEEE 802.15.4. It is capable of operating as a router (multi-hop packet forwarding) for its associated neighbours.

Neighbour Table A table which has the information of neighbour devices in a personal operating space.

LoWPAN Bootstrapping Information Base (LIB) A set of pre-deployed information that is necessary for a particular 6LoWPAN device to find the desired PAN and to successfully join with the PAN. We categorize this information into two groups; PAN Specific Information (PSI), which is the same for every device in a PAN, (for example, PAN ID), and Device Specific Information(DSI), which is specific for each particular node (for example short address).

PSI: PAN Specific Information Inside the LIB, a portion of information, called PSI, is the same for every device in the target PAN. For example, PAN_ID, PAN_Type, etc.

DSI: Device Specific Information Inside the LIB, other than PSI, there is some information that may vary from device to device. For example, Role_of_Device, Short_Addr, etc.

LoWPAN BootStrapping Device (LBD) LBD is a device that is needed to be deployed in the target network. LBD is assumed to have no priori information about the 6LoWPAN within which it is going to join. The only information it has is the EUI-64 address and a "Join key" (in case of secured PAN).

LoWPAN BootStrapping Server (LBS) An entity that contains LIB of each device to be bootstrapped. It indexes this information with the EUI-64 address of each 6LoWPAN device. LBS has two modules in it; Network management & Account Module (NAM) and Authentication Module (AM). NAM keeps track of the LIB of each device indexed by EUI-64 address whereas AM participates in authentication process on behalf of LBD using LBD's 'Authentication credentials'. Based on the 'LBP Message', LBS verifies LBD with the help of Authentication server (in case of secured PAN) and sends ACCEPT message with necessary information otherwise it sends DECLINE message. In the case of secured PAN, LBS initiates authentication mechanism issuing Authentication request into appropriate format that is acceptable by particular authentication server. Any challenge or reply message from the Authentication server is encapsulated in the 'LIB message' by LBS and is sent back to the LBD through LBA.

LoWPAN BootStrapping Agent (LBA) A FFD that has already joined in the PAN and thus, it is already a member of the PAN. It is also a neighbour of a new LBD, and thus it helps the bootstrapping LBD by receiving LBP message from LBD and forwarding it to LBS.

Open 6LoWPAN An open 6LoWPAN is a PAN where any device is welcomed.

Close 6LoWPAN A close 6LoWPAN is a PAN where only pre-defined set of devices are allowed to join based on their EUI-64 address. This account is managed by LBS. If close 6LoWPAN is secured, it is called secured 6LoWPAN.

Secured 6LoWPAN Secured 6LoWPAN is a Close 6LoWPAN that also maintains secured message exchange in the PAN.

PAN Id The 16 bit 6LoWPAN identifier which is administratively assigned to a 6LoWPAN and is unique within the PAN.

Passive Scan A passive scan, like an active scan, is used by an FFD to locate all coordinators transmitting beacon frames within its personal operating space, which is provided by IEEE 802.15.4. The difference is that the passive scan is a receive-only operation and does not request the beacon frame.

Personal Operating Space (POS) The area within the reception range of the wireless transmission of a IEEE 802.15.4 packet.

Reduced Function Device (RFD) A IEEE 802.15.4 device of 6LoWPAN which does not have the functionality of the router. That is, it can not forward IPv6 packets to the next hop device. It can only be the end device of 6LoWPAN.

Short Address A 16 bit address dynamically assigned to a device from the PAN.

A.I.2.1 Requirements notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC 2119].

A.I.3 Bootstrapping

Bootstrapping is defined as collecting LIB from LBS, obtaining security credentials (optional), associating with the right PAN, obtaining 16-bit short address (optional), and constructing IPv6 address using IPv6 prefix. Specifically, this includes the process of starting the network, associating with other nodes, obtaining the unique IPv6 address, and constructing security credentials for 6LoWPAN.

A.I.3.1 Resetting the device

After the device is started, it first performs a MAC layer reset.

A.I.3.2 Scanning through channels

During this phase, functions supported by 802.15.4 are used for scanning channels. Appendix (A.1) shows the scanning process in 802.15.4. For getting the information of other devices within POS, the device should perform scan. The device can use either an active scan or a passive scan. During scanning procedure, the device receive beacon frames from other devices.

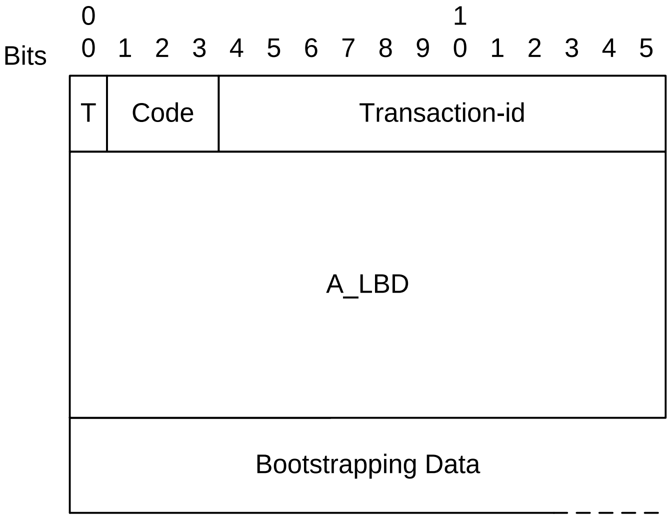
A.I.3.3 LoWPAN BootStrapping Mechanism

This protocol defines mechanism to extract LIB from currently unknown LBS and also defines a message format for LIB message exchange. In this protocol, LBD exchanges LBP message with LBS through its one hop neighbour LBA. So, at the beginning of LBP, it needs to find an LBA using 'LBA discovery phase' that is described in clause 3.3.2

3.3.1. LoWPAN BootStrapping Protocol message format

In this clause we define a message format which is necessary for LBP.

3.3.1.1. LBP message



T Type of message

It defines message type. value '0' represents 'Message from LBD' and '1' represents 'Message to LBD'.

Code:

000, 1xx: Reserved.

001 ACCEPTED. Authentication of LBD has been accepted.

010 CHALLENGE. It indicates that authentication process has not been finished. Authentication server has sent some challenge that has to be replied by LBD.

011 DECLINE. In the case of unsecured 6LoWPAN, LBS may send this code to indicate that LBD's EUI-64 address is not allowed to join the PAN.In case of secured 6LoWPAN, LBS may send this code to indicate that LBD's EUI-64 address is not allowed to join the PAN or the authentication of the LBD is failed.

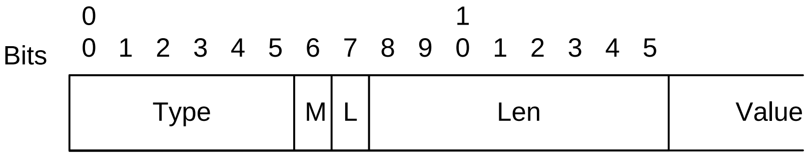
Seq Sequence Number

Seq identifies the number of messages transmitted by LBD. Corresponding incoming message from LBS should also have the same Seq.

A_LBD Address of Bootstrapping Device (LBD)

64-bit EUI-64 address of LBD.

Bootstrapping Data Format of bootstrapping data is given below.



Type: 6-bit represents the ID of the attribute in LIB if 'L' bit is set. Otherwise, this field defines particular authentication type.

A list of authentication mechanism and their corresponding 'Type' is TBD.

M: Type of the Attribute

This field defines the type of the attribute in LIB; whether it is PAN Specific Information (PSI) or Device Specific Information (DSI). 1 represents PSI and 0 represents DSI.

Len: 8-bit represents the length of the value in octet.

Value: This field represents the corresponding data of the type.

A.I.3.4 LoWPAN Bootstrapping Information Base

One of the important goal of LBP is to receive a set of information from LBS by a joining LBD. This information comprises of PSI and DSI. Following table shows attribute name, attribute ID (attr_ID), purpose of the attribute and type of it.

Attribute Name........Attribute ID......Attribute Description PSI/DSI

Attribute Name	Attr_ID	Type	Attribute Description
PAN_ID	1	P	This is the network identification for the default network
PAN_type	2	P	Secured/closed/open
Address_of_LBS	3	P	Address of the LBS. 0x0000 in case of no LBS. For example in open 6LoWPAN.
Join_Time	4	P	It specifies the time when this node should start trying to join the target PAN.
Role_of_Device	5	D	Agent/No_Agent
Allow_LBA_To_Send_PSI	6	P	This attribute allows any SF to provide GI to CD after getting the positive reply from LBS.
Short_Addr	7	D	16-bit address for new device which is unique inside the PAN
Short_Addr_Distribution_ Mechanism	8	P	Its Value is either 0 or 1 representing central or distributed respectively. If it is central, short address is provided by LBS itself otherwise assigning short address is
Other_Device_Specific _Info	15	D	Using this attribute, a device and LBS can exchange any types of data or security key required by the device.

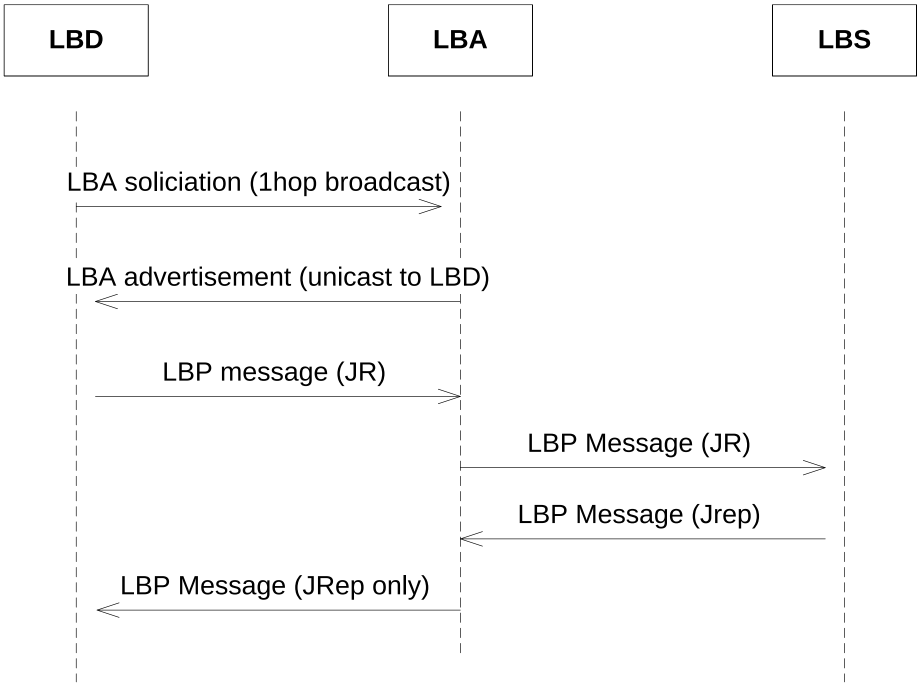
A.I.3.4.1 LBA discovering phase

LBD has to send LBP message to the LBS server under the support of a LBA. To find the LBA, it broadcasts a LBA solicitation message within its one hop neighbours and waits for a LBA advertisement. Any device capable of being LBS/LBA replies to the broadcast specifying its capability as LBS/LBA. If there is any LBS in its neighbour, LBD selects that LBS otherwise it selects one of the LBAs.

A.I.3.4.2 LoWPAN Bootstrapping Protocol (LBP)

LBD sends LBP message to LBA, as it does not know the address or path to the LBS of the target PAN. LBA forwards the LBP message to LBS on behalf of LBD. LBS replies with one or multiple LBP messages destined to LBA as LBD still is not part of the network. If the network is secured 6LoWPAN and the LBD is an authentic node, we assume that LBD has necessary pre-deployed keys and the knowledge of the authentication mechanism necessary to authenticate in target PAN. In this case, LBD sends necessary information in the 'bootstrapping data' field so that LBS can initiate the authentication process using that 'authentication credentials'. LBS converts the LBP message into appropriate authentication request for the particular authentication server and sends it. A reply/challenge from the authentication server, for example EAP authenticator or AAA server, is encapsulated in LBP message's 'bootstrapping data' field and is sent back to the LBD through LBA. LBA also keeps track of the successful authentication, failed authentication and incomplete conversation of the authentication process, and maintains a 'black list' of malicious devices to avoid repeated attack. Detecting malicious device based on those 3 information and marking that node as 'Black listed' belongs to the scope of security policy and out of the scope of this draft.

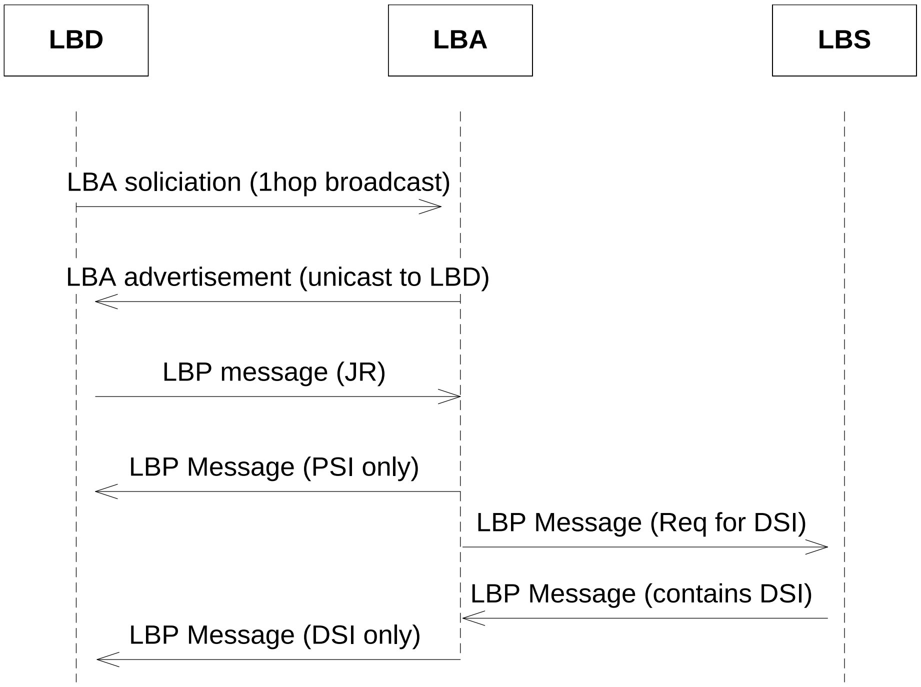
Following figure shows a simple example of Bootstrapping mechanism.



JR = Join Request, JRep = Join Reply.

3.3.5. Bootstrapping in open 6LoWPAN:

An open 6LoWPAN network, usually welcomes any willing LBD. In this case, it does not need to wait for reply from LBS. Instead, LBA can provide GI from its own LIB and can forward LIB request to LBS simultaneously.


LBD
LBA
LBS
LBA soliciation (1hop broadcast)
LBA advertisement (unicast to LBD)
LBP message (JR)
LBP Message (PSI only)
LBP Message (Req for DSI)
LBP Message (contains DSI)
LBP Message (DSI only)

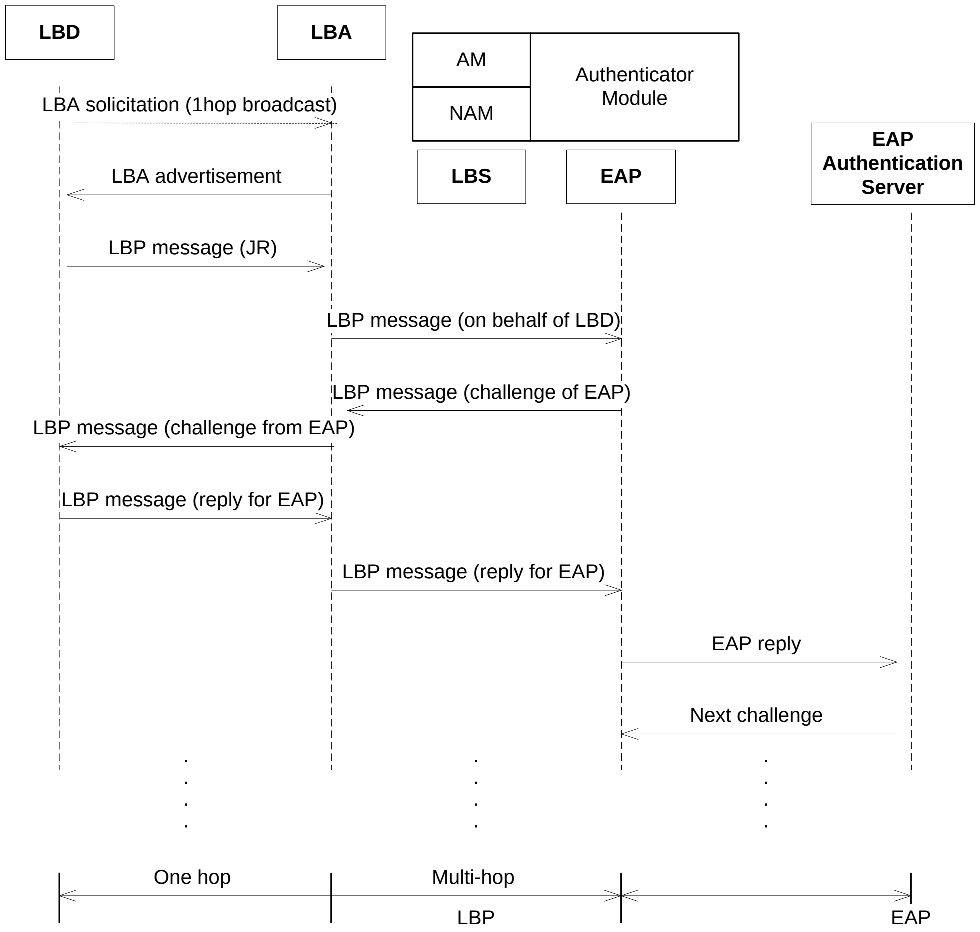
JR= Join Request.

A.I.3.4.3 LBP in secured 6LoWPAN

In secured 6LoWPAN, LBD must has to exchange authentication credentials using its join key. Apart from requesting network resources, in the case of secured network, this process may need to exchange several encrypted message between LBD and authentication server. LBA and LBS serves as 'secured tunnel' for authentication message exchange process. Both LBA and LBS keep the account of the last LIB request/reply processed by themselves.

Example: LBP with EAP

The following figure shows how LBP with other authentication protocol ike EAP works. At first LBD broadcasts a LIB request(1 hop) to LBA. LBA already has a secured route to LBP so it just unicasts the LIB request to LBS. LBS sends an EAP packet prepared with LBD's authentication credentials and sends it to authenticator. it is also possible that LBS entity and authenticator entity resides on a single system. As discussed earlier, LBS serves as translator between LBP and EAP message exchange in this authentication process and finally when AM indicates the success of authentication, it sends all network resources along with the ACCEPTED code. In the case of failure in authentication process, DECLINE code is sent to LBD.


LBD
LBA
AM
NAM
Authenticator Module
LBS
EAP
EAP Authentication Server
LBA solicitation (1hop broadcast)
LBA advertisement
LBP message (JR)
LBP message (on behalf of LBD)
LBP message (challenge of EAP)
LBP message (challenge from EAP)
LBP message (reply for EAP)
LBP message (reply for EAP)
EAP reply
Next challenge
One hop
Multi-hop
LBP
EAP

A.I.3.4.4 Role of Entities in LBP:

Role of LoWPAN Bootstrapping Device (LBD):

– It selects LBA using LBA discovery phase.

– If it doesn't find any LBA, it gives up after waiting for certain amount of time.

– if it receives any LBP message with code "CHALLENGE", it must send another LBP message containing the appropriate value against the challenge/query in the bootstrapping data field.

– It MUST increment seq for every new LBP message. For retransmission seq should remain same.

Role of LoWPAN Bootstrapping Agent (LBA):

When LBA receives LBP message from LBD.

1. If the LBD is already in the Black List, discard.

2. If the LBD is new, and 6LoWPAN is open network

   a) Send 'LBP message' with ACCEPTED along with all PSI from its own LIB.

   b) If there is any LBS in the PAN, Forward the 'LBP message' to LBS for DSI.

3. If the LBD is old, and 6LoWPAN is open network

   a) If it matches with the last seq no. send the last reply.

   b) Otherwise discard.

4. If the LBD is new, and 6LoWPAN is secured network

   a) Forward the LBP message to LBS.

5. If the LBD is old, and 6LoWPAN is secured network

   a) If it matches with the last seq no. send the previously saved last LBP message 'for LBD'.

   b) If the LBP has completed, discard.

   c) If the LBP is 'CHALLENGE' and new seq is right next of the last one, forward the message to LBS.

When LBA receives LBP message from LBS (for LBD)

– If it is ACCEPTED and 16-bit short address is the responsibility of LBA, it calculates and appends the 16-bit short address with the LIB reply.

– Otherwise, if it is ACCEPTED, DECLINED or CHALLENGE, forward it to the corresponding LBD.

– If it is not ACCEPTED or DECLINED, delete previously saved LBP message and save this LBP message.

– If it is DECLINED, based on the security policy, mark it as 'Blacklisted'.

– If there is no activity in some of the flow (LBD-LBS pair), mark the LBD and based on the security policy include it in 'Black list'.

Role of LoWPAN Bootstrapping Server (LBS):

In the case of open 6LoWPAN

– If the LBD is 'valid' that means its EUI-64 is in accepted list or not in the rejection list, it sends ACCEPTED code and necessary DSI and 16-bit short address(if the address should be assign centrally).

In the case of secured 6LoWPAN

– AM of LBS determines authentication server for particular EUI address and sends authentication mechanism initiation with the authentication credentials to that authentication server.

– When it gets reply from authentication server, if it is success, it prepares a success reply if it is failure, it prepares a failure reply f it is challenge/query, it prepares processing reply for LBDand sends to LBA.

– When AM module receives success from authentication server, it informs success to NAM module and sends the success response to NAM.NAM then, sends DSI along with the response in LBP message.

A.I.3.5 Assigning the short address

During LBP procedure, LBD may set a short address either by itself or receiving the address from the PAN. The short address must be unique in a PAN and may be given by a centralized or distributed way.

One of the approach to distribute the short address among the LBDs is centralized fashion where a centralized entity (e.g., LBS) assigns 16-bit short address for LBD. Allocation of short address MAY be based on First-Available-Address-First or randomly chosen one or using any other algorithm.

Distributed approach is another way to assign 16-bit short address to LBDs. In this approach, LBA assigns short address to the joining device, LBD. A hierarchical addressing scheme could be used by LBA in this purpose. Following figure describes the address calculation scheme. This scheme requires one parameter MC, the maximum number of addresses a LBA can assign. If the present LBD is the first children, then it gets the short address by following formula,

$$FC = MC * AP + 1$$

where FC is the LBD address, and AP is the address of the LBA.

If LBD is not the first child of this LBA, it receives the address which is next to the last address assigned by that LBA.

For example, if LBA(1) assigned address 6 to its last LBD, it assigns address 7 to its next LBD.

MC = 4

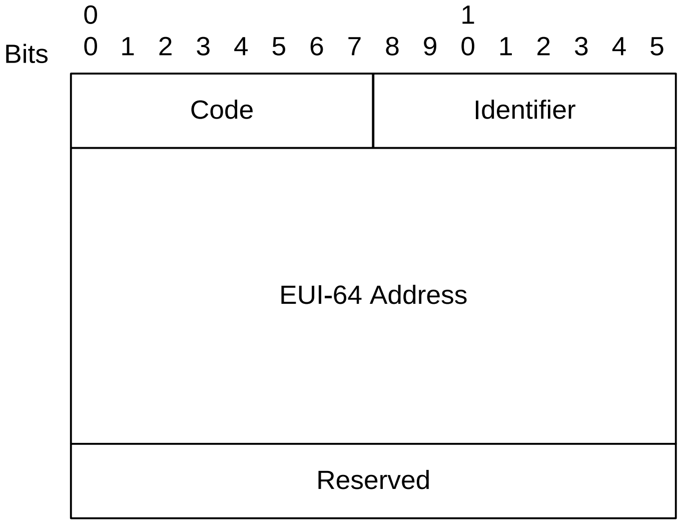
```
                  (0)            <= Coordinator
                 //   \\
                /  |   |  \
              /   /      \   \
            (1) (2)    (3)  (4)        <= Routers
           // \\ ......... // \\
         /  /   \ \        / /   \  \
       (5) (6) (7)(8)..(17)(18)(19)(20)
                   // \\
                 /  /   \  \
   ...........(69) (70)(71) (72)........
```

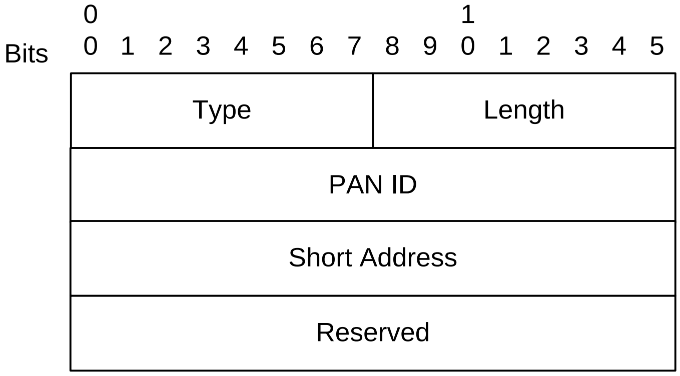
Fig. – The assignment scheme of the short address

A.I.3.6 Obtaining IPv6 address

The IPv6 interface identifier of a device can be obtained as described in Section 6 of [IETF RFC 4944]. After having a unique IPv6 interface identifier, the device begins to obtain an IPv6 address prefix. The IPv6 address prefix for a particular 6LoWPAN is stored by the IPv6 router in the 6LoWPAN. ICMPv6 is used to share these parameters. Routers in 6LoWPAN are supposed to broadcast Router Advertisements(RA) messages periodically. The RA message must contain the prefix option which can be used in the 6LoWPAN. Devices wish to obtain IPv6 address prefix may wait for an RA message until RA_WAIT_TIME elapsed. After that, if no RA message is received, they may broadcast Router Solicitation RS) message for requesting the RA message.

The RS and RA messages can have additional option fields as described in [RFC 4861]. Source/Target link-layer address option field should contain the EUI-64 address or the combined address with PAN ID and 16bit short address of the source or target device as below. The RS and RA messages can have additional option fields as described in [RFC 4861]. Source/Target link-layer address option field should contain the EUI-64 address or the combined address with PAN ID and 16 bit short address of the source or target device as below.





Source/Target Link-layer Address option field

Multiple IPv6 routers could form a single or multiple 6lowpan(s). If there are multiple routers in a 6LoWPAN, the device should consider which one is to be selected as a default router. One possible way of selection is to compare the hop counts travelled of the RA message of each router. The detailed algorithm for the selection is TBD.

A.I.3.7 Configuration Parameters

This clause gives default values for some important parameters associated with the 6LoWPAN commissioning protocol. A particular node may wish to change certain of the parameters.

Parameter Name	Value
---------------------	-----
CHANNEL_LIST	0xFFFF800
SCAN_DURATION	3
SUPERFRAME_ORDER	15
BEACON_ORDER	15
START_RETRY_TIME	1000 msec
JOIN_RETRY_TIME	4000 msec
ASSOCIATION_RETRY_TIME	4000 msec

A.I.4 IANA Consideration

TBD.

A.I.5 Security Considerations

IEEE 802.15.4 devices is required to support AES link-layer security. MAC layer also provides all keying material necessary to provide the security services. It isn't defined, however, when security shall be used especially combining with Bootstrapping. After the device start and join the network, security services such as key management and device authentication should be done automatically. Detailed algorithm for security on Bootstrapping is TBD.

A.I.6 Contributors

Thanks to the contribution from MD. Aminul Haque Chowdhury (Ajou Univ) and Chae-Seong Lim (Ajou Univ) for the review and useful discussion for writing this document.

A.I.7 Acknowledgments

Thanks to Hamid Mukhtar (PicosNet/Ajou Univ), Jae-ho Lee (NIA), and Dong-Gyu Nam (NIA) for their useful discussion and support for writing this document.

A.I.8 References

A.I.8.1 Normative References

[RFC 2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC 4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

[RFC 4944] Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks", RFC 4944, September 2007.

IEEE 802.15.4] IEEE Computer Society, "IEEE Std. 802.15.4-2006".

A.I.8.2 Informative References

[RFC 3513] Hinden, R. and S. Deering, "Internet Protocol Version 6 (IPv6) Addressing Architecture", RFC 3513, April 2003.

Authors' Addresses

Ki-Hyung Kim (editor)

picosNet Corp/Ajou Univ.

San 5 Wonchun-dong, Yeongtong-gu

Suwon-si, Gyeonggi-do 442-749

KOREA

Phone: +82 31 219 2433

Email: kkim86@picosnet.com

S M Saif Shams

picosNet Corp/Ajou Univ.

San 5 Wonchun-dong, Yeongtong-gu

Suwon-si, Gyeonggi-do 442-749

KOREA

Phone: +82 010 8690 8532

Email: saif95bd@gmail.com

Seung Wha Yoo
Ajou University
San 5 Wonchun-dong, Yeongtong-gu
Suwon-si, Gyeonggi-do 442-749
KOREA
Phone: +82 31 216 1603
Email: swyoo@ajou.ac.kr

Soohong Daniel Park (editor)
SAMSUNG Electronics
Mobile Platform Laboratory, SAMSUNG Electronics 416 Maetan-3dong
Yeongtong-gu, Suwon-si, Gyeonggi-do 442-742
KOREA
Phone: +82 31 200 4508
Email: soohong.park@samsung.com

Geoffrey Mulligan
Email: geoff@mulligan.com

Annex B

PRIME power line communications DLL

(This annex forms an integral part of this Recommendation.)

NOTE – This is a stand-alone annex which can be implemented independently from the main body of this Recommendation.

B.1 Introduction

This document is the technical specification for OFDM PRIME technology.

B.1.1 Scope

This document specifies a MAC layer and a Convergence layer for complexity-effective, narrowband (<200 kbps) data transmission over electrical power lines that could be part of a Smart Grid system.

B.1.2 Overview

The purpose of this document is to specify a narrowband data transmission system based on OFDM modulations scheme for providing mainly core utility services.

The specification currently describes the following:

- A MAC layer for the power line environment (see chapter 4).
- A Convergence layer for adapting several specific services (se chapter 5).
- A Management Plane (see chapter 6)

The specification is written from the transmitter perspective to ensure interoperability between devices and allow different implementations.

B.1.3 Normative references

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

[IEC 61334-4-1] IEC 61334-4-1 Ed.1996, *Distribution automation using distribution line carrier systems – Part 4: Data communication protocols – Section 1: Reference model of the communication system*.

[IEC 61334-4-32] IEC 61334-4-32 Ed.1996, *Distribution automation using distribution line carrier systems – Part 4: Data communication protocols – Section 32: Data link layer – Logical link control (LLC).*

[IEC 61334-4-511] IEC 61334-4-511 Ed. 2000*, Distribution automation using distribution line carrier systems – Part 4-511: Data communication protocols – Systems management – CIASE protocol*.

[IEC 61334-4-512] IEC 61334-4-512, Ed. 1.0:2001, *Distribution automation using distribution line carrier systems – Part 4-512: Data communication protocols – System management using profile 61334-5-1 – Management Information Base (MIB).*

[IEEE 802-2001] IEEE Std 802-2001 (R2007), *IEEE Standard for Local and Metropolitan Area Networks. Overview and Architecture*.

[IETF RFC 768] IETF RFC 768 (1980), *User Datagram Protocol (UDP).*

[IETF RFC 791] IETF RFC 791 (1981), *Internet Protocol, DARPA Internet Program, Protocol Specification.*

[IETF RFC 793] IETF RFC 793 (1981), *Transmission Control Protocol (TCP).*

[IETF RFC 1144] IETF RFC 1144 (1990), *Compressing TCP/IP Headers for Low-Speed Serial Links*.

[IETF RFC 2131] IETF RFC 2131 (1997), *Dynamic Host Configuration Protocol (DHCP).*

[IETF RFC 2460] IETF RFC 2460 (1998), *Internet Protocol, Version 6 (IPv6) Specification.*

[IETF RFC 3022] IETF RFC 3022 (2001), *Traditional IP Network Address Translator (Traditional NAT).*

[PUB 197] NIST FIPS PUB 197 (2001), *Advanced Encryption Standard (AES).*

[SP 800-38A] NIST SP 800-38A (2001), *Recommendation for Block Cipher Modes of Operation. Methods and Techniques*.

[SP 800-57] NIST SP 800-57 (2007), *Recommendation for Key Management – Part 1: General* (Revised).

B.1.4 Document conventions

This document is divided into chapters and annexes. The document body (all chapters) is normative (except for italics).

Binary numbers are indicated by the prefix '0b' followed by the binary digits, e.g., '0b0101'. Hexadecimal numbers are indicated by the prefix '0x'.

Mandatory requirements are indicated with 'shall' in the main body of this document.

Optional requirements are indicated with 'may' in the main body of this document. If an option is incorporated in an implementation, it shall be applied as specified in this document.

roof (.) denotes rounding to the closest higher or equal integer.

floor (.) denotes rounding to the closest lower or equal integer.

A mod B denotes the remainder (from 0, 1, …, B-1) obtained when an integer A is divided by an integer B.

B.1.5 Definitions

B.1.5.1 Terms defined elsewhere

None.

B.1.5.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

B.1.5.2.1 base node: The master node which controls and manages the resources of a subnetwork.

B.1.5.2.2 beacon slot: The location of the beacon PDU within a frame.

B.1.5.2.3 destination node: A node that receives a frame.

B.1.5.2.4 downlink: Data travelling in the direction from the base node towards the service nodes.

B.1.5.2.5 level (PHY layer): When used in the physical layer (PHY) context, it implies the transmit power level.

B.1.5.2.6 level (MAC layer): When used in the medium access control (MAC) context, it implies the position of the reference device in switching hierarchy.

B.1.5.2.7 MAC frame: A composite unit of abstraction of time for channel usage. A MAC frame is comprised of one or more beacons, one SCP, and zero or one CFP. The transmission of the beacon by the base node acts as a delimiter for the MAC frame.

B.1.5.2.8 neighbour node: Node A is neighbour node of node B if A can directly transmit to and receive from B.

B.1.5.2.9 node: Any one element of a subnetwork which is able to transmit to and receive from other subnetwork elements.

B.1.5.2.10 PHY frame: The set of OFDM symbols and preamble which constitute a single PHY layer protocol data unit (PPDU).

B.1.5.2.11 preamble: The initial part of a PHY frame, used for synchronization purposes.

B.1.5.2.12 registration: The process by which a service node is accepted as member of a subnetwork and allocated an LNID.

B.1.5.2.13 service node: Any one node of a subnetwork which is not a base node.

B.1.5.2.14 source node: A node that sends a frame.

B.1.5.2.15 subnetwork: A set of elements that can communicate by complying with this Recommendation and share a single base node.

B.1.5.2.16 subnetwork address: The property that universally identifies a subnetwork. It is its base node EUI-48 address.

B.1.5.2.17 switching: Providing connectivity between nodes that are not neighbour nodes.

B.1.5.2.18 unregistration: The process by which a service node leaves a subnetwork.

B.1.5.2.19 uplink: Data travelling in the direction from the service node towards the base node.

B.1.6 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

AC	Alternating Current
AES	Advanced Encryption Standard
AMM	Advanced Meter Management
ARQ	Automatic Repeat request
ATM	Asynchronous Transfer Mode
BER	Bit Error Rate
BPDU	Beacon PDU
BPSK	Binary Phase Shift Keying
CFP	Contention-Free Period
CID	Connection Identifier

ClMTUSize	Convergence layer Maximum Transmit Unit Size
CL	Convergence Layer
CPCS	Common Part Convergence Sublayer
CRC	Cyclic Redundancy Check
CSMA-CA	Carrier Sense Multiple Access-Collision Avoidance
D8PSK	Differential Eight-Phase Shift Keying
DBPSK	Differential Binary Phase Shift Keying
DHCP	Dynamic Host Configuration Protocol
DPSK	Differential Phase Shift Keying (general)
DQPSK	Differential Quaternary Phase Shift Keying
DSK	Device Secret Key
ECB	Electronic Code Book
EMA	Exponential Moving Average
ENOB	Effective Number Of Bits
EUI-48	48-bit Extended Unique Identifier
EVM	Error Vector Magnitude
FCS	Frame Check Sequence
FEC	Forward Error Correction
FFT	Fast Fourier Transform
GK	Generation Key
GPDU	Generic MAC PDU
HCS	Header Check Sum
IFFT	Inverse Fast Fourier Transform
IGMP	Internet Group Management Protocol
IPv4	Internet Protocol version 4
KDIV	Key Diversifier
LCID	Local Connection Identifier
LFSR	Linear Feedback Shift Register
LLC	Logical Link Control
LNID	Local Node Identifier
LSID	Local Switch Identifier
LWK	Local Working Key
MAC	Medium Access Control
MK	Master Key
MLME	MAC Layer Management Entity
MPDU	MAC Protocol Data Unit
MSDU	MAC Service Data Unit

MTU	Maximum Transmission Unit
MSPS	Million Samples Per Second
NAT	Network Address Translation
NID	Node Identifier
NSK	Network Secret Key
OFDM	Orthogonal Frequency Division Multiplexing
PDU	Protocol Data Unit
PHY	Physical Layer
PIB	PLC Information Base
PLC	Power Line Communications
PLME	PHY Layer Management Entity
PNPDU	Promotion Needed PDU
PPDU	PHY Protocol Data Unit
ppm	Parts per million
PSD	Power Spectral Density
PSDU	PHY Service Data Unit
QoS	Quality of Service
SAP	Service Access Point
SAR	Segmentation And Reassembly
SCP	Shared-Contention Period
SCRC	Secure CRC
SDU	Service Data Unit
SEC	Security
SID	Switch Identifier
SNA	Subnetwork Address
SNK	Subnetwork Key (corresponds to either REG.SNK or SEC.SNK)
SNR	Signal to Noise Ratio
SP	Security Profile
SSCS	Service Specific Convergence Sublayer
SWK	Subnetwork Working Key
TCP	Transmission Control Protocol
TOS	Type Of Service
UI	Unique Identifier
USK	Unique Secret Key
VJ	Van Jacobson
WK	Working Key

B.2 General description

B.2.1 Introduction

This document is the Specification for a solution for PLC in the CENELEC A-Band using OFDM modulation scheme.

B.2.2 General description of the architecture

Figure B.1 below depicts the communication layers and the scope of this specification. This specification focuses mainly on the data, control and management plane.

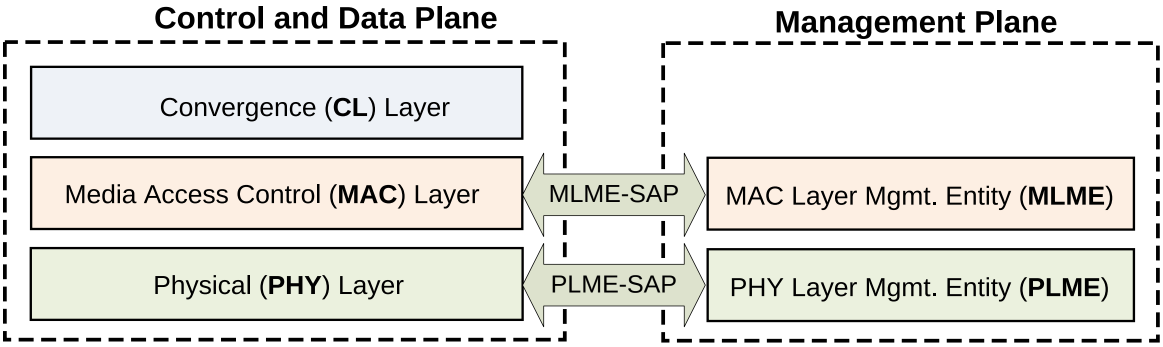


Figure B.1 – Reference model of protocol layers used in the OFDM TYPE 1 specification

The CL classifies traffic associating it with its proper MAC connection; this layer performs the mapping of any kind of traffic to be properly included in MSDUs. It may also include header compression functions. Several SSCSs are defined to accommodate different kinds of traffic into MSDUs.

The MAC layer provides core MAC functionalities of system access, bandwidth allocation, connection establishment/maintenance and topology resolution.

The PHY layer transmits and receives MPDUs between Neighbour Nodes using orthogonal frequency division multiplexing (OFDM). OFDM is chosen as the modulation technique because of:

- its inherent adaptability in the presence of frequency selective channels (which are common but unpredictable, due to narrowband interference or unintentional jamming);
- its robustness to impulsive noise, resulting from the extended symbol duration and use of FEC;
- its capacity for achieving high spectral efficiencies with simple transceiver implementations.

The PHY specification, described in Chapter B.3, also employs a flexible coding scheme. The PHY data rates can be adapted to channel and noise conditions by the MAC.

B.3 Physical layer

The physical layer is specified in [ITU-T G.9955]/Annex B.

B.4 MAC layer

B.4.1 Overview

A Subnetwork can be logically seen as a tree structure with two types of Nodes: the Base Node and Service Nodes.

- **Base Node**: It is at the root of the tree structure and it acts as a master Node that provides all Subnetwork elements with connectivity. It manages the Subnetwork resources and connections. There is only one Base Node in a Subnetwork. The Base Node is initially the Subnetwork itself, and any other Node should follow a Registration process to enrol itself on the Subnetwork.
- **Service Node**: They are either leaves or branch points of the tree structure. They are initially in a Disconnected functional state and follow the Registration process in B.4.6.1 to become part of the Subnetwork. Service Nodes have two functions in the Subnetwork: keeping connectivity to the Subnetwork for their Application layers, and switching other Nodes' data to propagate connectivity.

Devices elements that exhibit Base Node functionality continue to do so as long as they are not explicitly reconfigured by mechanisms that are beyond the scope of this specification. Service Nodes, on the other hand, change their behaviour dynamically from "Terminal" functions to "Switch" functions and vice-versa. The changing of functional states occurs in response to certain pre-defined events on the network. Figure B.2 shows the functional state transition diagram of a Service Node.

The three functional states of a Service Node are ***Disconnected***, ***Terminal*** and ***Switch***:

- **Disconnected**: This is the initial functional state for all Service Nodes. When Disconnected, a Service Node is not able to communicate data or switch other Nodes' data; its main function is to search for a Subnetwork within its reach and try to register on it.
- **Terminal**: When in this functional state a Service Node is able to establish connections and communicate data, but it is not able to switch other Nodes' data.
- **Switch**: When in this functional state a Service Node is able to perform all Terminal functions. Additionally, it is able to forward data to and from other Nodes in the same Subnetwork. It is a branch point on the tree structure.

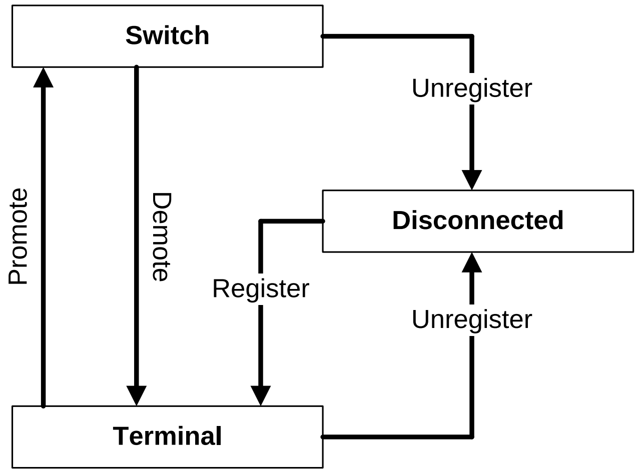


Figure B.2 – Service Node states

The events and associated processes that trigger changes from one functional state to another are:

- **Registration**: the process by which a Service Node includes itself in the Base Node's list of registered Nodes. Its successful completion means that the Service Node is part of a Subnetwork. Thus, it represents the transition between Disconnected and Terminal.
- **Unregistration**: the process by which a Service Node removes itself from the Base Node's list of registered Nodes. Unregistration may be initiated by either of Service Node or Base Node. A Service Node may unregister itself to find a better point of attachment, i.e., change Switch Node through which it is attached to the network. A Base Node may unregister a

registered Service Node as a result of failure of any of the MAC procedures. Its successful completion means that the Service Node is Disconnected and no longer part of a Subnetwork;

- **Promotion**: the process by which a Service Node is qualified to switch (repeat, forward) data traffic from other Nodes and act as a branch point on the Subnetwork tree structure. A successful promotion represents the transition between Terminal and Switch. When a Service Node is Disconnected it cannot directly transition to Switch;
- **Demotion**: the process by which a Service Node ceases to be a branch point on the Subnetwork tree structure. A successful demotion represents the transition between Switch and Terminal.

B.4.2 Addressing

B.4.2.1 General

Each Node has a 48-bit universal MAC address, defined in [IEEE 802-2001] and called EUI-48. Every EUI-48 is assigned during the manufacturing process and it is used to uniquely identify a Node during the Registration process.

The EUI-48 of the Base Node uniquely identifies its Subnetwork. This EUI-48 is called the Subnetwork Address (SNA).

The Switch Identifier (LSID) is a unique 8-bit identifier for each Switch Node inside a Subnetwork. The Subnetwork Base Node assigns an LSID during the promotion process. A Switch Node is universally identified by the SNA and LSID. LSID = 0x00 is reserved for the Base Node. LSID = 0xFF is reserved to mean "unassigned" or "invalid" in certain specific fields (see Table B.13).). This special use of the 0xFF value is always made explicit when describing those fields and it shall not be used in any other field.

During its Registration process, every Service Node receives a 14-bit Local Node Identifier (LNID). The LNID identifies a single Service Node among all Service Nodes that directly depend on a given Switch. The combination of a Service Node's LNID and SID (its immediate Switch's LSID) forms a 22-bit Node Identifier (NID). The NID identifies a single Service Node in a given Subnetwork. LNID = 0x0000 cannot be assigned to a Terminal, as it refers to its immediate Switch. LNID = 0x3FFF is reserved for broadcast and multicast traffic (see clause B.4.2.3 for more information). In certain specific fields, the LNID = 0x3FFF may also be used as "unassigned" or "invalid" (see Table B.1 and Table B.9). This special use of the 0x3FFF value is always made explicit when describing the said fields and it shall not be used in this way in any other field.

During connection establishment a 9-bit Local Connection Identifier (LCID) is reserved. The LCID identifies a single connection in a Node. The combination of NID and LCID forms a 31-bit Connection Identifier (CID). The CID identifies a single connection in a given Subnetwork. Any connection is universally identified by the SNA and CID. LCID values are allocated with the following rules:

LCID=0x000 to 0x0FF, for connections requested by the Base Node. The allocation shall be made by the Base Node.

LCID=0x100 to 0x1FF, for connections requested by a Service Node. The allocation shall be made by a Service Node.

The full addressing structure and field lengths are shown in Figure B.3.

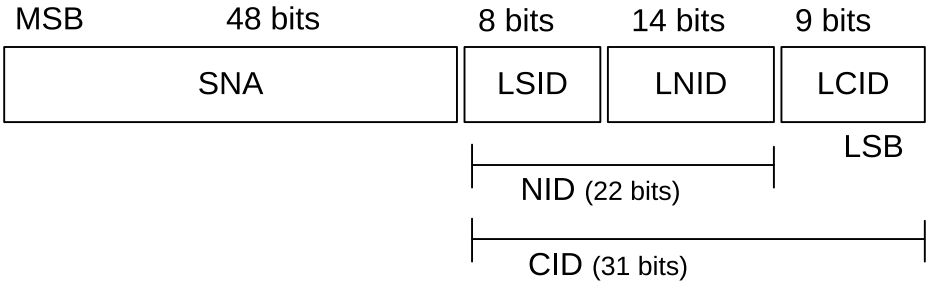


Figure B.3 – Addressing Structure

When a Service Node in *Terminal* state starts promotion process, the Base Node allocates a unique switch identifier which is used by this device after transition to switch state as SID of this switch. The promoted Service Node continues to use the same NID that it used before promotion, i.e., it maintains SID of its next level switch for addressing all traffic generated/destined to its local application processes. To maintain distinction between the two switch identifiers, the switch identifier allocated to a Service Node during its promotion is referred to as Local Switch Identifier (LSID). Note that the LSID of a switch device will be SID of devices that connects to the subnetwork through it.

Each Service Node has a level in the topology tree structure. Service Nodes which are directly connected to the Base Node have level 0. The level of any Service Node not directly connected to the Base Node is the level of its immediate Switch plus one.

B.4.2.2 Example of address resolution

Figure B.4 shows an example where Disconnected Service Nodes are trying to register on the Base Node. In this example, addressing will have the following nomenclature: (SID, LNID). Initially, the only Node with an address is Base Node A, which has an NID=(0, 0).

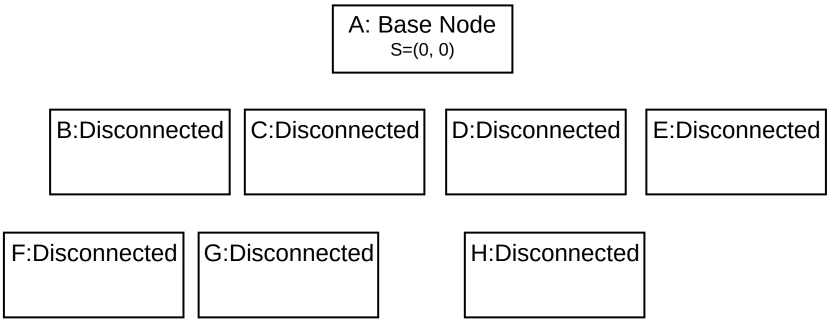


Figure B.4 – Example of address resolution: phase 1

Every other Node of the Subnetwork will try to register on the Base Node. Only B, C, D and E Nodes are able to register on this Subnetwork and get their NIDs. Figure B.5 shows the status of Nodes after the Registration process. Since they have registered on the Base Node, they get the SID of the Base Node and a unique LNID. The level of newly registered Nodes is 0 because they are connected directly to the Base Node.

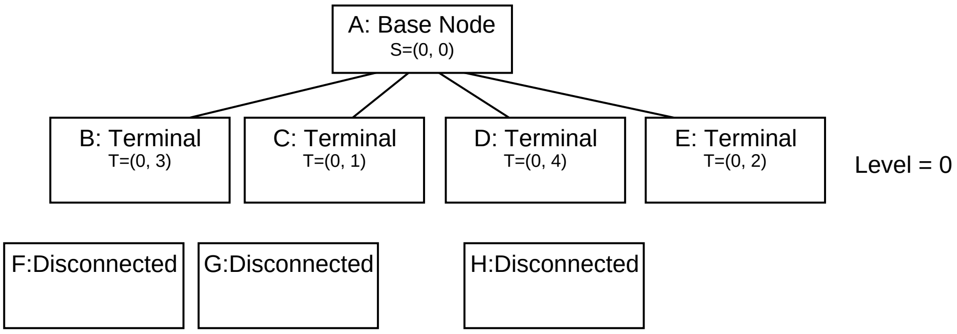


Figure B.5 – Example of address resolution: phase 2

Nodes F, G and H cannot connect directly to the Base Node, which is currently the only Switch in the Subnetwork. F, G and H will send PNPDU broadcast requests, which will result in Nodes B and D requesting promotion for themselves in order to extend the Subnetwork range. During promotion, they will both be assigned unique SIDs. Figure B.6 shows the new status of the network after the promotion of Nodes B and D. Each Switch Node will still use the NID that was assigned to it during the Registration process for its own communication as a Terminal Node. The new SID shall be used for all switching functions.

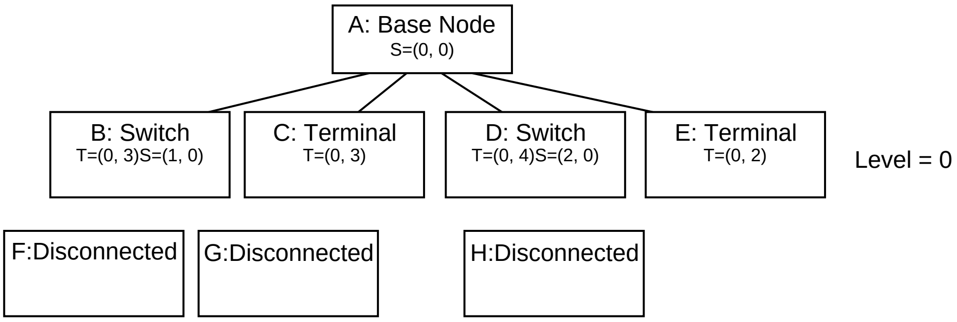


Figure B.6 – Example of address resolution: phase 3

On completion of the B and D promotion process, Nodes F, G and H shall start their Registration process and have a unique LNID assigned. Every Node on the Subnetwork will then have a unique NID to communicate like a Terminal, and Switch Nodes will have unique SIDs for switching purposes. The level of newly registered Nodes is 1 because they register with level 0 Nodes. On the completion of topology resolution and address allocation, the example Subnetwork would be as shown in Figure B.7.

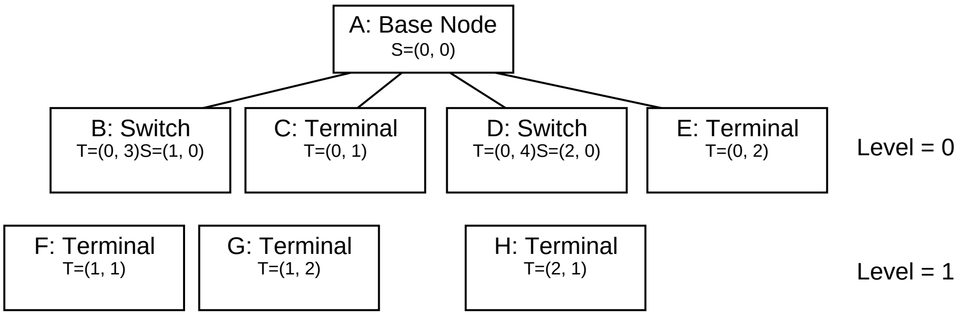


Figure B.7 – Example of address resolution: phase 4

B.4.2.3 Broadcast and multicast addressing

Multicast and broadcast addresses are used for communicating data to multiple Nodes. There are several broadcast and multicast address types, depending on the context associated with the traffic flow. Table B.1 describes different broadcast and multicast addressing types and the SID and LNID fields associated with each one.

Table B.1 – Broadcast and multicast address

Type	LNID	Description
Broadcast	0x3FFF	Using this address as a destination, the packets should reach every Node of the Subnetwork.
Multicast	0x3FFE	This type of address refers to multicast groups. The multicast group is defined by the LCID.
Unicast	not 0x3FFF not 0x3FFE	The address of this type refers to the only Node of the Subnetwork whose SID and LNID match the address fields.

B.4.3 MAC functional description

B.4.3.1 Service node start-up

A Service Node is initially Disconnected. The only functions that may be performed in a *Disconnected* functional state are: reception of any beacons on the channel and sending of the PNPDUs. Each Service Node shall maintain a Switch table that is updated with the reception of a beacon from any new Switch Node. Based on local implementation policies, a Service Node may select any Switch Node from the Switch table and proceed with the Registration process with that Switch Node. The criterion for selecting a Switch Node from the Switch table is beyond the scope of this specification.

A Service Node shall listen on the channel for at least *macMinSwitchSearchTime* before deciding that no beacon is being received. It may optionally add some random variation to *macMinSwitchSearchTime,* but this variation cannot be more than 10% of *macMinSwitchSearchTime.* If no beacons are received in this time, the Service Node shall broadcast a PNPDU. The PNPDU shall be broadcast with the most robust modulation scheme to ensure maximum coverage. A Service Node seeking promotion of any of the Terminal Nodes in its proximity shall not transmit more than *macMaxPromotionPdu* PNPDUs per *macPromotionPduTxPeriod* units of time. The Service Nodes shall also ensure that the broadcast of PNPDUs is randomly spaced. There must always be a random time separation between successive broadcasts.

So as not to flood the network with PNPDUs, especially in cases where several devices are powered up at the same time, the Terminal Nodes shall reduce the PNPDU transmission rate by a factor of PNPDUs received from other sources. For example, if a Node receives one PNPDU when it is transmitting its own PNPDUs, it shall reduce it**s** own transmissions to no more than *macMaxPromotionPdu/2* per *macPromotionPduTxPeriod* units of time. Likewise, if it receives PNPDUs from two different sources, it shall slow down its rate to no more than *macMaxPromotionPdu/3* per *macPromotionPduTxPeriod* units of time.

On the selection of a specific Switch Node, a Service Node shall start a Registration process by transmitting the REG control packet (B.4.4.5.3) to the Base Node. The Switch Node through which the Service Node intends to carry out its communication is indicated in the REG control packet.

B.4.3.2 Starting and maintaining subnetworks

Base Nodes are primarily responsible for setting up and maintaining a Subnetwork. In order to execute the latter, the Base Node shall perform the following:

- **Beacon transmission**: The Base Node and all the Switch Nodes on the Subnetwork shall broadcast beacons at fixed intervals of time. The Base Node shall always transmit exactly one beacon per frame. Switch Nodes shall transmit beacons with a frequency prescribed by the Base Node at the time of their promotion.
- **Promotion and demotion of Terminals and switches**: All promotion requests generated by Terminal Nodes upon reception of PNPDUs are directed to the Base Node. The Base Node maintains a table of all the Switch Nodes on the Subnetwork and allocates a unique SID to new incoming requests. Upon reception of multiple promotion requests, the Base Node can, at its own discretion, reject some of the requests. Likewise, the Base Node is responsible for demoting registered Switch Nodes. The demotion may either be initiated by the Base Node (based on an implementation-dependent decision process) or be requested by the Switch Node itself.
- **Registration management**: The Base Node receives Registration requests from all new Nodes trying to be part of the Subnetwork it manages. The Base Node shall process each Registration request it receives and respond with an accept or reject message. When the Base Node accepts the registration of a service node, it shall allocate an unique NID to it to be used for all subsequent communication on the Subnetwork. Likewise, the Base Node is responsible for deregistering any registered Service Node. The unregistration may be initiated by the Base Node (based on an implementation-dependent decision process) or requested by the Service Node itself.
- **Connection setup and management**: The MAC layer specified in this document is connection-oriented, implying that data exchange is necessarily preceded by connection establishment. The Base Node is always required for all connections on the Subnetwork, either as an end point of the connection or as a facilitator (direct connections; clause B.4.3.6) of the connection.
- **Channel access arbitration**: The usage of the channel by devices conforming to this specification may be controlled and contention-free at certain times and open and contention-based at others. The Base Node prescribes which usage mechanism shall be in force at what time and for how long. Furthermore, the Base Node shall be responsible for assigning the channel to specific devices during contention-free access periods.
- **Distribution of random sequence for deriving encryption keys**: When using Security Profile 1 (see 4.3.8.2), all control messages in this MAC specification shall be encrypted before transmission. Besides control messages, data transfers may be optionally encrypted as well. The encryption key is derived from a 128-bit random sequence. The Base Node shall periodically generate a new random sequence and distribute it to the entire Subnetwork, thus helping to maintain the Subnetwork security infrastructure.
- **Multicast group management**: The Base Node shall maintain all multicast groups on the Subnetwork. This shall require the processing of all join and leave requests from any of the Service Nodes and the creation of unsolicited join and leave messages from Base Node application requests.

Additional information regarding promotion and connection procedures can be found in clauses B.4.6.3 and B.4.6.6.

B.4.3.3 Channel access

B.4.3.3.1 General

Devices on a Subnetwork access the channel based on specific guidelines laid down in this clause. Time is divided into composite units of abstraction for channel usage, called MAC Frames. The Service Nodes and Base Node on a Subnetwork can access the channel in the Shared Contention Period (SCP) or request a dedicated Contention-Free Period (CFP).

CFP channel access needs devices to request allocation from the Base Node. Depending on channel usage status, the Base Node may grant access to the requesting device for specific duration or deny the request.

SCP channel access does not require any arbitration. However, the transmitting devices need to respect the SCP timing boundaries in a MAC Frame. The composition of a MAC Frame in terms of SCP and CFP is communicated in every frame as part of beacon.

A MAC Frame is comprised of one or more Beacons, one Shared-Contention Period and zero or one Contention-Free Period (CFP). When present, the length of the CFP is indicated in the BPDU.

Beacon 0	Beacon 1	Beacon 2	Beacon 3	Beacon 4	SCP	CFP

Figure B.8 – Structure of a MAC Frame

B.4.3.3.2 Beacon

B.4.3.3.2.1 General

A BPDU is transmitted by the Base Node every (*MACFrameLength – MACBeaconLength)* symbols. The Switch Nodes also transmit BPDU to maintain their part of the subnetwork. They transmit BPDUs at regular times, but the transmission frequency does not need to be the same as that of the Base Node, i.e., a Switch Node may not transmit its BPDU in every frame.

A beacon is always *MACBeaconLength* symbols long. This length is the beacon duration excluding the PHY PREAMBLE overhead. Since the BPDU is to be received by all devices in the originating Switch domain, it is transmitted with the most robust PHY modulation scheme and FEC coding at the maximum output power level implemented in the device. Details of the BPDU structure and contents are given in B.4.4.4.

All Service Nodes shall track beacons as explained in B.4.3.4.1.

B.4.3.3.2.2 Beacon slots

A single frame may contain *macBeaconsPerFrame* BPDUs. The unit of time in which a BPDU is transmitted, is referred to as a beacon-slot. All beacon-slots are located at the beginning of a frame, as shown in Figure B.8 above. The first beacon-slot in every frame is reserved for the Base Node. The number of beacon-slots in a frame may change from one frame to another and is indicated by the Base Node in its BPDU.

The Switch Nodes are allocated a beacon-slot at the time of their promotion. Following the PRO control packet, the Base Node transmits the BSI control packet that would list specific details on which beacon-slot should be used by the new Switch device.

The number of beacon-slots in a frame should be increased from 1 to at least 2 on the promotion of the first Switch device on a subnetwork by the Base Node. Similarly, a Base Node cannot decrease the number of beacon-slots in the subnetwork to 1 when there is a Switch Node on its subnetwork.

With the Registration of each new Switch on the subnetwork, the Base Node may change the beacon-slot or BPDU transmission frequency (or both) of already registered Switch devices. When such a change occurs, the Base Node transmits a Beacon Slot Information (BSI) control packet to each individual Switch device that is affected. The Switch device addressed in the BSI packet sends an acknowledgement back to the Base Node. Switch devices are required to relay the BSI control packet that is addressed to Switch devices connected through them. During the reorganization of beacon-slots, if there is a change in the beacon-slot count per frame, the Base Node should transmit an FRA (FRAme) control packet to the entire Subnetwork. The FRA control packet is an indication of change in the overall Frame structure. In this specific case, it would imply an increase in SCP slots and a decrease in the number of Beacon Slots.

Switch devices that receive an FRA control packet should relay it to their entire control domain because FRA packets are broadcast information about changes to frame structures.

This is required for the entire Subnetwork to have a common understanding of frame structure, especially in regions where the controlling Switch devices transmit BPDUs at frequencies below once per frame.

Figure B.9 below shows a sample beacon-slot change sequence for an existing Switch device. The example shows a beacon-slot change triggered by the promotion of a Terminal device (PRO_ACK). In this case, the promotion is followed by a change in both the number of beacon-slots per frame and of the specific beacon-slot parameters already allocated to a Switch.

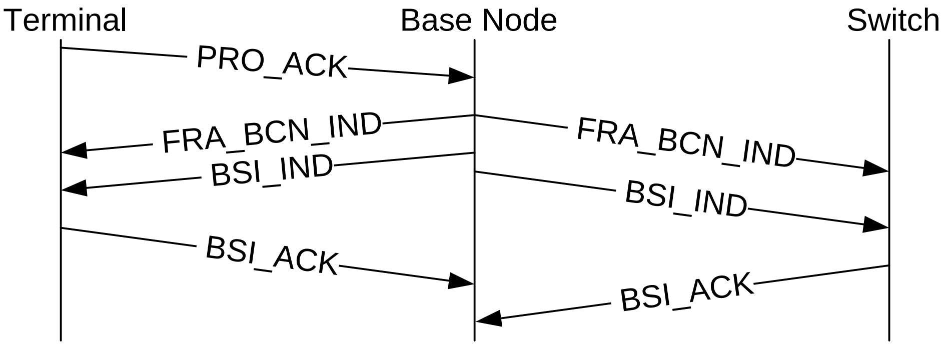


Figure B.9 – Example of control packet sequencing following a promotion

B.4.3.3.2.3 Beacon-slot allocation policy

The Beacon Slot allocation policy shall ensure that during promotion a Service Node never receives a BSI control packet that enforces it to transmit a beacon consecutive to every beacon of the Node it is registered to.

This behaviour shall be ensured if the BSI information follows one, or more, of the following rules (BCN represents the information of the beacons the Service Node is registered to):

- BSI.SLT is not consecutive to BCN.POS;
- BSI.SEQ is not equal to any BCN.SEQ in a superframe;
- BSI.FRQ is greater than BCN.FRQ.

B.4.3.3.2.4 Beacon superframes

When changing the frame structure, to add or remove Beacon Slots, or to change which Beacon Slot a Switch should use, it is necessary to indicate when such a change should occur. All Nodes must change at the same time otherwise there will be collisions with the beacons, etc.

To solve this problem a Beacon superframe is defined. Each Beacon contains a 5 bit sequence number. Thus 32 frames form a superframe. Any messages which contain changes to the structure or usage of the frame include a sequence number for when the change should occur. The changes requested should only happen when the beacon sequence number matches the sequence number in the change request.

B.4.3.3.3 Shared-contention period

B.4.3.3.3.1 General

Shared-contention period (SCP) is the time when any of the devices on the Subnetwork can transmit data. The SCP starts immediately after the end of the beacon-slot(s) in a frame. Collisions resulting from simultaneous attempt to access the channel are avoided by the CSMA-CA mechanism specified in this clause.

The length of the SCP may change from one frame to another and is indicated by information in the Beacon. At all times, the SCP is at least *MACMinSCPLength* symbols long. The maximum permissible length of an SCP in a frame is (*MACFrameLength* – *MACBeaconLength*) symbols. Maximum length SCPs can only occur when there are no dedicated channel access grants to any of the devices (no CFP) on a Subnetwork that has no Switch Nodes (only one Beacon Slot).

The use of SCP is not restricted to frames in which beacons are received. In lower Levels of the Subnetwork, the controlling Switch Node may transmit beacons at a much lower frequency than once per frame. For these parts of the Subnetwork, the frame structure would still continue to be the same in frames where no beacons are transmitted. Thus, the Service Nodes in that segment may still use SCP at their discretion.

B.4.3.3.3.2 CSMA-CA algorithm

The CSMA-CA algorithm implemented in devices works as shown in the Figure B.10.

Implementations start with a random backoff time (*macSCPRBO*) based on the priority of data queued for transmission. *MACPriorityLevels* levels of priority need to be defined in each implementation, with a lower value indicating higher priority. In the case of data aggregation, the priority of aggregate bulk is governed by the highest priority data it contains. The *MacSCPRBO* for a transmission attempt is give as below:

$$\text{macSCPRBO} = \text{random}\ (0, \text{MIN}\ ((2^{(\text{Priority+txAttempts})} +1), (\text{macSCPLength}/2)))$$

Before a backoff period starts, a device should ensure that the remaining SCP time is long enough to accommodate the backoff, the number of iterations for channel-sensing (based on data priority) and the subsequent data transmission. If this is not the case, backoff should be aborted till the SCP starts in the next frame. Aborted backoffs that start in a subsequent frame should not carry *macSCPRBO* values of earlier attempts. *macSCPRBO* values should be regenerated on the resumption of the transmission attempt in the SCP time of the next frame.

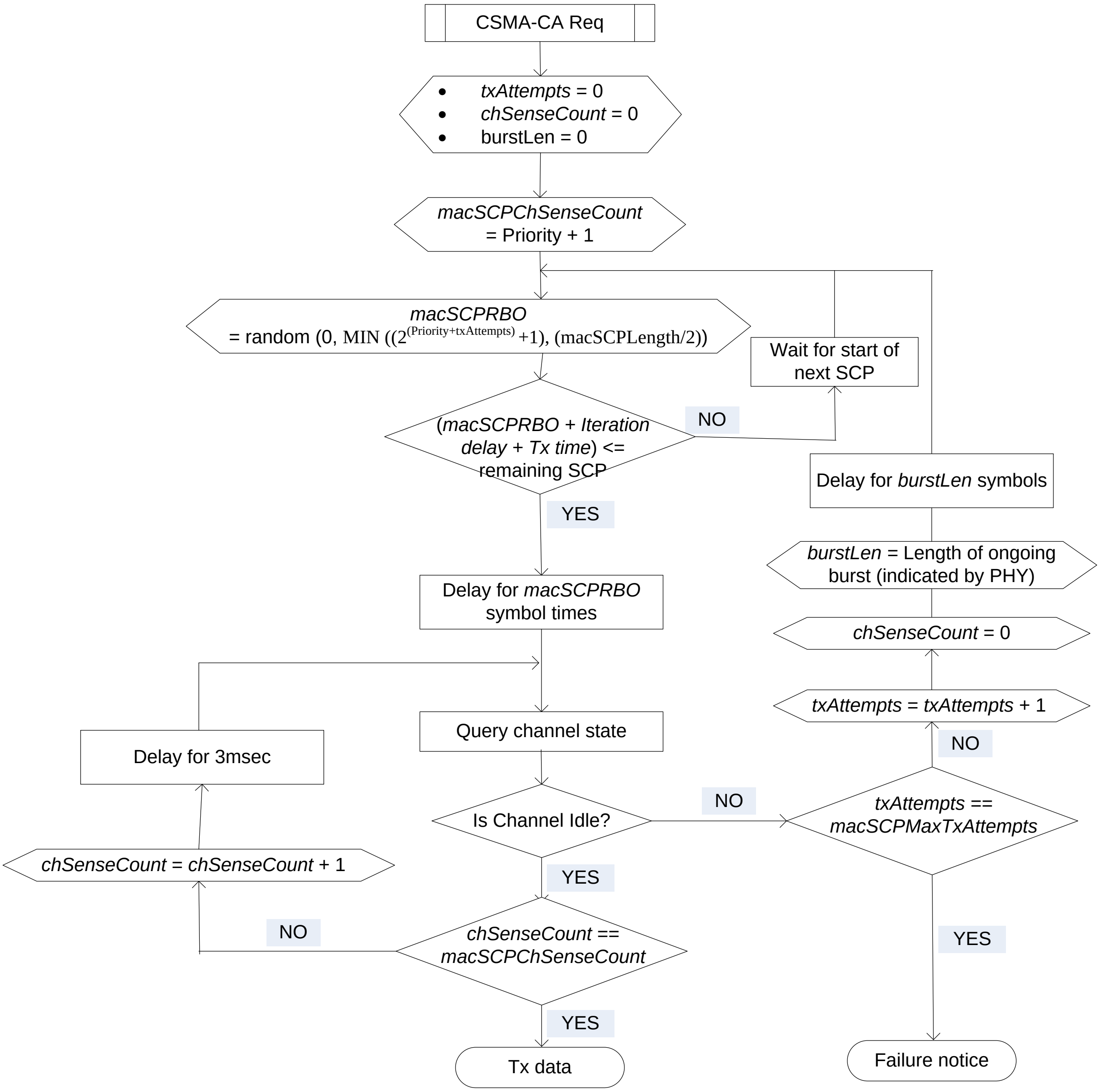

CSMA-CA Req
txAttempts = 0
chSenseCount = 0
burstLen = 0
macSCPChSenseCount
= Priority + 1
macSCPRBO
= random (0, MIN ((2(Priority+txAttempts) +1), (macSCPLength/2))
Wait for start of next SCP
(macSCPRBO + Iteration delay + Tx time) <= remaining SCP
NO
YES
Delay for burstLen symbols
burstLen = Length of ongoing burst (indicated by PHY)
chSenseCount = 0
txAttempts = txAttempts + 1
NO
Delay for macSCPRBO symbol times
Query channel state
Delay for 3msec
Is Channel Idle?
NO
txAttempts == macSCPMaxTxAttempts
chSenseCount = chSenseCount + 1
YES
NO
chSenseCount == macSCPChSenseCount
YES
YES
Tx data
Failure notice

Figure B.10 – Flow chart for CSMA-CA algorithm

On the completion of *macSCPRBO* symbol time, implementations perform channel-sensing. Channel sensing shall be performed one or more times depending on priority of data to be transmit. The number of times for which an implementation has to perform channel-sensing (*macSCPChSenseCount*) is defined by the priority of the data to be transmitted with the following relation:

$$macSCPChSenseCount = Priority + 1$$

and each channel sense should be separated by a 3 ms delay.

When a channel is sensed to be idle on all *macSCPChSenseCount* occasions, an implementation may conclude that the channel status is idle and carry out its transmission immediately.

During any of the *macSCPChSenseCount* channel-sensing iterations, if the channel is sensed to be occupied, implementations should reset all working variables. The local counter tracking the number of times a channel is found to be busy should be incremented by one and the CSMA-CA process should restart by generating a new *macSCPRBO*. The remaining steps, starting with the backoff, should follow as above.

If the CSMA-CA algorithm restarts *macSCPMaxTxAttempts* number of times due to ongoing transmissions from other devices on the channel, the transmission shall abort by informing the upper layers of CSMA-CA failure.

B.4.3.3.3.3 MAC control packets

MAC control packets should be transmitted in the SCP with a priority of one. Refers to priorities in clause B.4.4.2.3.

B.4.3.3.4 Contention-free period

Each MAC frame may optionally have a contention-free period where devices are allocated channel time on an explicit request to do so. If no device on a Subnetwork requests contention-free channel access, the CFP_ALC_REQ_S may be entirely absent and the MAC frame would comprise only SCP. All CFP_ALC_REQ_S requests coming from Terminal or Switch Nodes are addressed to the Base Node. Intermediate Switch Nodes along the transmission path merely act on the allocation decision by the Base Node. A single MAC frame may contain up to *MACCFPMaxAlloc* non-overlapping contention-free periods.

Base Nodes may allocate overlapping times to multiple requesting Service Nodes. Such allocations may lead to potential interference. Thus, a Base Node should make such allocations only when devices that are allocated channel access for concurrent usage are sufficiently separated.

Service Nodes make channel allocation request in a CFP MAC control packet. The Base Node acts on this request and responds with a request acceptance or denial. In the case of request acceptance, the Base Node shall respond with the location of allocation time within MAC frame, the length of allocation time and number of future MAC frames from which the allocation pattern will take effect. The allocation pattern remains effective unless there is an unsolicited location change of the allocation period from the Base Node (as a result of a channel allocation pattern reorganization) or the requesting Service Node sends an explicit de-allocation request using a CFP MAC control packet.

Changes resulting from action taken on a CFP MAC control message that impact overall MAC frame structure are broadcast to all devices using an FRA MAC control message.

In a multi-level Subnetwork, when a Service Node that is not directly connected to the Base Node makes a request for CFP, the Base Node shall allocate CFPs to all the intermediate Switch Nodes so that the entire transit path from the source Service Node to Base has contention-free time-slots reserved. The Base Node shall transmit multiple CFP control packets. The first of these CFP_ALC_IND will be for the requesting Service Node. Each of the rest will be addressed to an intermediate Switch Node.

B.4.3.4 Tracking switches and peers

B.4.3.4.1 Tracking switches

Service Nodes should keep track of all neighbouring Switch Nodes by maintaining a list of the beacons received. Such tracking shall keep a Node updated on reception signal quality from Switch Nodes other than the one to which it is connected, thus making it possible to change connection points (Switch Node) to the Subnetwork if link quality to the existing point of connectivity degrades beyond an acceptable level.

Note that such a change of point of connectivity may be complex for Switch Nodes because of devices connected through them. However, at certain times, network dynamics may justify a complex reorganization rather than continue with existing limiting conditions.

B.4.3.4.2 Tracking disconnected nodes

Terminals shall process all received PNPDUs. When a Service Node is Disconnected, it does not have information on current MAC frame structure so the PNPDUs may not necessarily arrive during the SCP. Thus, Terminals shall also keep track of PNPDUs during the CFP or beacon-slots.

On processing a received PNPDU, a Terminal Node may decide to ignore it and not generate any corresponding promotion request (PRO_REQ_S). A Terminal Node shall ignore no more than *MACMaxPRNIgnore* PNPDUs from the same device. Receiving multiple PNPDUs from the same device indicates that there is no other device in the vicinity of the *Disconnected* Node, implying that there will be no possibility of this new device connecting to any Subnetwork if the Terminal Node does not request promotion for itself.

B.4.3.5 Switching

B.4.3.5.1 General

On a Subnetwork, the Base Node cannot communicate with every Node directly. Switch Nodes relay traffic to/from the Base Node so that every Node on the Subnetwork is effectively able to communicate with the Base Node. Switch Nodes selectively forward traffic that originates from or is destined to one of the Service Nodes in its control hierarchy. All other traffic is discarded by Switches, thus reducing traffic flow on the network.

Different names of MAC header and packets are used in this clause. Please refer to the clause B.4.4.2 to find their complete specification.

B.4.3.5.2 Switching table

Each Switch Node maintains a table of other Switch Nodes that are connected to the Subnetwork through it. Maintaining this information is sufficient for switching because traffic to/from Terminal Nodes will also contain the identity of their respective Switch Nodes (PKT.SID). Thus, the switching function is simplified in that maintaining an exhaustive listing of all Terminal Nodes connected through it is not necessary.

Switch Nodes start with no entries in their switching table. The switching table is dynamically updated by keeping track of promotion and demotion control packets flowing on the network. A new entry is created for every promotion acknowledgement (PRO_ACK) that has a PKT.SID matching either the SID of the Switch Node itself or any of the existing entries in the switching table. Likewise, an entry corresponding to a PRO.NSID field is deleted when a demotion request is acknowledged (PRO_DEM_x).

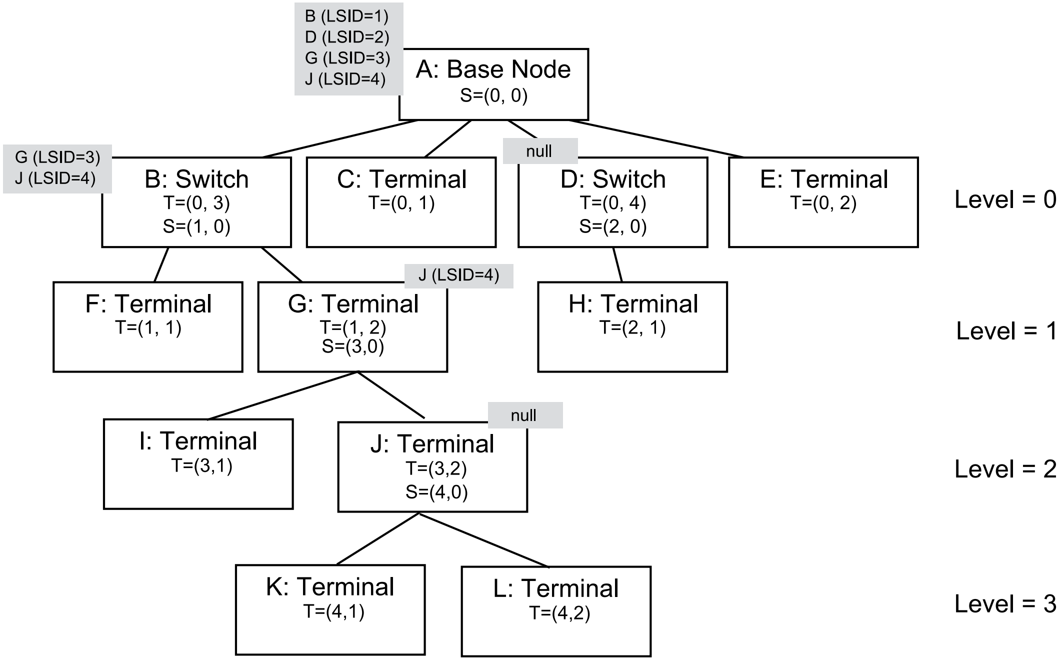


Figure B.11 – Switching tables example

Figure B.11 shows an example Subnetwork where entries in the switching table of individual Switch Nodes are highlighted. In this example, when Service Node G receives a PRO_REQ_B packet for promotion, it turns into a Switch Node. Its Switch identifier will be (PRO.NSID, 0) = (3, 0). The receipt and acceptance of PRO_REQ_B is acknowledged with a PRO_ACK by G. The intermediate Switch Node B will sniff HDR.DO=0, PKT.CTYPE=3, PKT.SID=1 and PRO.N=0, to conclude that this is a PRO_ACK from one of the Service Nodes in its own switching hierarchy. Node B will forward this packet towards the Base Node and it will add PRO.NSID to its switching table, as shown in Figure B.12.

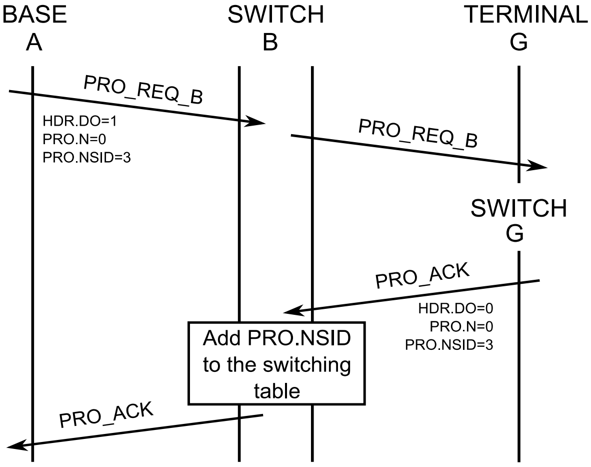


Figure B.12 – Fill in the switching table

Removing a Switch table entry is more complex because of retries. On reception of a demotion acknowledgement (PRO_DEM_x), the switching table entry corresponding to the LSID is marked as to be removed and a timer is started with a value of ((*macMaxCtlReTx* + 1) * *macCtlReTxTimer*) seconds. This timer ensures that all retransmit packets which might use the LSID have left the Subnetwork. When the timer expires the Switch table entry is removed

B.4.3.5.3 Switching process

Switch Nodes forward traffic to their control domain in a selective manner. The received data shall fulfil the conditions listed below for it to be switched. If the conditions are not met, the data shall be silently discarded.

Downlink packets (HDR.DO=1) shall meet any of the following conditions in order to be switched:

- Destination Node of the packet is connected to the Subnetwork through this Switch Node, i.e., PKT.SID is equal to this Switch Node's SID or its switching table contains an entry for PKT.SID.
- The packet has broadcast destination (PKT.LNID = 0x3FFF) and was sent by the Switch this Node is registered through (PKT.SID=SID of this Switch Node).
- The packet has a multicast destination (PKT.LNID=0x3FFE), it was sent by the Switch this Node is registered through (PKT.SID=SID of this Switch Node) and at least one of the Service Nodes connected to the Subnetwork through this Switch Node is a member of the said multicast group, i.e., LCID specifies a group that is requested by any downstream Node in its hierarchy.

Uplink packets (HDR.DO=0) shall meet either of the following conditions in order to be switched:

- The packet source Node is connected to the Subnetwork through this Switch Node, i.e., PKT.SID is equal to this Switch Node's SID or its switching table contains an entry for PKT.SID.
- The packet has a broadcast or multicast destination (PKT.LNID = 0x3FFF or 0x3FFE) and was transmitted by a Node registered through this Switch Node (PKT.SID=LSID of this Switch Node).

If a packet meets previous conditions, it shall be switched. For unicast packets, the only operation to be performed during switching is queuing it to be resent in a MAC PDU with the same HDR.DO.

In case of broadcast or multicast packets, the PKT.SID must be replaced with:

- The Switch Node's LSID for Downlink packets.
- The Switch Node's SID for uplink packets.

B.4.3.5.4 Switching of broadcast packets

The switching of broadcast MAC frames operates in a different manner to the switching of unicast MAC frames. Broadcast MAC frames are identified by PKT.LNID=0x3FFF.

When HDR.DO=0, i.e., the packet is an uplink packet, it is unicast to the Base Node. A Switch which receives such a packet should apply the scope rules to ensure that it comes from a lower level and, if so, Switch it upwards towards the base. The rules given in clause B.4.3.5.3 must be applied.

When HDR.DO=1, i.e., the packet is a Downlink packet, it is broadcast to the next level. A Switch which receives such a packet should apply the scope rules to ensure that it comes from the higher level and, if so, switch it further to its Subnetwork. The most robust PHY modulation scheme and FEC coding at the maximum output power level implemented in the device should be used so that all the devices directly connected to the Switch Node can receive the packet. The rules given in clause B.4.3.5.3 must be applied. The Service Node should also pass the packet up to its MAC SAP to applications which have registered to receive broadcast packets using the MAC_JOIN service.

When the Base Node receives a broadcast packet with HDR.DO=0, it should pass the packet up its MAC SAP to applications which have registered to receive broadcast packets. The Base Node should also transmit the packet as a Downlink packet, i.e., HDR.DO=1, using the most robust PHY modulation scheme and FEC coding at the maximum output power level and following the rules given in clause B.4.3.5.3.

B.4.3.5.5 Switching of multicast packets

B.4.3.5.5.1 General

Multicast packet switching operates in a very similar way to broadcast packet switching. Multicast is an extension of broadcast. If a switching Node does not implement multicasting, it should handle all multicast packets as broadcast packets.

Different names of MAC header and packets are use in this clause. Refers to the clause B.4.4.2 to find proper definitions.

B.4.3.5.5.2 Multicast switching table

Switch Nodes which implement multicast should maintain a multicast switching table. This table contains a list of multicast group LCIDs that have members connected to the Subnetwork through the Switch Node. The LCID of multicast traffic in both Downlink and uplink directions is checked for a matching entry in the multicast switching table. Multicast traffic is only switched if an entry corresponding to the LCID is available in the table; otherwise, the traffic is silently discarded.

A multicast switching table is established and managed by examining the multicast join and leave messages (MUL control packet) which pass through the Switch. Note that multiple Service Nodes from a Switch Node's control hierarchy may be members of the same group.

On a successful group join from a Service Node in its control hierarchy, a Switch Node adds a new multicast Switch entry for the group LCID, where necessary.

When a successful group leave is indicated, the Switch removes the NID from the multicast Switch entry. If the multicast Switch entry then has no NID associated with it, the multicast Switch entry is immediately removed.

Switch Nodes shall also examine the Keep-Alive packets being passed upwards. When a Service Node that is also a member of a multicast group fails the Keep-Alive process, its NID is removed from any multicast Switch entries and, if necessary, the multicast Switch entry is removed.

Switch Nodes should use a timer to trigger the actual removal of Switch entries. The timer is started when it is decided that an entry should be removed. This timer has value ((*macMaxCtlReTx* + 1) * *macCtlReTxTimer)*. Only once the timer has expired is the multicast Switch entry removed. This allows the Terminal Node a short amount of time to flush any remaining multicast packets before the connection is removed and the Switch Node implementation is simplified since it only needs to process MUL_LEAVE_B or MUL_LEAVE_S (refers to clause B.4.4.5.10), but not both.

B.4.3.5.5.3 Switching process of multicast packets

The multicast packet switching process depends on the packet direction.

When HDR.DO=0 and PKT.LNID=0x3FFE, i.e., the packet is an uplink multicast packet, it is unicast towards the Base Node. A Switch Node that receives such a packet should apply the scope rules to ensure it comes from a lower hierarchical level and, if so, switch it upwards towards the Base Node. No LCID-based filtering is performed. All multicast packets are switched, regardless of any multicast Switch entries for the LCID. The coding rate most applicable to the unicast may be used and the rules given in clause B.4.3.5.3 shall be applied.

When HDR.DO=1 and PKT.LNID=0x3FFE, i.e., the packet is a Downlink multicast packet, the multicast switching table is used. If there is an entry with the LCID corresponding to PKT.LCID in the packet, the packet is switched downwards to the part of subnetwork controlled by this switch. The most robust PHY modulation scheme and FEC coding at the maximum output power level should be used so that all its devices in the lower level can receive the packet. The rules given in clause B.4.3.5.3 shall be applied. If the Service Node is also a member of the multicast group, it should also pass the packet up its MAC SAP to applications which have registered to receive the multicast packets for that group.

When the Base Node receives a multicast packet with HDR.DO=0 and it is a member of the multicast group, it should pass the packet up its MAC SAP to applications which have registered to receive multicast packets for that group. The Base Node should Switch the multicast packet if there is an appropriate entry in its multicast switching table for the LCID, transmitting the packet as a Downlink packet, i.e., HDR.DO=1, using the most robust PHY modulation scheme and FEC coding at the maximum output power level. The rules given in clause B.4.3.5.3 shall be used.

B.4.3.6 Direct connections

B.4.3.6.1 Direct connection establishment

The direct connection establishment is a little different from a normal connection although the same packets and processes are used. It is different because the initial connection request may not be acknowledged until it is already acknowledged by the target Node. It is also different because the CON_REQ_B packets shall carry information for the "direct Switch" to update the "direct switching table".

A direct switch is not different than a general switch. It is only a logical distinction of identifying the first common switch between two service-nodes that need to communicate with each other. Note that in absence of such a common switch, the Base Node would be the direct switch.

There are two different scenarios for using directed connections. These scenarios use the network shown in Figure B.13.

The first is when the source Node does not know the destination Service Node's EUI-48 address. The Service Node initiates a connection to the Base Node and the Base Node Convergence layer redirects the connection to the correct Service Node.

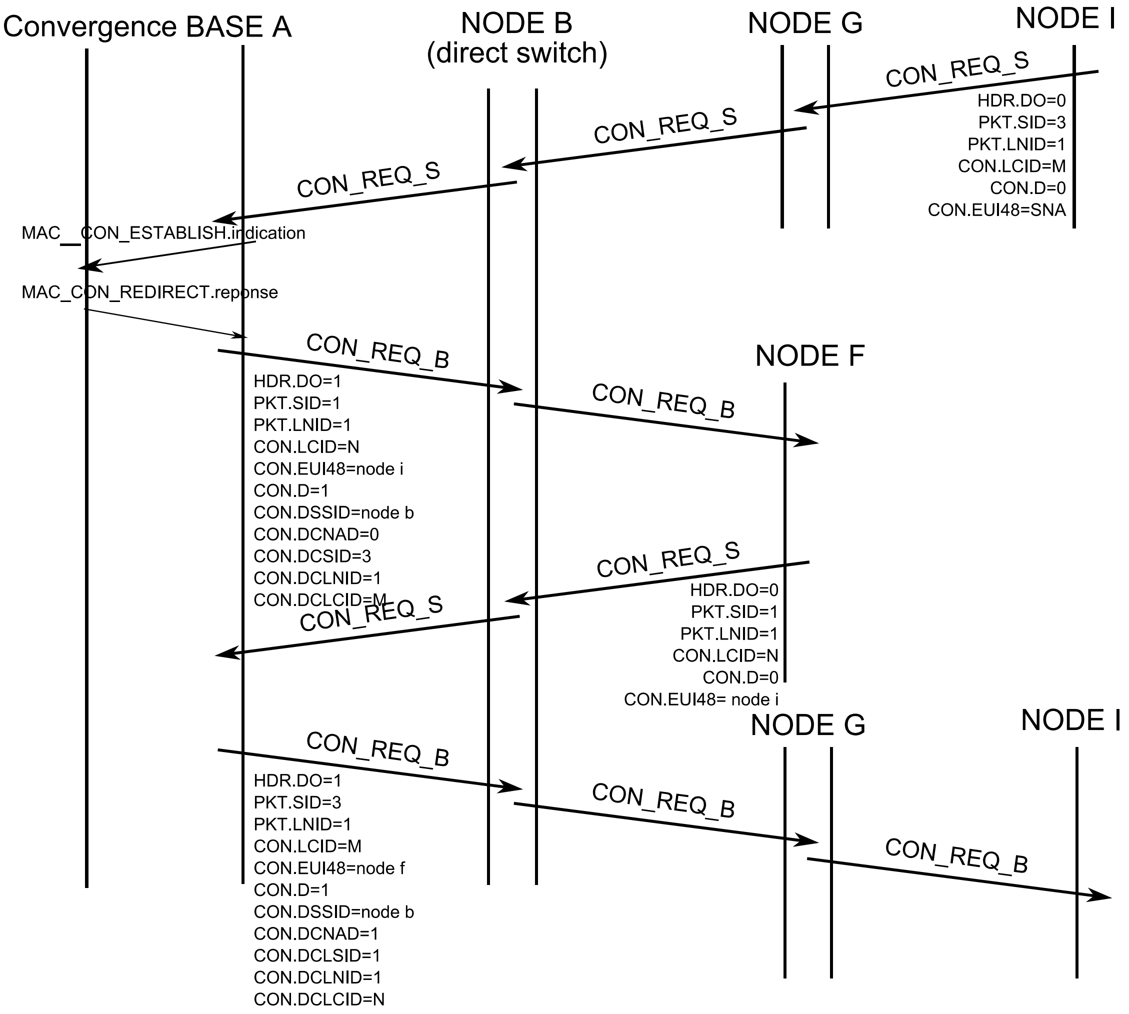


Figure B.13 – Directed connection to an unknown service node

The steps to establish a direct connection, as shown in Figure B.13, shall be:

- When Node I tries to establish connection with Node F, it shall send a normal connection request (CON_REQ_S).
- Then, due to the fact that the Base Node knows that F is the target Service Node, it should send a connection request to F (CON_REQ_B). This packet will carry information for direct Switch B to include the connection in its direct switching table.
- F may accept the connection. (CON_REQ_S).
- Now that the connection with F is fully established, the Base Node will accept the connection with I (CON_REQ_B). This packet will carry information for the direct Switch B to include in its direct switching table.

After finishing this connection-establishment process, the direct Switch (Node B) should contain a direct switching table with the entries shown in Table B.2.

Table B.2 – Direct connection example: Node B's Direct switching table

Uplink			Downlink			
SID	**LNID**	**LCID**	**DSID**	**DLNID**	**DLCID**	**NAD**
1	1	N	3	1	M	0
3	1	M	1	1	N	1

The direct switching table should be updated every time a Switch receives a control packet that meets the following requirements.

- It is CON_REQ_B packet: HDR.DO=1, CON.TYPE=1 and CON.N=0.
- It contains "direct" information: CON.D=1.
- The direct information is for itself: CON.DSSID is the SID of the Switch itself.

Then, the direct switching table is updated with the information:

- Uplink (SID, LNID, LCID) = (PKT.SID, PKT.LNID, CON.LCID).
- Downlink (SID, LNID, LCID, NAD) = (CON.DCSID, CON.DCLNID, CON.DCLCID, CON.DCNAD).

The connection closing packets should be used to remove the entries.

The second scenario for using directed connections is when the initiating Service Node already knows the destination Service Node's EUI-48 address. In this case, rather than using the Base Node's address, it uses the Service Node's address. In this case, the Base Node Convergence layer is not involved. The Base Node MAC layer connects Service Node I directly to Service Node F. The resulting Switch table entries are identical to the previous example. The exchange of signals is shown in Figure B.14.

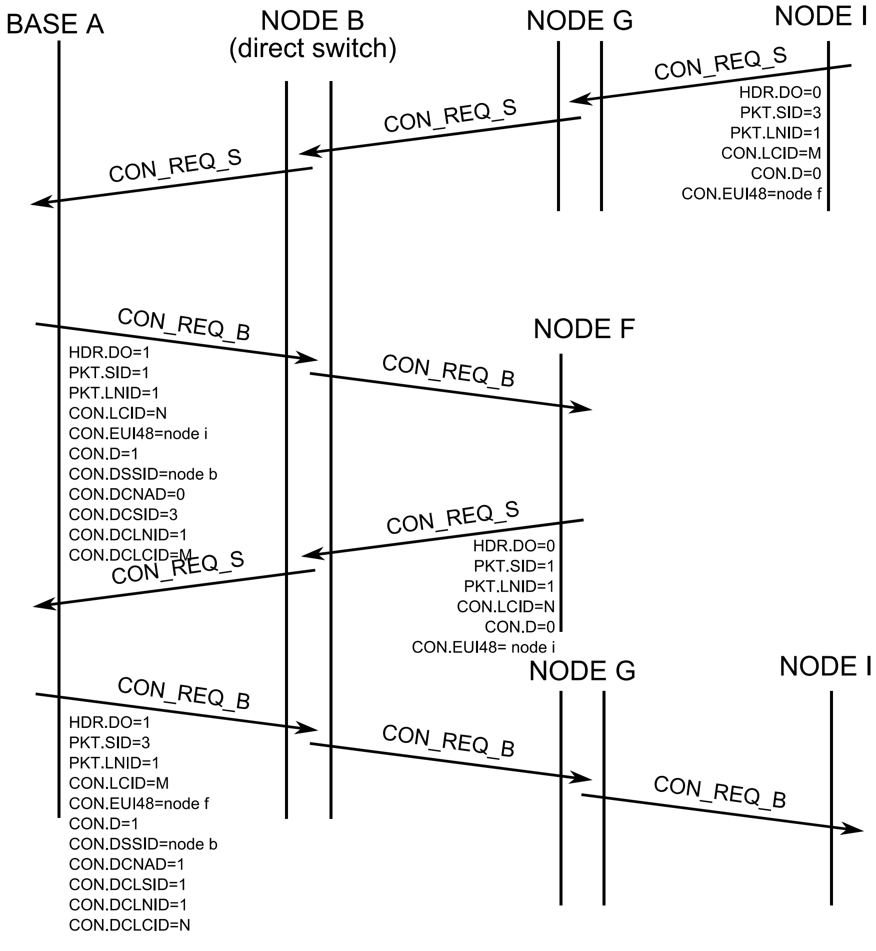


Figure B.14 – Example of direct connection: connection establishment to a known service node

B.4.3.6.2 Direct connection release

The release of a direct connection is shown in Figure B.15. The signalling is very similar to connection establishment for a direct connection. The D fields are used to tell the direct Switch which entries it should remove. The direct switching table should be updated every time a Switch receives a control packet that meets the following requirements.

- It is CON_CLOSE_B packet: HDR.DO=1, CON.TYPE=1 and CON.N=1.
- It contains "direct" information: CON.D=1.
- The direct information is for itself: CON.DSSID is the SID of the Switch itself.
- Then, the direct switching table entry with the following information is removed:
- Uplink (SID, LNID, LCID) = (PKT.SID, PKT.LNID, CON.LCID).
- Downlink (SID, LNID, LCID, NAD) = (CON.DCSID, CON.DCLNID, CON.DCLCID, CON.DCNAD).

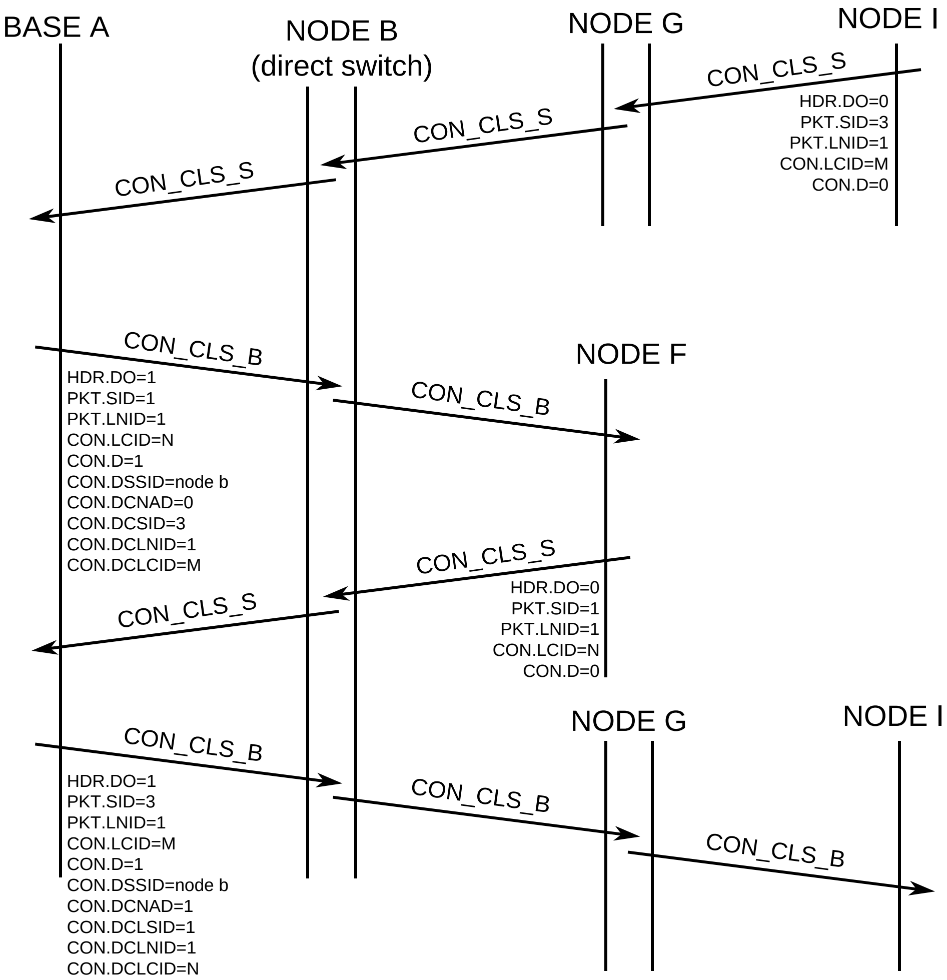


Figure B.15 – Release of a direct connection

B.4.3.6.3 Direct connection switching

As explained in clause B.4.3.5.3, the normal switching mechanism is intended to be used for forwarding communication data between the Base Node and each Service Node. The "direct switching" is a mechanism to let two Nodes communicate with each other, switching the packets in a local way, i.e., without passing through the Base Node. It is not a different form of packet-switching, but rather an additional feature of the general switching process.

The first shared Switch in the paths that go from two Service Nodes to the Base Node will be called the "direct Switch" for the connections between the said Nodes. This is the Switch that will have the possibility of performing the direct switching to make the two Nodes communicate efficiently. As a special case, every Switch is the "direct Switch" between itself and any Node that is lower down in the hierarchy.

The "direct switching table" is a table every Switch should contain in order to perform the direct switching. Each entry on this table is a direct connection that must be switched directly. It is represented by the origin CID and the destination CID of the direct connection. It is not a record of every connection identifier lower down in its hierarchy, but contains only those that should be directly switched by it. The Destination Node's ability to receive aggregated packets shall also be included in the "direct switching table" in order to fill the PKT.NAD field.

B.4.3.6.4 Direct switching operation

If a Switch receives an uplink (HDR.DO=0) MAC frame that is to be switched (see clause B.4.3.5.3 for the requirements) and its address is in the direct switching table, then the procedure is as follows:

- Change the (SID, LNID, LCID, NAD) by the Downlink part of the entry in the direct switching table.
- Queue the packet to be transmitted as a Downlink packet (HDR.DO=1).

B.4.3.7 Packet aggregation

B.4.3.7.1 General

The GPDU may contain one or more packets. The functionality of including multiple packets in a GPDU is called packet aggregation. Packet aggregation is an optional part of this specification and devices do not need to implement it for compliance with this specification. It is however suggested that devices should implement packet aggregation in order to improve MAC efficiency.

To maintain compatibility between devices that implement packet aggregation and ones that do not, there must be a guarantee that no aggregation takes place for packets whose data transit path from/to the Base Node crosses (an) intermediate Service Node(s) that do(es) not implement this function. Information about the aggregation capability of the data transit path is exchanged during the Registration process (B.4.6.1). A registering Service Node notifies this capability to the Base Node in the REG.CAP_PA field (1 bit, see Table B.8) of its REG_REQ message. It gets feedback from the Base Node on the aggregation capability of the whole Downlink transit path in the REG.CAP_PA field of the REG_RSP message.

Based on initial information exchanged on Registration, each subsequent data packet in either direction contains aggregation information in the PKT.NAD field. In the Downlink direction, the Base Node will be responsible for filling PKT.NAD based on the value it communicated to the destination Service Node in the REG.CAP_PA field of the REG_RSP message. Likewise, for uplink data, the source Service Node will fill PKT.NAD based on the REG.CAP_PA field received in the initial REG_RSP from the Base Node. The last Switch shall use the PKT.NAD field to avoid packet aggregation when forwarding the packet to destination Service Nodes without packet aggregation capability. Intermediate Switch Nodes should have information about the aggregation capability in their switching table and shall not aggregate packets when it is known that next level Switch Node does not support this feature.

Devices that implement packet aggregation shall ensure that the size of the MSDU comprising the aggregates does not exceed the maximum capacity of the most robust transmission scheme of a PHY burst. The most robust transmission scheme refers to the most robust combination of modulation scheme and convolutional coding.

B.4.3.7.2 Packet aggregation when switching

Switch Nodes maintain information on the packet aggregation capability of all entries in their switching table, i.e., of all switches that are connected to the Subnetwork through them. This capability information is then used during traffic switching to/from the connected Switch Nodes.

The packet aggregation capability of a connecting Switch Node is registered at each transit Switch Node at the time of its promotion by sniffing relevant information in the PRO_ACK message.

- If the PKT.SID in a PRO_ACK message is the same as the switching Node, the Node being promoted is connected directly to the said Switch Node. The aggregation capability of this new Switch Node is registered as the same as indicated in PKT.NAD of the PRO_ACK packet.
- If the PKT.SID in a PRO_ACK message is different from the SID of the switching Node, it implies that the Node being promoted is indirectly connected to this Switch. The aggregation capability for this new Switch Node will thus be the same as the aggregation capability registered for its immediate Switch, i.e., PKT.SID.

Aggregation while switching packets in uplink direction is performed if the Node performing the Switch knows that its uplink path is capable of handling aggregated packets, based on capability information exchanged during Registration (REG.CAP_PA field in REG_RSP message).

Downlink packets are aggregated by analysing the following:

- If the PKT.SID is the same as the switching Node, then it is the last switching level and the packet will arrive at its destination. In this case, the packet may be aggregated if PKT.NAD=0.
- If the PKT.SID is different, this is not the last level and another Switch will receive the packet. The information of whether or not the packet could be aggregated should be extracted from the switching table.

B.4.3.8 Security

B.4.3.8.1 General

The security functionality provides the MAC layer with privacy, authentication and data integrity through a secure connection method and a key management policy. All packets must use the negotiated security profile. The only exceptions to this rule are the REG and SEC control messages, and the BPDU and PNPDU PDUs which are transferred non-encrypted.

B.4.3.8.2 Security profiles

Several security profiles are provided for managing different security needs, which can arise in different network environments. This version of the specification lists two security profiles and leaves scope for adding up to two new security profiles in future versions.

B.4.3.8.2.1 Security profile 0

Communications having Security Profile 0 are based on the transmission of MAC SDUs without encryption. This profile may be used in application scenarios where either sufficient security is provided by upper communication layers or where security is not a major requirement for application use-case.

B.4.3.8.2.2 Security profile 1

B.4.3.8.2.2.1 General

Security Profile 1 is based on 128-bit AES encryption of data and its associated CRC. This profile is specified with the aim of fulfilling all security requirements:

- Privacy is guaranteed by the encryption itself and by the fact that the encryption key is kept secret.
- Authentication is guaranteed by the fact that each Node has its own secret key known only by the Node itself and the Base Node.
- Data integrity is guaranteed by the fact that the payload CRC is encrypted.

B.4.3.8.2.2.2 Cryptographic primitives

The cryptographic algorithm used in this specification is the AES, as specified in [PUB 197]. The specification describes the algorithm with three possible key sizes; the 128-bit secret key represents a good level of security for preserving privacy up to 2030 and beyond, as specified in [SP 800-57], page 66, table 4.

AES is used according to the so-called ECB, as specified in [SP 800-38A]. It is a block-ciphering mode where plain text is divided into 128-bit blocks. Padding is applied if the last block is smaller than 128 bits. Padding is implemented with the addition of a bit equal to 1 and as many zeroes as necessary to reach a length of the string to be encrypted as a multiple of 128 bits. Encryption is performed one block at a time, using the same working key for all the data.

B.4.3.8.2.2.3 Key derivation algorithm

The method for deriving working keys from secret keys is to apply the AES algorithm to a constant (C) as plain text and generation key (GK) as an encryption key. If the constant is shorter than 128 bits, it must be aligned to the LSB, as shown in Figure B.16. The various key derivation equations specified in the following subclauses follow the convention:

Generated Key = AES_enc (*Generation Key*, *Constant*)

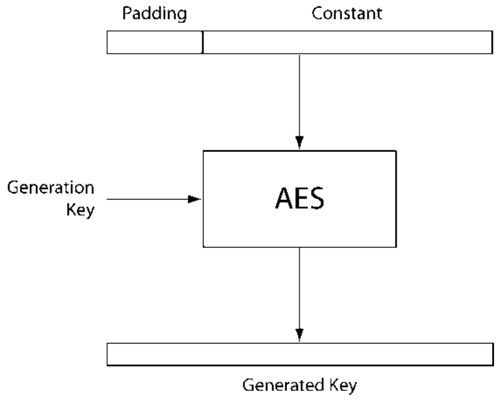


Figure B.16 – Key derivation concept

B.4.3.8.3 Negotiation of the security profile

All MAC data, including signalling PDUs (all MAC control packets defined in clause B.4.4.5) use the same security profile. This profile is negotiated during the device Registration. In the REG_REQ message the Terminal indicates a security profile it is able to support in the field REG.SPC. The Base Node may accept this security profile and so accept the Registration, sending back a REG_RSP with the same REG.SPC value. The Base Node may also accept the Registration, however it sets REG.SPC to 0 indicating that security profile 0 is to be used. Alternatively, the Base Node may reject the Registration if the Terminal does not provide an acceptable security profile.

It is recommended that the Terminal first attempts to register using the highest security profile it supports and only use lower security profiles when the Base Node rejects the Registration request.

B.4.3.8.3 Key hierarchy

B.4.3.8.4.1 Security profile 0

Not applicable.

B.4.3.8.4.2 Security profile 1

Service Nodes and Base Nodes use a set of three working keys to encrypt all data. The keys and their respective usage are:

Initial Working Key (WK0): This key has limited scope and is used to decrypt the REG.SNK and REG.AUK fields of the REG_RSP message. The WK0 is thus used by a Service Node in a *Disconnected* functional state. This key is computed using the following formula:

WK0 = AES_enc (*USK*, *0*)

Working Key (WK): This key is used to encrypt all the unicast data that is transmitted from the Base Node to a Service Node and vice versa. Each registered Service Node would have a unique WK that is known only to the Base Node and itself. The WK is computed as follows:

WK = AES_enc (*USK, Random sequence received in SEC.RAN*)

Subnetwork Working Key (SWK): The SWK is shared by the entire Subnetwork. To ensure the security of this key, it is never transmitted over the physical channel, but is computed from other keys which are transmitted encrypted in REG and non-encrypted in SEC control packets. The SWK shall be used to encrypt the following:

- Broadcast data, including MAC broadcast control packets.
- Multicast data.
- Unicast data that is transacted over direct connections, i.e., not involving the Base Node.

The SWK is computed as follows:

SWK = AES_enc (*SNK, Random sequence received in SEC.SNK*)

The WK and the SWK have a limited validity time related to the random sequence generation period. The random sequence is regenerated and distributed by the Base Node at least every *MACRandSeqChgTime* seconds through the SEC control packet. If a device does not receive an update of a random sequence within 2 * *MACRandSeqChgTime*, it should consider the WK and SWK as no longer valid. Lack of availability of WK will render Service Node to very limited functionality of unregistering from the subnetwork. It is therefore advised that in such cases, Service Nodes should unregister from the network and initiate a re-registration procedure.

The key derivation procedures have been designed to be indirect and multi-staged to ensure security. The parameters involved in the derivation of the working keys are defined below.

Master Key (MK1, MK2): Two Master Keys (MK1 and MK2) are defined in this specification. MK1 is used to compute the DSK. MK2 is used to compute the KDIV. Both of these keys are administered on the Base Node by implementation-dependent means that are beyond the scope of this specification. Specifying two master keys makes the USK generation a two stage process, i.e., derivation of DSK and KDIV in the first stage and using them to derive the USK in the second stage. Note that the DSK and KDIV are unique to each registering Service Node.

Device Secret key (DSK): DSK is unique to each Service Node on the Subnetwork and is hard-coded in the device during production. The DSK is constant for the entire life of the Service Node. The Base Node uses MK1 to derive Service Node-specific DSK using the following equation:

DSK = AES_enc (*MK1*, *UI*)

Key Diversifier (KDIV): This quantity is also unique to each Service Node, but unlike DSK, it does not have to be a fixed constant for the entire life of the Service Node. The KDIV is provisioned on each Service Node by means that are beyond the scope of this specification. The Base Node computes device-specific KDIV using the equation:

$$KDIV = \text{AES\_enc}\ (MK2, UI)$$

Unique Secret Key (USK): The USK is used to derive WK0 and WK as defined in the above equations. The USK is in turn computed by applying AES to KDIV, using DSK as the generation key, as shown in the equation below. Note that this is a single-step process in Service Nodes because both KDIV and DSK are already known or provisioned, but a three-step process in the Base Node. The first two steps in the Base Node comprise deriving the DSK and KDIV using the MK1 and MK2, respectively.

$$USK = \text{AES\_enc}\ (DSK, KDIV)$$

Unique Identifier (UI): The UI of a Service Node shall be its EUI-48.

B.4.3.8.5 Key distribution and management

The Security Profile for data traffic is negotiated when a device is registered. The REG control packet contains specific fields to indicate Security Profile for respective devices. All connections to/from the device would be required to follow the Security Profile negotiated at the time of Registration. There cannot be a difference in Security Profile across multiple connections involving the same device. The only exception to this would be the Base Node.

The SWK used as a working key for non-unicast traffic and direct connections is never transmitted in non-encrypted form over the physical channel. The SEC broadcast messages transmitted by the Base Node (and relayed by all Switch Nodes) at regular intervals contain random keys for both unicast and non-unicast traffic. When a device initially registers on a Subnetwork, the REG response from the Base Node contains the random sequence used to derive WK for unicast traffic. The REG message is followed by a unicast SEC message from Base Node to the registering device.

B.4.3.8.6 Encryption

B.4.3.8.6.1 Security profile 0

Not Applicable.

B.4.3.8.6.2 Security profile 1

Connections working with "Security Profile 1" would always transmit a CRC with every packet. This field shall be called SCRC (Security CRC) and is calculated over the unencrypted packet payload. The SCRC helps confirming the integrity of the packet on its decryption at the receiving end.

The SCRC shall be calculated as the remainder of the division (Modulo 2) by the generator polynomial $g(x)=x^8+x^2+x+1$ of the polynomial x^8 multiplied by the unencrypted packet payload.

The data block obtained by the concatenation of the unencrypted payload of the packet and the calculated SCRC is padded with a 1 followed by as many zeroes as necessary to reach a multiple of 128 and then divided into 128-bit blocks. The 1 inserted as the first padding bit is useful to detect the start of the padding at the receiver without notification of the number of padded bits.

Each 128-bit block is encrypted with the AES algorithm using a valid working key. The result of this encryption process is the encrypted payload of the packet.

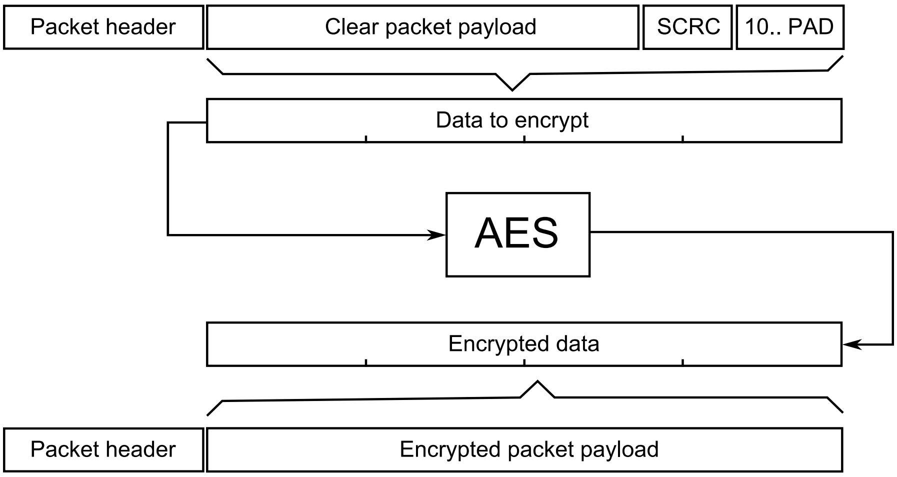


Figure B.17 – Security Profile 1 encryption algorithm

B.4.4 MAC PDU format

B.4.4.1 General

There are different types of MAC PDUs for different purposes.

B.4.4.2 Generic MAC PDU

B.4.4.2.1 General

Most Subnetwork traffic comprises Generic MAC PDUs (GPDU). GPDUs are used for all data traffic and most control traffic. All MAC control packets are transmitted as GPDUs.

GPDU composition is shown in Figure B.18. It is composed of a Generic MAC Header followed by one or more MAC packets and 32 bit CRC appended at the end.

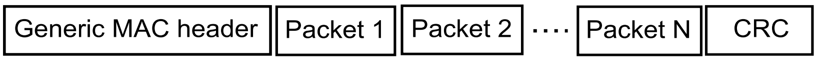


Figure B.18 – Generic MAC PDU format

B.4.4.2.2 Generic MAC header

The Generic MAC Header format is represented in Table B.3. The size of the Generic MAC Header is 3 bytes. Table B.3 enumerates each field of a Generic MAC Header.

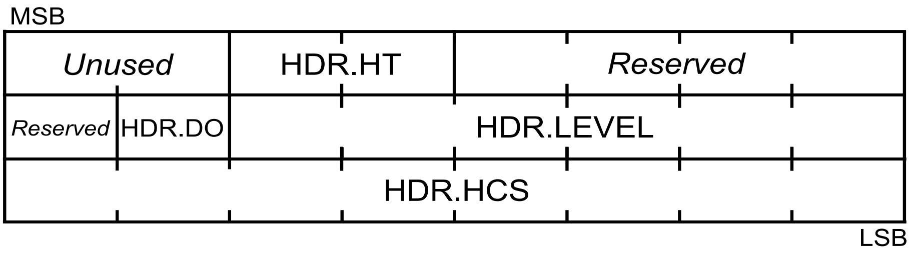


Figure B.19 – Generic MAC header

Table B.3 – Generic MAC header fields

Name	Length	Description
Unused	2 bits	Unused bits which are always 0; included for alignment with MAC_H field in PPDU header (see Figure B.3/ [ITU-T G.9955]).
HDR.HT	2 bits	Header Type. HDR.HT = 0 for GPDU
Reserved	5 bits	Always 0 for this version of the specification. Reserved for future use.
HDR.DO	1 bit	Downlink/Uplink. • HDR.DO=1 if the MAC PDU is Downlink. • HDR.DO=0 if the MAC PDU is uplink.
HDR.LEVEL	6 bits	Level of the PDU in switching hierarchy. The packets between the level 0 and the Base Node are of HDR.LEVEL=0. The packets between levels k and k-1 are of HDR.LEVEL=k. • If HDR.DO=0, HDR.LEVEL represents the level of the transmitter of this packet. • If HDR.DO=1, HDR.LEVEL represents the level of the receiver of this packet.
HDR.HCS	8 bits	Header Check Sequence. A field for detecting errors in the header and checking that this MAC PDU is from this Subnetwork. The transmitter shall calculate the CRC of the SNA concatenated with the first 2 bytes of the header and insert the result into the HDR.HCS field (the last byte of the header). The CRC shall be calculated as the remainder of the division (Modulo 2) of the polynomial M(x)·x8 by the generator polynomial g(x)=x8+x2+x+1. M(x) is the input polynomial, which is formed by the bit sequence of the concatenation of the SNA and the header excluding the HDR.HCS field, and the msb of the bit sequence is the coefficient of the highest order of M(x).

B.4.4.2.3 Packet structure

A packet is comprised of a Packet Header and Packet Payload. Figure B.20 shows the structure.

Packet header
Packet payload

Figure B.20 – Packet structure

Packet header is 6 bytes in length and its composition is shown in Figure B.21. Table B.4 enumerates the description of each field.

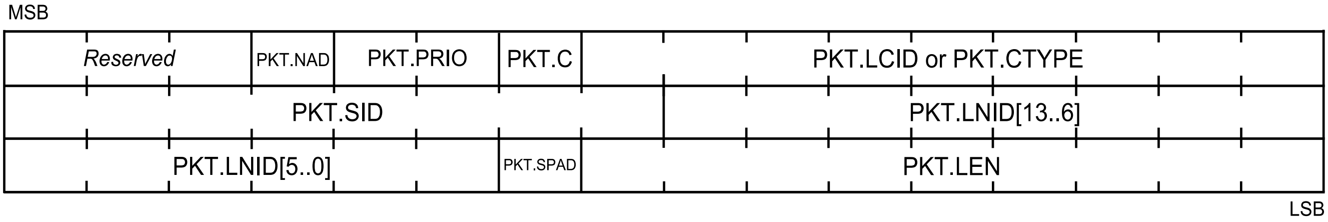


Figure B.21 – Packet header

To simplify, the text contains references to the PKT.NID fields as the composition of the PKT.SID and PKT.LNID. The field PKT.CID is also described as the composition of the PKT.NID and the PKT.LCID. The composition of these fields is described in Figure B.22.

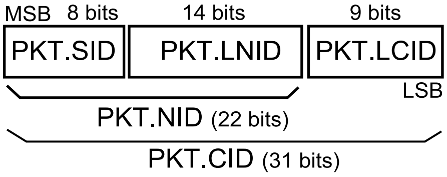


Figure B.22 – PKT.CID structure

Table B.4 – Packet header fields

Name	Length	Description
Reserved	3 bits	Always 0 for this version of the specification. Reserved for future use.
PKT.NAD	1 bit	No Aggregation at Destination • If PKT.NAD=0 the packet may be aggregated with other packets at destination. • If PKT.NAD=1 the packet may not be aggregated with other packets at destination.
PKT.PRIO	2 bits	Indicates packet priority between 0 and 3.
PKT.C	1 bits	Control • If PKT.C=0 it is a data packet. • If PKT.C=1 it is a control packet.
PKT.LCID/ PKT.CTYPE	9 bits	Local Connection Identifier or Control Type • If PKT.C=0, PKT.LCID represents the Local Connection Identifier of data packet. • If PKT.C=1, PKT.CTYPE represents the type of the control packet.
PKT.SID	8 bits	Switch identifier • If HDR.DO=0, PKT.SID represents the SID of the packet source. • If HDR.DO=1, PKT.SID represents the SID of the packet destination.
PKT.LNID	14 bits	Local Node identifier. • If HDR.DO=0, PKT.LNID represents the LNID of the packet source • If HDR.DO=1, PKT.LNID represents the LNID of the packet destination.
PKT.SPAD	1bit	Indicates if padding is inserted while encrypting payload. Note that this bit is only of relevance when Security Profile 1 (see B.4.3.8.2.2) is used.
PKT.LEN	9 bits	Length of the packet payload in bytes.

B.4.4.2.4 CRC

The CRC is the last field of the GPDU. It is 32 bits long. It is used to detect transmission errors. The CRC shall cover the concatenation of the SNA with the GPDU except for the CRC field itself.

The input polynomial M(x) is formed as a polynomial whose coefficients are bits of the data being checked (the first bit to check is the highest order coefficient and the last bit to check is the coefficient of order zero). The Generator polynomial for the CRC is $G(x)=x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^{8}+x^{7}+x^{5}+x^{4}+x^{2}+x+1$. The remainder R(x) is calculated as the remainder from the division of $M(x)\cdot x^{32}$ by G(x). The coefficients of the remainder will then be the resulting CRC.

B.4.4.3 Promotion needed PDU

If a Node is Disconnected and it does not have connectivity with any existing Switch Node, it shall send notifications to its neighbours to indicate the need for the promotion of any available Terminal

Node. Figure B.23 represents the Promotion Needed MAC PDU (PNPDU) that must be sent on an irregular basis in this situation.

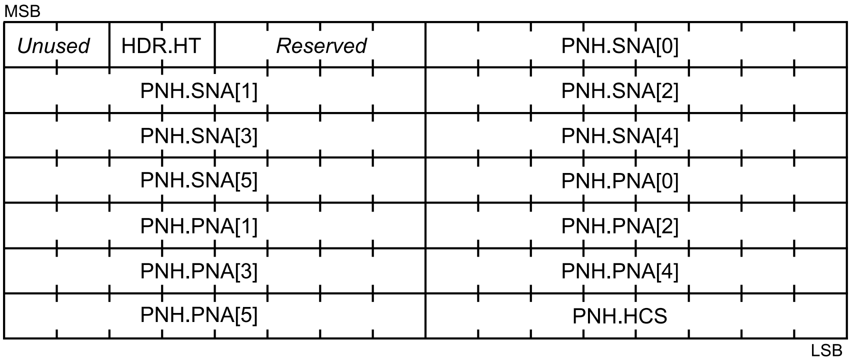


Figure B.23 – Promotion need MAC PDU

Table B.5 shows the promotion need MAC PDU fields.

Table B.5 – Promotion need MAC PDU fields

Name	Length	Description
Unused	2 bits	Unused bits which are always 0; included for alignment with MAC_H field in PPDU header (clause B.3.4.3 of [ITU-T G.9955]).
HDR.HT	2 bits	Header Type HDR.HT = 1 for the Promotion Need MAC PDU
Reserved	4 bits	Always 0 for this version of the specification. Reserved for future use.
PNH.SNA	48 bits	Subnetwork Address. The EUI-48 of the Base Node of the Subnetwork the Service Node is trying to connect to. FF:FF:FF:FF:FF:FF to ask for the promotion in any available Subnetwork. SNA[0] is the most significant byte of the OUI/IAB and SNA[5] is the least significant byte of the extension identifier, as defined in: http://standards.ieee.org/regauth/oui/tutorials/EUI-48.html. The above notation is applicable to all EUI-48 fields in the specification.
PNH.PNA	48 bits	Promotion Need Address. The EUI-48 of the Node that needs the promotion. It is the EUI-48 of the transmitter.
PNH.HCS	8 bits	Header Check Sequence. A field for detecting errors in the header. The transmitter shall calculate the PNH.HCS of the first 13 bytes of the header and insert the result into the PNH.HCS field (the last byte of the header). It shall be calculated as the remainder of the division (Modulo 2) of the polynomial $M(x)\cdot x^8$ by the generator polynomial $g(x)=x^8+x^2+x+1$. M(x) is the input polynomial, which is formed by the bit sequence of the header excluding the PNH.HCS field, and the msb of the bit sequence is the coefficient of the highest order of M(x).

As it is always transmitted by unsynchronized Nodes and, therefore, prone to creating collisions, it is a special reduced size header. It is broadcast to any other Terminal Node and shall therefore be transmitted with the most robust scheme of the PHY layer.

B.4.4.4 Beacon PDU

Beacon PDU (BPDU) is transmitted by every Switch device on the Subnetwork, including the Base Node. The purpose of this PDU is to circulate information on MAC frame structure and therefore channel access to all devices that are part of this Subnetwork. The BPDU is transmitted at definite fixed intervals of time and is also used as a synchronization mechanism by Service Nodes. Figure B.24 below shows contents of a beacon transmitted by the Base Node and each Switch Device.

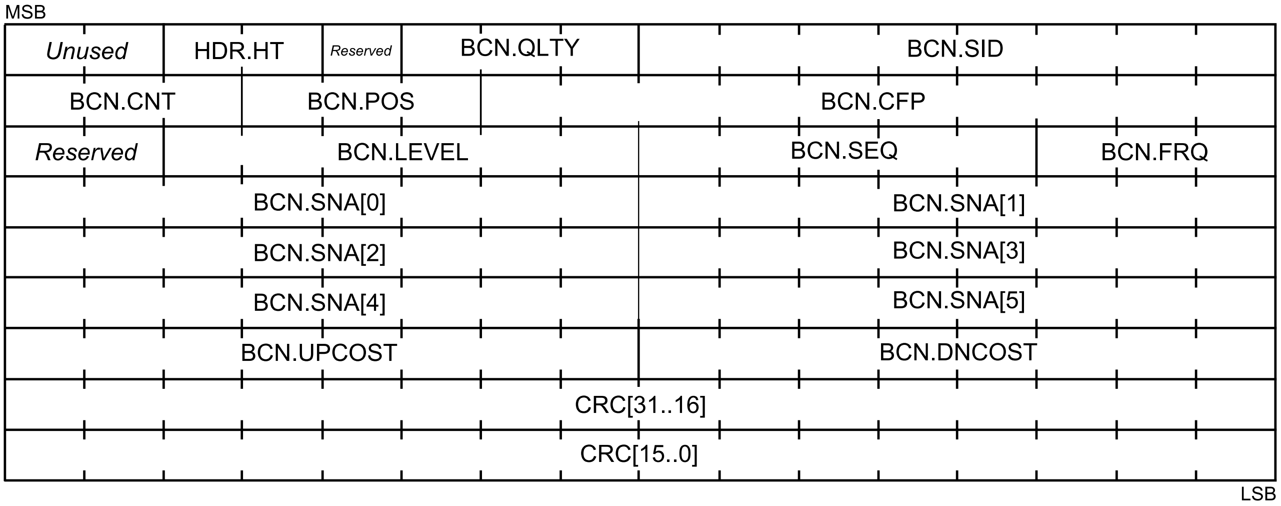


Figure B.24 – Beacon PDU structure

Table B.6 shows the beacon PDU fields.

Table B.6 – Beacon PDU fields

Name	Length	Description
Unused	2 bits	Unused bits which are always 0; included for alignment with MAC_H field in PPDU header (Figure B.7 of [ITU-T G.9955]).
HDR.HT	2 bits	Header Type HDR.HT = 2 for Beacon PDU
Reserved	1 bit	Always 0 for this version of the specification. Reserved for future use.
BCN.QLTY	3 bits	Quality of round-trip connectivity from this Switch Node to the Base Node. BCN.QLTY=7 for best quality (Base Node or very good Switch Node), BCN.QLTY=0 for worst quality (Switch having unstable connection to Subnetwork)
BCN.SID	8 bits	Switch identifier of transmitting Switch
BCN.CNT	3 bits	Number of beacon-slots in this frame
BCN.SLT	3 bits	Beacon-slot in which this BPDU is transmitted BCN.SLT=0 is reserved for the Base Node
BCN.CFP	10 bits	Offset of CFP from start of frame BCN.CFP=0 indicates absence of CFP in a frame.
Reserved	1 bit	Always 0 for this version of the specification. Reserved for future use.
BCN.LEVEL	6 bits	Hierarchy of transmitting Switch in Subnetwork
BCN.SEQ	5 bits	Sequence number of this BPDU in super frame. Incremented for every beacon the Base Node sends and is propagated by Switch through its BPDU such that entire Subnetwork has the same notion of sequence number at a given time.

Table B.6 – Beacon PDU fields

Name	Length	Description
BCN.FRQ	3 bits	Transmission frequency of this BPDU. Values are interpreted as follows: 0 = 1 beacon every frame 1 = 1 beacon every 2 frames 2 = 1 beacon every 4 frames 3 = 1 beacon every 8 frames 4 = 1 beacon every 16 frames 5 = 1 beacon every 32 frames 6 = Reserved 7 = Reserved
BCN.SNA	48 bits	Subnetwork identifier in which the Switch transmitting this BPDU is located

BCN.UPCOST	8 bits	Total uplink cost from the transmitting Switch Node to the Base Node. The cost of a single hop is calculated based on modulation scheme used on that hop in uplink direction. Values are derived as follows:
		8PSK = 0
		QPSK = 1
		BPSK = 2
		8PSK_F = 1
		QPSK_F = 2
		BPSK_F = 4
		The Base Node will transmit in its beacon a BCN.UPCOST of 0. A Switch Node will transmit in its beacon the value of BCN.UPCOST received from its upstream Switch Node, plus the cost of the upstream uplink hop to its upstream Switch. When this value is larger than what can be held in BCN.UPCOST the maximum value of BCN.UPCOST should be used.

BCN.DNCOST	8 bits	Total Downlink cost from the Base Node to the transmitting Switch Node. The cost of a single hop is calculated based on modulation scheme used on that hop in Downlink direction. Values are derived as follows: 8PSK 0 QPSK 1 BPSK 2 8PSK_F 1 QPSK_F 2 BPSK_F 4 The Base Node will transmit in its beacon a BCN.DNCOST of 0. A Switch Node will transmit in its beacon the value of BCN.DNCOST received from its upstream Switch Node, plus the cost of the upstream Downlink hop from its upstream Switch. When this value is larger than what can be held in BCN.DNCOST the maximum value of BCN.DNCOST should be used.
CRC	32 bits	The CRC shall be calculated with the same algorithm as the one defined for the CRC field of the MAC PDU (see clause B.4.4.2.4 for details). This CRC shall be calculated over the complete BPDU except for the CRC field itself.

The BPDU is also used to detect when the uplink Switch is no longer available either by a change in the characteristics of the medium or because of failure, etc. If a Service Node fails to receive $N_{miss\text{-}beacon}$ in a row it should declare the link to its Switch as unusable. The Service Node should stop sending beacons itself if it is acting as a Switch. It should close all existing MAC connections.

The Service Node then enters the initial unregistered functional state and searches for a Subnetwork join. This mechanism complements the Keep-Alive mechanism which is used by a Base Node and its switches to determine when a Service Node is lost.

B.4.4.5 MAC control packets

B.4.4.5.1 General

MAC control packets enable a Service Node to communicate control information with their Switch Node, Base Node and vice versa. A control packet is transmitted as a GPDU and is identified with PKT.C bit set to 1 (See clause B.4.4.2 for more information about the fields of the packets).

There are several types of control messages. Each control message type is identified by the field PKT.CTYPE. Table B.7 lists the types of control messages. The packet payload (see clause B.4.4.2.3) shall contain the information carried by the control packets. This information differs depending on the packet type.

Table B.7 – MAC control packet types

Type (PKT.CTYPE)	Packet name	Packet description
1	REG	Registration management
2	CON	Connection management
3	PRO	Promotion management
4	BSI	Beacon Slot Indication
5	FRA	Frame structure change
6	CFP	Contention-Free Period request
7	ALV	Keep-Alive
8	MUL	Multicast Management
9	PRM	PHY Robustness Management
10	SEC	Security information

B.4.4.5.2 Control packet retransmission

For recovery from lost control messages, a retransmit scheme is defined. MAC control transactions comprising of exchange of more than one control packet may follow the retransmission mechanism described in this clause.

The retransmission scheme shall be applied to the following packets when they require a response:

- CON_REQ_S, CON_REQ_B;
- CON_CLS_S, CON_CLS_B;
- REG_RSP;
- PRO_REQ_B;
- BSI_IND;
- MUL_JOIN_S, MUL_JOIN_B;
- MUL_LEAVE_S, MUL_LEAVE_B;

Devices involved in a MAC control transaction using retransmission mechanism shall keep count of number of times a message has been retransmitted and maintain a retransmit timer.

At the requester of a control message transaction:

- When the first message in a transaction is transmitted, the retransmit timer is started with value *macCtlReTxTimer* and the retransmit count is set to 0.

If a response message is received the retransmit timer is stopped and the transaction is considered complete. Note that it is possible to receive further response messages. These would be messages that encountered network delays.

- If the retransmit timer expires, the retransmit counter is incremented. If the retransmit counter is less than *macMaxCtlReTx* the control message is retransmitted. If the counter is equal to the maximum number of retransmits, failure result corresponding to respective MAC-SAP should be returned to the calling entity. Implementations may also choose to inform their local management entity of such failure. If the retransmission is done by the Service Node, the device should return to the disconnected state.

At the responder of a control message transaction:

- The receiver of a message must determine itself if this message is a retransmit. If so, no local action is needed other than sending a reply to the response.

If the received message is not a retransmit, the message should be processed and a response returned to the sender.

- For transactions which use three messages in the transaction, e.g., promotion as shown in B.4.6.3, the responder should perform retransmits in exactly the same way as the requester. This ensures that if the third message in the transaction is lost, the message will be retried and the transaction completed.

The following message sequence charts show some examples of retransmission. Figure B.25 shows two successful transactions without requiring retransmits.

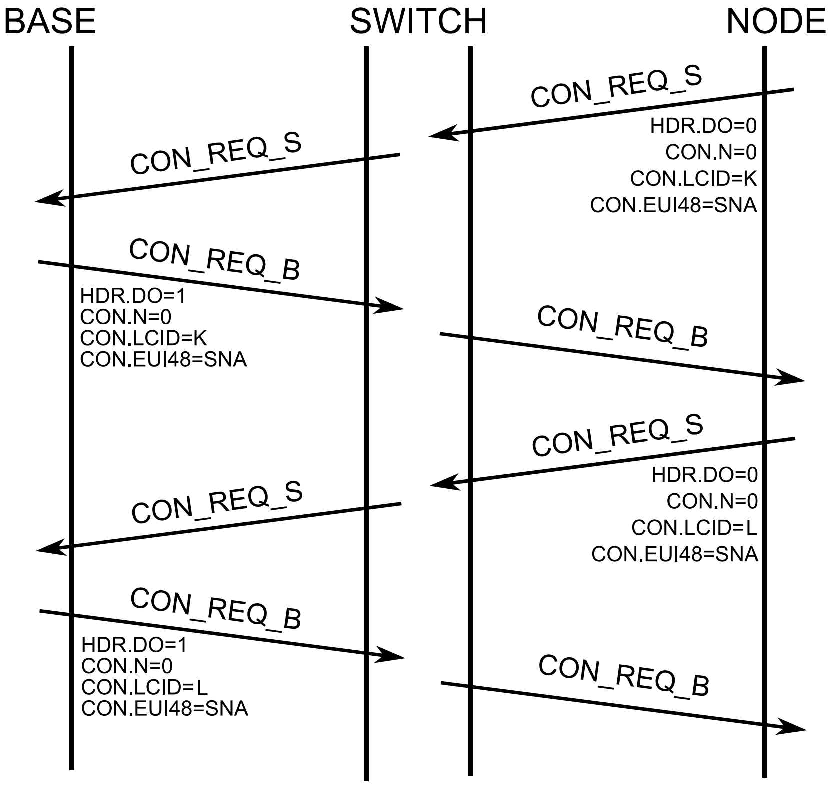


Figure B.25 – Two transactions without requiring retransmits

Figure B.26 shows a more complex example, where messages are lost in both directions causing multiple retransmits before the transaction completes.

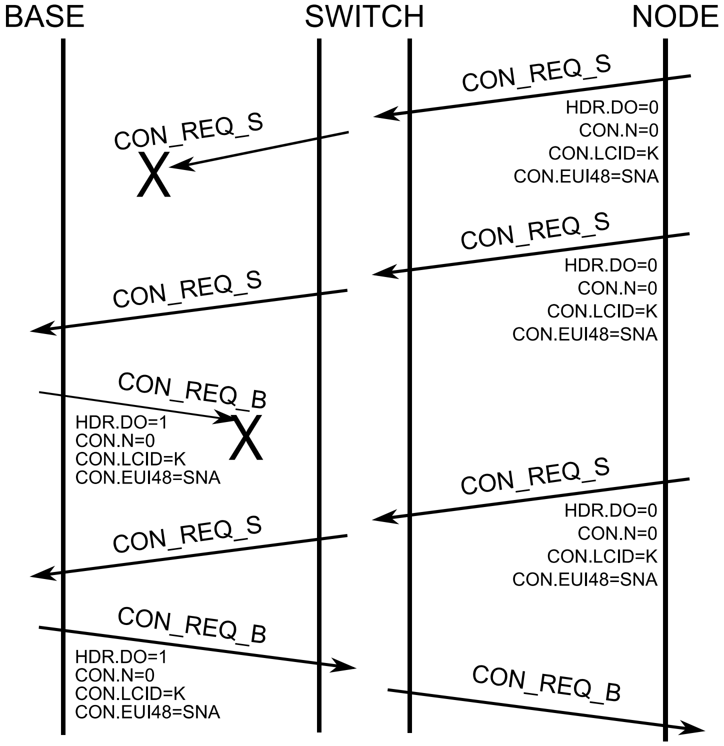


Figure B.26 – Transaction with packet loss requiring retransmits

Figure B.27 shows the case of a delayed response causing duplication at the initiator of the control transaction.

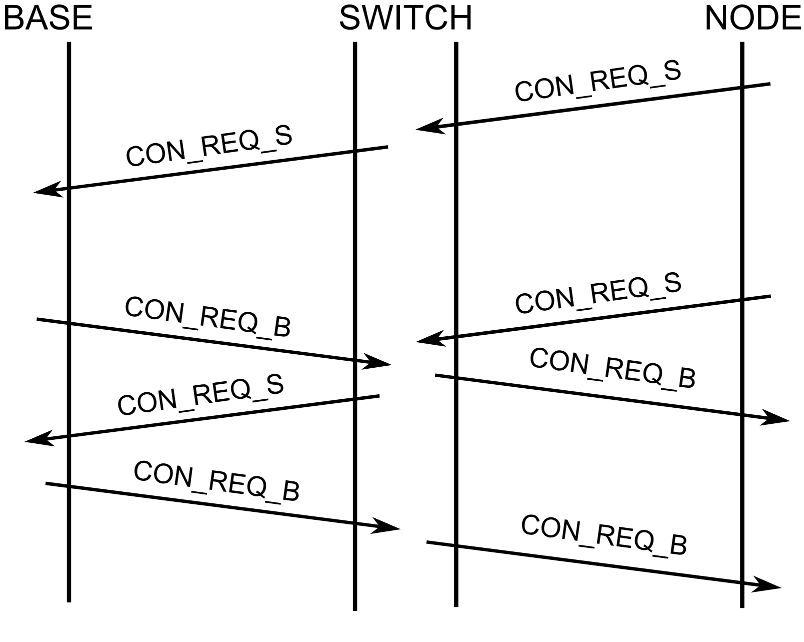


Figure B.27 – Duplicate packet detection and elimination

B.4.4.5.3 REG control packet (PKT.CTYPE=1)

This control packet is used to negotiate the Registration process. The description of data fields of this control packet is described in Table B.8 and Figure B.28. The meaning of the packets differs depending on the direction of the packet. This packet interpretation is explained in Table B.9. These packets are used during the registration and unregistration processes, as explained in B.4.6.1 and B.4.6.2.

The PKT.SID field is used in this control packet as the Switch where the Service Node is registering. The PKT.LNID field is used in this control packet as the Local Node Identifier being assigned to the Service Node during the registration process negotiation.

The REG.CAP_PA field is used to indicate the packet aggregation capability as discussed in clause B.4.3.7. In the uplink direction, this field is an indication from the registering Terminal Node about its own capabilities. For the Downlink response, the Base Node evaluates whether or not all the devices in the cascaded chain from itself to this Terminal Node have packet-aggregation capability. If they do, the Base Node shall set REG.CAP_PA=1; otherwise REG.CAP_PA=0.

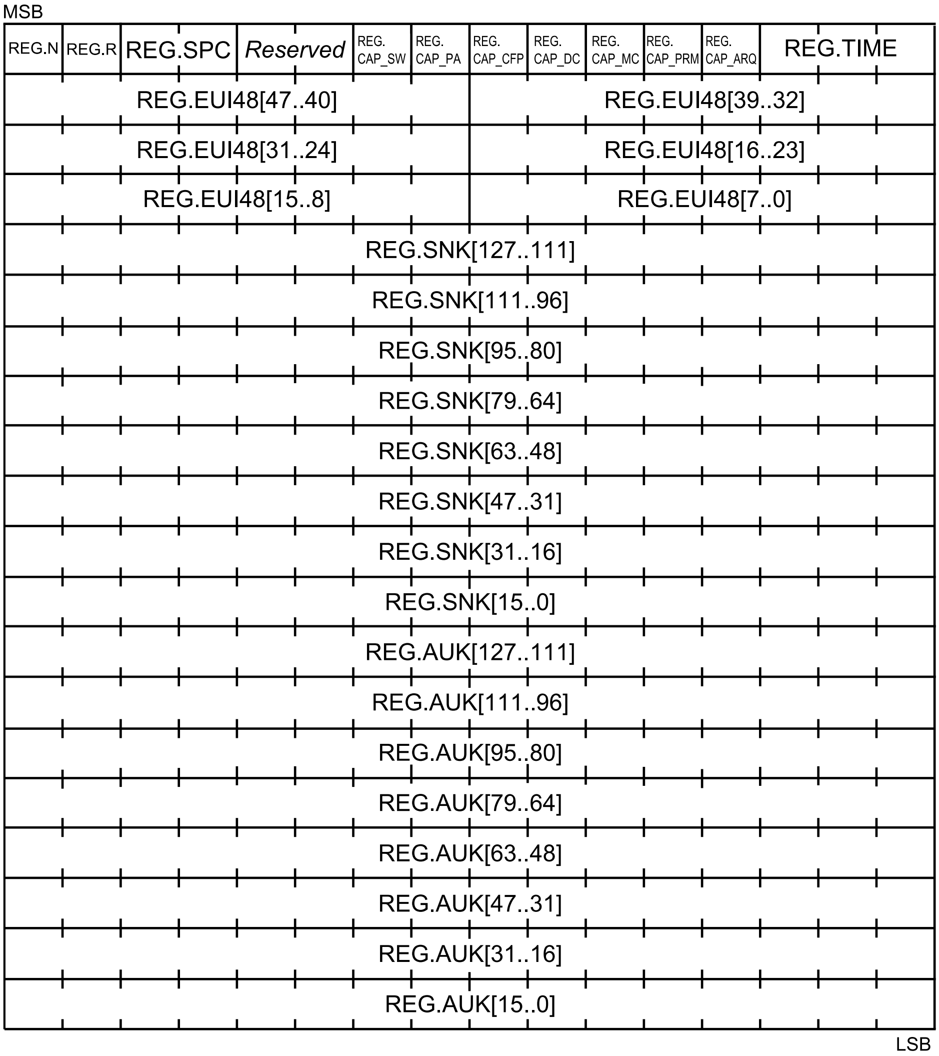


Figure B.28 – REG control packet structure

Table B.8 – REG control packet fields

Name	Length	Description
REG.N	1 bit	Negative • REG.N=1 for the negative register; • REG.N=0 for the positive register. (see Table B.9)
REG.R	1 bit	Roaming • REG.R=1 if Node already registered and wants to perform roaming to another Switch; • REG.R=0 if Node not yet registered and wants to perform a clear registration process.
REG.SPC	2 bits	Security Profile Capability for Data PDUs: • REG.SPC=0 No encryption capability; • REG.SPC=1 Security profile 1 capable device; • REG.SPC=2 Security profile 2 capable device (not yet specified); • REG.SPC=3 Security profile 3 capable device (not yet specified).
Reserved	2 bits	Reserved for future versions of the protocol. Should be set to 0 for this version of the protocol.
REG.CAP_SW	1 bit	Switch Capable 1 if the device is able to behave as a Switch Node; 0 if the device is not.
REG.CAP_PA	1 bit	Packet Aggregation Capability 1 if the device has packet aggregation capability (uplink) if the data transit path to the device has packet aggregation capability (Downlink) 0 otherwise.
REG.CAP_CFP	1 bit	Contention Free Period Capability 1 if the device is able to perform the negotiation of the CFP; 0 if the device cannot use the Contention Free Period in a negotiated way.
REG.CAP_DC	1 bit	Direct Connection Capability 1 if the device is able to perform direct connections; 0 if the device is not able to perform direct connections.
REG.CAP_MC	1 bit	Multicast Capability 1 if the device is able to use multicast for its own communications; 0 if the device is not able to use multicast for its own communications.
REG.CAP_PRM	1 bit	PHY Robustness Management Capable 1 if the device is able to perform PHY Robustness Management; 0 if the device is not able to perform PHY Robustness Management.
REG.CAP_ARQ	1 bit	ARQ Capable 1 if the device is able to establish ARQ connections; 0 if the device is not able to establish ARQ connections.

Table B.8 – REG control packet fields

Name	Length	Description

REG.TIME	3 bits	Time to wait for an ALV_B messages before assuming the Service Node has been unregistered by the Base Node. For all messages except REG_RSP this field should be set to 0. For REG_RSP its value means:
		ALV.TIME = 0 => 32 seconds;
		ALV.TIME = 1 => 64 seconds;
		ALV.TIME = 2 => 128 seconds ~ 2.1 minutes;
		ALV.TIME = 3 => 256 seconds ~ 4.2 minutes;
		ALV.TIME = 4 => 512 seconds ~ 8.5 minutes;
		ALV.TIME = 5 => 1024 seconds ~ 17.1 minutes;
		ALV.TIME = 6 => 2048 seconds ~ 34.1 minutes;
		ALV.TIME = 7 => 4096 seconds ~ 68.3 minutes.

REG.EUI-48	48 bit	EUI-48 of the Node EUI-48 of the Node requesting the Registration.
REG.SNK	128 bits	Encrypted Subnetwork key that shall be used to derive the Subnetwork working key
REG.AUK	128 bits	Encrypted authentication key. This is a random sequence meant to act as authentication mechanism.

Table B.9 – REG control packet types

Name	HDR.DO	PKT.LNID	REG.N	REG.R	Description
REG_REQ	0	0x3FFF	0	R	Registration request • If R=0 any previous connection from this Node should be lost; • If R=1 any previous connection from this Node should be maintained.
REG_RSP	1	< 0x3FFF	0	R	Registration response. This packet assigns the PCK.LNID to the Service Node.
REG_ACK	0	< 0x3FFF	0	R	Registration acknowledged by the Service Node.
REG_REJ	1	0x3FFF	1	0	Registration rejected by the Base Node.
REG_UNR_S	0	< 0x3FFF	1	0	• After a REG_UNR_B: Unregistration acknowledge; • Alone: Unregistration request initiated by the Node.
REG_UNR_B	1	< 0x3FFF	1	0	• After a REG_UNR_S: Unregistration acknowledge; • Alone: Unregistration request initiated by the Base Node

Fields REG.SNK and REG.AUK are of significance only for REG_RSP and REG_ACK messages with Security Profile 1 (REG.SCP=1). For all other message-exchange variants using the REG control packet, these fields shall not be present reducing the length of payload.

In REG_RSP message, the REG.SNK and REG.AUK shall always be inserted encrypted with WK0.

In the REG_ACK message, the REG.SNK field shall be set to zero. The contents of the REG.AUK field shall be derived by decrypting the received REG_RSP message with WK0 and re-encrypting the decrypted REG.AUK field with SWK derived from the decrypted REG.SNK and random sequence previously received in SEC control packets.

B.4.4.5.4 CON control packet (PKT.CTYPE = 2)

This control packet is used for negotiating the connections. The description of the fields of this packet is given in Table B.10 and Figure B.29 The meaning of the packet differs depending on the direction of the packet and on the values of the different types. Table B.11 shows the different interpretation of the packets. The packets are used during the connection establishment and closing. These processes are explained in more detail in B.4.6.6.

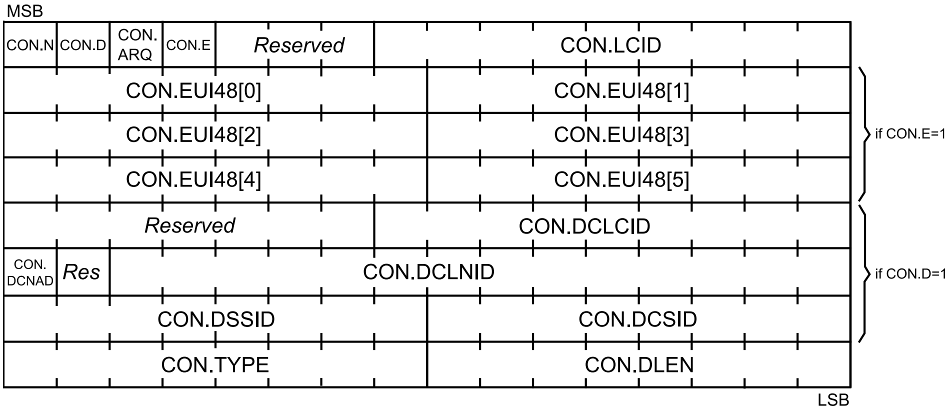


Figure B.29 – CON control packet structure

Note that Figure B.29 shows the complete message with all optional parts. When CON.D is 0, CON.DCNAD, CON.DSSID, CON.DCLNID, CON.DCLID, CON.DCSID and the reserved field between CON.DCNAD and CON.DSSID will not be present in the message. Thus, the message will be 6 octets smaller. Similarly, when CON.E is zero, the field CON.EUI-48 will not be present, making the message 6 octets smaller.

Table B.10 – CON control packet fields

Name	Length	Description
CON.N	1 bit	Negative • CON.N=1 for the negative connection; • CON.N=0 for the positive connection.
CON.D	1 bit	Direct connection • CON.D=1 if information about direct connection is carried by this packet; • CON.D=0 if information about direct connection is not carried by this packet.
CON.ARQ	1 bit	ARQ mechanism enable • CON.ARQ=1 if ARQ mechanism is enabled for this connection; • CON.ARQ=0 if ARQ mechanism is not enabled for this connection.

Table B.10 – CON control packet fields

Name	Length	Description
CON.E	1 bit	EUI-48 presence • CON.E = 1 to have a CON.EUI-48; • CON.E = 0 to not have a CON.EUI-48 so that this connection establishment is for reaching the Base Node CL.
Reserved	3 bits	Reserved for future version of the protocol. This shall be 0 for this version of the protocol.
CON.LCID	9 bits	Local Connection Identifier. The LCID is reserved in the connection request. LCIDs from 0 to 255 are assigned by the connection requests initiated by the Base Node. LCIDs from 256 to 511 are assigned by the connection requests initiated by the local Node. This is the identifier of the connection being managed with this packet. This is not the same as the PKT.LCID of the generic header, which does not exist for control packets.
CON.EUI-48	48 bits (Present if CON.E=1)	EUI-48 of destination/source Service Node/Base Node for connection request. When not performing a directed connection, this field should not be included. When performing a directed connection, it may contain the SNA, indicating that the Base Node Convergence layer should determine the EUI-48. • CON.D = 0, Destination EUI-48; • CON.D = 1, Source EUI-48.
Reserved	7 bits (Present if CON.D=1)	Reserved for future version of the protocol. This shall be 0 for this version of the protocol.
CON.DCLCID	9 bits (Present if CON.D=1)	Direct Connection LCID This field represents the LCID of the connection identifier to which the one being established shall be directly switched.
CON.DCNAD	1 bit (Present if CON.D=1)	Reserved for future version of the protocol. Direct Connection Not Aggregated at Destination This field represents the content of the PKT.NAD field after a direct connection Switch operation.
Reserved	1 bits (Present if CON.D=1)	Reserved for future version of the protocol. This shall be 0 for this version of the protocol.
CON.DCLNID	14 bits (Present if CON.D=1)	Direct Connection LNID This field represents the LNID part of the connection identifier to which the one being established shall be directly switched.
CON.DSSID	8 bits (Present if CON.D=1)	Direct Switch SID This field represents the SID of the Switch that should learn this direct connection and perform direct switching.
CON.DCSID	8 bits (Present if CON.D=1)	Direct Connection SID This field represents the SID part of the connection identifier to which the one being established shall be directly switched.

Table B.10 – CON control packet fields

Name	Length	Description
CON.TYPE	8 bits	Connection type. The connection type (see PRIME Annex B) specifies the Convergence layer to be used for this connection. They are treated transparently through the MAC common part sublayer, and are used only to identify which Convergence layer may be used.
CON.DLEN	8 bits	Length of CON.DATA field in bytes
CON.DATA	*(variable)*	Connection specific parameters. These connections specific parameters are Convergence layer specific. They should be defined in each Convergence layer to define the parameters that are specific to the connection. These parameters are handled in a transparent way by the common part sublayer.

Table B.11 – CON control packet types

Name	HDR.DO	CON.N	Description
CON_REQ_S	0	0	Connection establishment request initiated by the Service Node.
CON_REQ_B	1	0	The Base Node will consider that the connection is established with the identifier CON.LCID. • After a CON_REQ_S: Connection accepted; • Alone: Connection establishment request.
CON_CLS_S	0	1	The Service Node considers this connection closed: • After a CON_REQ_B: Connection rejected by the Node; • After a CON_CLS_B: Connection closing acknowledge; • Alone: Connection closing request.
CON_CLS_B	1	1	The Base Node will consider that the connection is no longer established. • After a CON_REQ_S: Connection establishment rejected by the Base Node; • After a CON_CLS_S: Connection closing acknowledge; • Alone: Connection closing request.

B.4.4.5.5 PRO control packet (PKT.CTYPE = 3)

This control packet is used to promote a Service Node from Terminal function to Switch function. The description of the fields of this packet is given in Table B.12, Figure B.30 and Figure B.31. The meaning of the packet differs depending on the direction of the packet and on the values of the different types. Table B.13 shows the different interpretation of the packets. The promotion process is explained in more detail in B.4.6.3.

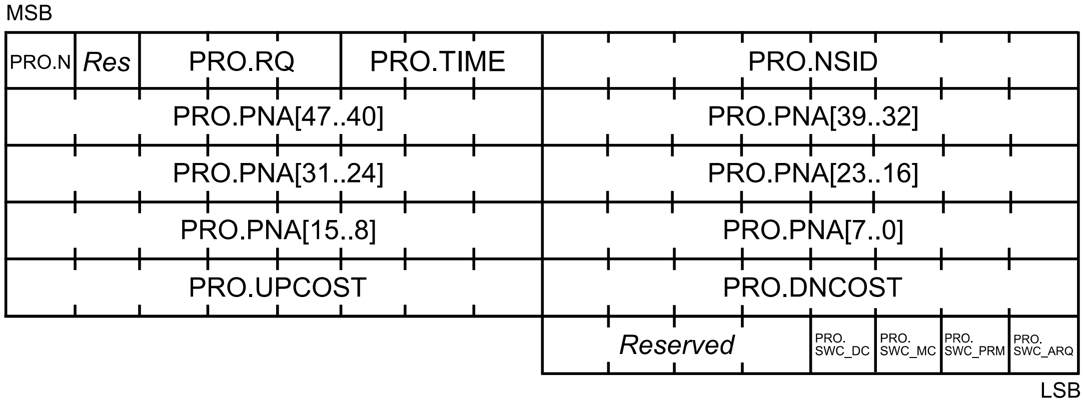


Figure B.30 – PRO_REQ_S control packet structure

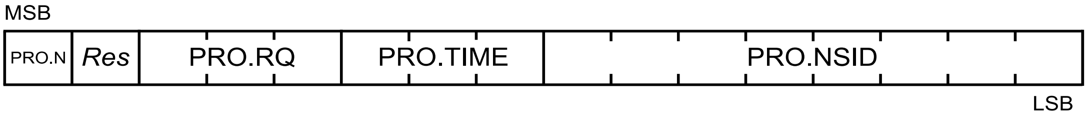


Figure B.31 – PRO control packet structure

Note that Figure B.30 includes all fields as used by a PRO_REQ_S message. All other messages are much smaller, containing only PRO.N, PRO.RC, PRO.TIME and PRO.NSID as shown in Figure B.31.

Table B.12 – PRO control packet fields

Name	Length	Description
PRO.N	1 bit	Negative PRO.N=1 for the negative promotion PRO.N=0 for the positive promotion
Reserved	1 bit	Reserved for future version of this protocol This shall be 0 for this version of the protocol.
PRO.RQ	3 bits	Receive quality of the PNPDU message received from the Service Node requesting the Terminal to promote.
PRO.TIME	3 bits	The ALV.TIME which is being used by the terminal which will become a switch. On a reception of this time in a PRO_REQ_B the Service Node should reset the Keep-Alive timer in the same way as receiving an ALV_B.
PRO.NSID	8 bits	New Switch Identifier. This is the assigned Switch identifier of the Node whose promotion is being managed with this packet. This is not the same as the PKT.SID of the packet header, which must be the SID of the Switch this Node is connected to, as a Terminal Node.
PRO.PNA	0 or 48 bits	Promotion Need Address, contains the EUI-48 of the Terminal requesting the Service Node promotes to become a Switch. This field is only included in the PRO_REQ_S message.
PRO.UPCOST	0 or 8 bits	Total uplink cost from the Terminal Node to the Base Node. This value is calculated in the same way a Switch Node calculates the value it places into its own Beacon PDU. This field is only included in the PRO_REQ_S message.

Table B.12 – PRO control packet fields

Name	Length	Description
PRO.DNCOST	0 or 8 bits	Total Downlink cost from the Base Node to the Terminal Node. This value is calculated in the same way a Switch Node calculates the value it places into its own Beacon PDU. This field is only included in the PRO_REQ_S message.
Reserved	4 bits	Reserved for future versions of the protocol. Should be set to 0 for this version of the protocol.
PRO.SWC_DC	1 bit	Direct Connection Switching Capability 1 if the device is able to behave as Direct Switch in direct connections. 0 otherwise
PRO.SWC_MC	1 bit	Multicast Switching Capability 1 if the device is able to manage the multicast traffic when behaving as a Switch. 0 otherwise
PRO.SWC_PRM	1 bit	PHY Robustness Management Switching Capability 1 if the device is able to perform PRM for the Terminal Nodes when behaving as a Switch. 0 if the device is not able to perform PRM when behaving as a Switch.
PRO.SWC_ARQ	1 bit	ARQ Buffering Switching Capability 1 if the device is able to perform buffering for ARQ connections while switching. 0 if the device is not able to perform buffering for ARQ connections while switching.

Table B.13 – PRO control packet types

Name	HDR.DO	PRO.N	PRO.NSID	Description
PRO_REQ_S	0	0	0xFF	Promotion request initiated by the Service Node.
PRO_REQ_B	1	0	< 0xFF	The Base Node will consider that the Service Node has promoted with the identifier PRO.NSID. • After a PRO_REQ: Promotion accepted; • Alone: Promotion request initiated by the Base Node.
PRO_ACK	0	0	< 0xFF	Promotion acknowledge
PRO_REJ	1	1	0xFF	The Base Node will consider that the Service Node is demoted. It is sent after a PRO_REQ to reject it.
PRO_DEM_S	0	1	< 0xFF	The Service Node considers that it is demoted: • After a PRO_DEM_B: Demotion accepted; • After a PRO_REQ_B: Promotion rejected; • Alone: Demotion request.
PRO_DEM_B	1	1	< 0xFF	The Base Node considers that the Service Node is demoted. • After a PRO_DEM_S: Demotion accepted; • Alone: Demotion request.

B.4.4.5.6 BSI control packet (PKT.CTYPE = 4)

The Beacon Slot Information (BSI) control packet is only used by the Base Node and Switch Nodes. It is used to exchange information that is further used by a Switch Node to transmit its beacon. The description of the fields of this packet is given in Table B.14 and Figure B.32. The meaning of the packet differs depending on the direction of the packet and on the values of the different types. Table B.15 represents the different interpretation of the packets. The promotion process is explained in more detail in B.4.6.3.

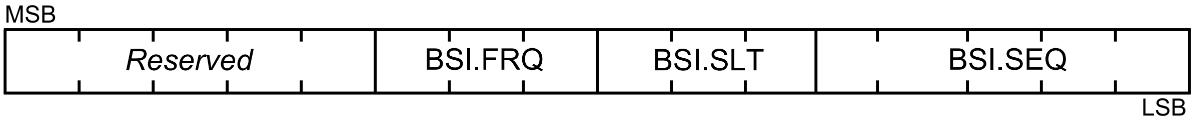


Figure B.32 – BSI control packet structure

Table B.14 – BSI control packet fields

Name	Length	Description
Reserved	5 bits	*Reserved for future version of this protocol. In this version, this field should be initialized to 0.*

BSI.FRQ	3 bits	Transmission frequency of Beacon Slot, encoded as:
		FRQ = 0 => 1 beacon every frame
		FRQ = 1 => 1 beacon every 2 frames
		FRQ = 2 => 1 beacon every 4 frames
		FRQ = 3 => 1 beacon every 8 frames
		FRQ = 4 => 1 beacon every 16 frames
		FRQ = 5 => 1 beacon every 32 frames
		FRQ = 6 => *Reserved*
		FRQ = 7 => *Reserved*

BSI.SLT	3 bits	Beacon Slot to be used by target Switch
BSI.SEQ	5 bits	The Beacon Sequence number when the specified change takes effect.

Table B.15 – BSI control message types

Name	HDR.DO	Description
BSI_ACK	0	Acknowledgement of receipt of BSI control message
BSI_IND	1	Beacon-slot change command

B.4.4.5.7 FRA control packet (PKT.CTYPE = 5)

This control packet is broadcast from the Base Node and relayed by all Switch Nodes to the entire Subnetwork. It is used to circulate information on the change of Frame structure at a specific time in future. The description of fields of this packet is given in Table B.16 and Figure B.33. Table B.17 shows the different interpretation of the packets.

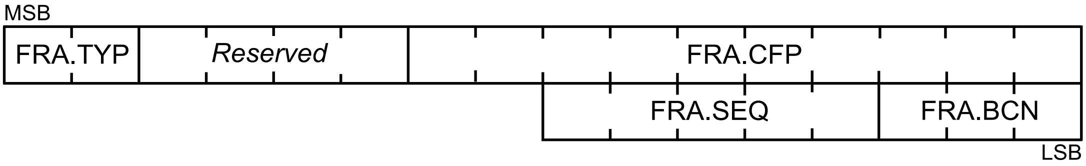


Figure B.33 – FRA control packet structure

Table B.16 – FRA control packet fields

Name	Length	Description
FRA.TYP	2 bits	0: Beacon count change 1: CFP duration change
Reserved	4 bits	Reserved for future version of this protocol. In this version, this field should be initialized to 0.
FRA.CFP	10 bits	Offset of CFP from start of frame
FRA.SEQ	5 bits	The Beacon Sequence number when the specified change takes effect.
FRA.BCN	3 bits	Number of beacons in a frame

Table B.17 – FRA control packet types

Name	FRA.TYP	Description
FRA_BCN_IND	0	Indicates changes to frame structure due to change in beacon-slot count
FRA_CFP_IND	1	Indicates changes to frame structure due to change in CFP duration as a result of grant of CFP or end of CFP period for any requesting Service Node in the subnetwork.

B.4.4.5.8 CFP control packet (PKT.CTYPE = 6)

This control packet is used for dedicated contention-free channel access time allocation to individual Terminal or Switch Nodes. The description of the fields of this packet is given in Table B.18 and Figure B.34. The meaning of the packet differs depending on the direction of the packet and on the values of the different types. Table B.19 represents the different interpretation of the packets.

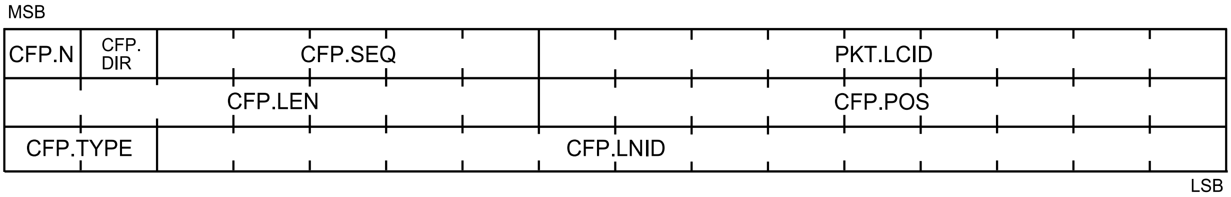


Figure B.34 – CFP control packet structure

Table B.18 – CFP control message fields

Name	Length	Description
CFP.N	1 bit	0: denial of allocation/deallocation request; 1: acceptance of allocation/deallocation request.
CFP.DIR	1 bit	Indicate direction of allocation. 0: allocation is applicable to uplink (towards Base Node) direction; 1: allocation is applicable to Downlink (towards Service Node) direction.
CFP.SEQ	5 bits	The Beacon Sequence number when the specified change takes effect.
CFP.LCID	9 bits	LCID of requesting connection.
CFP.LEN	7 bits	Length (in symbols) of requested/allocated channel time per frame.
CFP.POS	9 bits	Offset (in symbols) of allocated time from beginning of frame.

Table B.18 – CFP control message fields

Name	Length	Description

CFP.TYPE	2 bits	0: Channel allocation packet;
		1: Channel de-allocation packet;
		2: Channel change packet.

CFP.LNID	14 bits	LNID of Service Node that is the intended user of the allocation.

Table B.19 – CFP control packet types

Name	CFP.TYP	HDR.DO	Description
CFP_ALC_REQ_S	0	0	Service Node makes channel allocation request
CFP_ALC_IND	0	1	• After a CFP_ALC_REQ_S: Requested channel is allocated • Alone: Unsolicited channel allocation by Base Node
CFP_ALC_REJ	0	1	Requested channel allocation is denied
CFP_DALC_REQ	1	0	Service Node makes channel de-allocation request
CFP_DALC_RSP	1	1	Base Node confirms de-allocation
CFP_CHG_IND	2	1	Change of location of allocated channel within the CFP.

B.4.4.5.9 ALV control packet (PKT.CTYPE = 7)

The ALV control message is used for Keep-Alive signalling between a Service Node, the Service Nodes above it and the Base Node. The message exchange is bidirectional, that is, a message is periodically exchanged in each direction. The structure of these messages are shown in Figure B.35 and Table B.20. The different Keep-Alive message types are shown in Table B.21. These messages are sent periodically, as described in clause B.4.6.5.

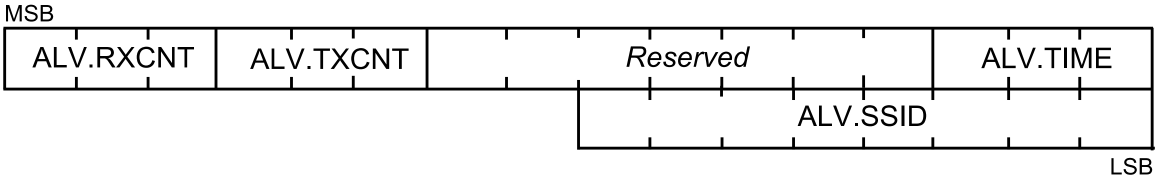


Figure B.35 – ALV Control packet structure

Table B.20 – ALV control message fields

Name	Length	Description
ALV.RXCNT	3 bits	Modulo 8 counter to indicate number of received ALV messages.
ALV.TXCNT	3 bits	Modulo 8 counter to indicate number of transmitted ALV messages.
Reserved	7 bits	Should always be encoded as 0 in this version of the specification.

ALV.TIME	3 bits	Time to wait for an ALV_B messages before assuming the Service Node has been unregistered by the Base Node.
		ALV.TIME = 0 => 32 seconds;
		ALV.TIME = 1 => 64 seconds;
		ALV.TIME = 2 => 128 seconds ~ 2.1 minutes;
		ALV.TIME = 3 => 256 seconds ~ 4.2 minutes;
		ALV.TIME = 4 => 512 seconds ~ 8.5 minutes;
		ALV.TIME = 5 => 1024 seconds ~ 17.1 minutes;
		ALV.TIME = 6 => 2048 seconds ~ 34.1 minutes;
		ALV.TIME = 7 => 4096 seconds ~ 68.3 minutes.

ALV.SSID	8 bits	For a Terminal, this should be 0xFF. For a Switch, this is its Switch Identifier.

Table B.21 – Keep-alive control packet types

Name	HDR.DO	Description
ALV_S	0	Keep-Alive message from a Service Node
ALV_B	1	Keep-Alive message from the Base Node

B.4.4.5.10MUL control packet (PKT.CTYPE = 8)

The MUL message is used to control multicast group membership. The structure of this message and the meanings of the fields are described in Table B.22 and Figure B.36. The message can be used in different ways as described in Table B.23.

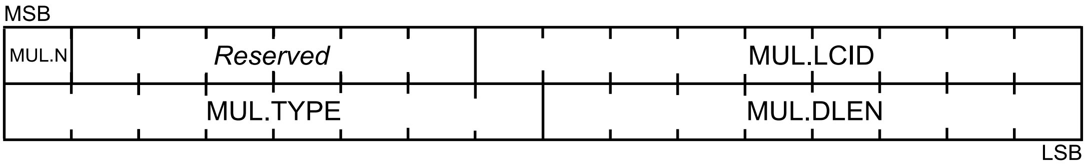


Figure B.36 – MUL control packet structure

Table B.22 – MUL control message fields

Name	Length	Description
MUL.N	1 bit	Negative - MUL.N = 1 for the negative multicast connection, i.e., multicast group leave. - MUL.N = 0 for the positive multicast connection, i.e., multicast group join.
Reserved	6 bits	Reserved for future version of the protocol. This shall be 0 for this version of the protocol.
MUL.LCID	9 bits	Local Connection Identifier. The LCID indicates which multicast distribution group is being managed with this message.
MUL.TYPE	8 bits	Connection type. The connection type specifies the Convergence layer to be used for this connection. They are treated transparently through the MAC common part sublayer, and are used only to identify which Convergence layer may be used. See PRIME Annex B.
MUL.DLEN	8 bits	Length of data in bytes in the MUL.DATA field
MUL.DATA	(variable)	Connection specific parameters. These connections specific parameters are Convergence layer specific. They should be defined in each Convergence layer to define the parameters that are specific to the connection. These parameters are handled in a transparent way by the common part sublayer.

Table B.23 – MUL control message types

Name	HDR.DO	MUL.N	Description
MUL_JOIN_S	0	0	Multicast group join request initiated by the Service Node, or an acknowledgement when sent in response to a MUL_JOIN_B.
MUL_JOIN_B	1	0	The Base Node will consider that the group has been joined with the identifier MUL.LCID. - After a MUL_JOIN_S: join accepted; - Alone: group join request.
MUL_LEAVE_S	0	1	The Service Node leaves the multicast group: - After a MUL_JOIN_B: Join rejected by the Node; - After a MUL_LEAVE_B: group leave acknowledge; - Alone: group leave request.
MUL_LEAVE_B	1	1	The Base Node will consider that the Service Node is no longer a member of the multicast group. - After a MUL_JOIN_S: Group join rejected by the Base Node; - After a MUL_LEAVE_S: Group leave acknowledge; - Alone: Group leave request.

B.4.4.5.11 PRM control packet (PKT.CTYPE = 9)

The PHY Robustness Management packets are used to control the parameters that affect the robustness and efficiency of the PHY. These packets are sent to notify to the peer of the need to improve robustness of the transmission, or to notify the peer that the reception quality is so good that a less robust and so more efficient modulation scheme can be transmitted.

The fields of the PRM control packet are described in Table B.24 and Figure B.37 and the types of messages are described in Table B.25.

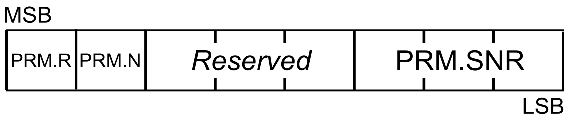


Figure B.37 – PHY control packet structure

Table B.24 – PRM control message fields

Name	Length	Description
PRM.R	1 bit	Response • PRM.R=1 if this message is a response; • PRM.R=0 if this message is a request.
PRM.N	1 bit	Negative • PRM.N=1 if the operation could not be performed; • PRM.N=0 if the operation was performed.
Reserved	3 bits	Should always be encoded as 0 in this version of the specification.
PRM.SNR	3 bits	Indicates the SNR at the end that initiates a change request, obtained using PHY_SNR primitive (clause B.3.10.3.11/ [ITU-T G.9955].)

Table B.25 – PRM control message types

Name	PRM.R	PRM.N	Description
PRM_REQ	0	0	PHY modulation management request.
PRM_ACK	1	0	PHY modulation management acknowledge.
PRM_REJ	1	1	PHY modulation management rejected.

B.4.4.5.12 SEC control packet (PKT.CTYPE = 10)

The SEC control message is broadcast unencrypted by the Base Node and all Switch Nodes to the rest of the Subnetwork in order to circulate the random sequence used to generate working keys. The random sequence used by devices in a Subnetwork is dynamic and changes from time to time to ensure a robust security framework. The structure of this message is shown in Table B.26 and Figure B.38. Further details of security mechanisms are given in clause B.4.3.8.

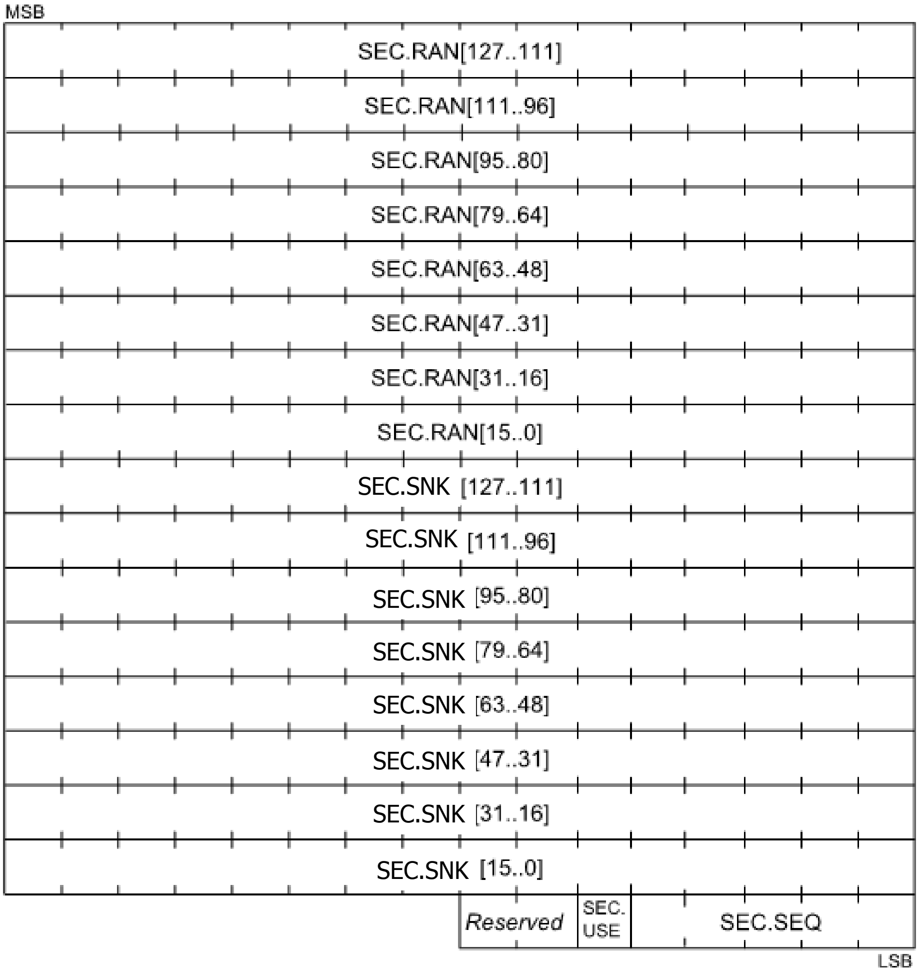


Figure B.38 – SEC control packet structure

Table B.26 – SEC control message fields

Name	Length	Description
SEC.RAN	128 bits	Random sequence to be used to derive WK.
SEC.SNK	128 bits	Random sequence to be used to derive SWK.
Reserved	2 bits	Should always be encoded as 0 in this version of the specification.
SEC.USE	1 bits	When 1 indicates the random sequences are already in use. When 0 indicates that SE.SEQ should be used to determine when to start using these random sequences.
SEC.SEQ	5 bits	The Beacon Sequence number when the specified change takes effect.

B.4.5 MAC service access point

B.4.5.1 General

The MAC service access point provides several primitives to allow the Convergence layer to interact with the MAC layer. This clause aims to explain how the MAC may be used. An implementation of the MAC may not use all the primitives listed here; it may use other primitives; or it may have a function-call based interface rather than message-passing, etc. These are all implementation issues which are beyond the scope of this specification.

The .request primitives are passed from the CL to the MAC to request the initiation of a service. The .indication and .confirm primitives are passed from the MAC to the CL to indicate an internal

MAC event that is significant to the CL. This event may be logically related to a remote service request or may be caused by an event internal to the local MAC. The .response primitive is passed from the CL to the MAC to provide a response to a .indication primitive. Thus, the four primitives are used in pairs, the pair .request and .confirm and the pair .indication and .response. This is shown in Figure B.39, Figure B.40, Figure B.41 and Figure B.42.

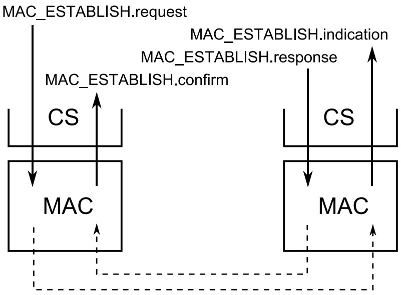


Figure B.39 – Establishment of a connection

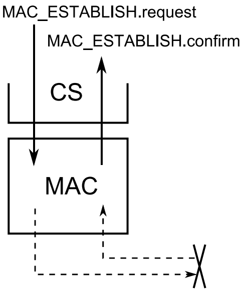


Figure B.40 – Failed establishment of a connection

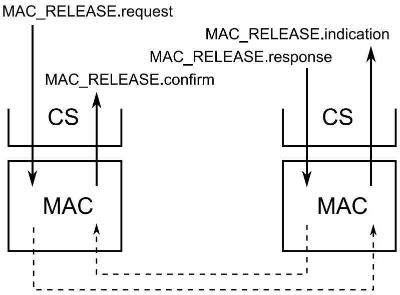


Figure B.41 – Release of a connection

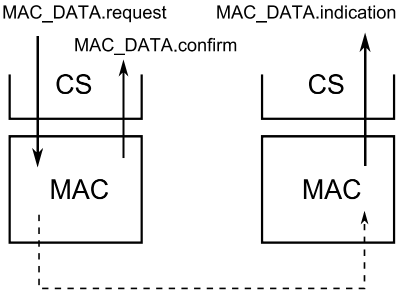


Figure B.42 – Transfer of data

Table B.27 represents the list of available primitives in the MAC-SAP:

Table B.27 – List of MAC primitives

Service node primitives	Base node primitives
MAC_ESTABLISH.request	MAC_ESTABLISH.request
MAC_ESTABLISH.indication	MAC_ESTABLISH.indication
MAC_ESTABLISH.response	MAC_ESTABLISH.response
MAC_ESTABLISH.confirm	MAC_ESTABLISH.confirm
MAC_RELEASE.request	MAC_RELEASE.request
MAC_RELEASE.indication	MAC_RELEASE.indication
MAC_RELEASE.response	MAC_RELEASE.response
MAC_RELEASE.confirm	MAC_RELEASE.confirm
MAC_JOIN.request	MAC_JOIN.request
MAC_JOIN.Response	MAC_JOIN.response
MAC_JOIN.indication	MAC_JOIN.indication
MAC_JOIN.confirm	MAC_JOIN.confirm

Table B.27 – List of MAC primitives

Service node primitives	Base node primitives
MAC_LEAVE.request	MAC_LEAVE.request
MAC_LEAVE.indication	MAC_LEAVE.indication
MAC_LEAVE.confirm	MAC_LEAVE.confirm
MAC_DATA.request	MAC_REDIRECT.response
MAC_DATA.confirm	MAC_DATA.request
MAC_DATA.indication	MAC_DATA.confirm
	MAC_DATA.indication

B.4.5.2 Service node and base node signalling primitives

B.4.5.2.1 General

The following clauses describe primitives which are available in both the Service Node and Base Node MAC-SAP. These are signalling primitives only and used for establishing and releasing MAC connections.

B.4.5.2.2 MAC_ESTABLISH

B.4.5.2.2.1 General

The MAC_ESTABLISH primitives are used to manage a connection establishment.

B.4.5.2.2.2 MAC_ESTABLISH.request

The MAC_ESTABLISH.request primitive is passed to the MAC layer entity to request the connection establishment.

The semantics of this primitive are as follows:

MAC_ESTABLISH.*request*{EUI-48, Type, Data, DataLength, ARQ, CfBytes}

The *EUI-48* parameter of this primitive is used to specify the address of the Node to which this connection will be addressed. The MAC will internally transfer this to an address used by the MAC layer. When the CL of a Service Node wishes to connect to the Base Node, it uses the EUI-48 00:00:00:00:00:00. However, when the CL of a Service Node wishes to connect to another Service Node on the Subnetwork, it uses the EUI-48 of that Service Node. This will then trigger a direct connection establishment. However, whether a normal or a directed connection is established is transparent to the Service Node MAC SAP. As the EUI-48 of the Base Node is the SNA, the connection could also be requested from the Base Node using the SNA.

The *Type* parameter is an identifier used to define the type of the Convergence layer that should be used for this connection (see PRIME Annex B). This parameter is 1 byte long and will be transmitted in the CON.TYPE field of the connection request.

The *Data* parameter is a general purpose buffer to be interchanged for the negotiation between the local CL and the remote CL. This parameter will be transmitted in the CON.DATA field of the connection request.

The *DataLength* parameter is the length of the *Data* parameter in bytes.

The *ARQ* parameter indicates whether or not the ARQ mechanism should be used for this connection. It is a Boolean type with a value of true indicating that ARQ will be used.

The *CfBytes* parameter is used to indicate whether or not the connection should use the contention or contention-free channel access scheme. When *CfBytes* is zero, contention-based access should be used. When *CfBytes* is not zero, it indicates how many bytes per frame should be allocated to the connection using CFP packets.

B.4.5.2.2.3 MAC_ESTABLISH.indication

The MAC_ESTABLISH.indication is passed from the MAC layer to indicate that a connection establishment was initiated by a remote Node.

The semantics of this primitive are as follows:

MAC_ESTABLISH.indication{ConHandle, EUI-48, Type, Data, DataLength, CfBytes}

The *ConHandle* is a unique identifier interchanged to uniquely identify the connection being indicated. It has a valid meaning only in the MAC SAP, used to have a reference to this connection between different primitives.

The *EUI-48* parameter indicates which device on the Subnetwork wishes to establish a connection.

The *Type* parameter is an identifier used to define the type of the Convergence layer that should be used for this connection. This parameter is 1 byte long and it is received in the CON.TYPE field of the connection request.

The *Data* parameter is a general purpose buffer to be interchanged for the negotiation between the remote CL and the local CL. This parameter is received in the CON.DATA field of the connection request.

The *DataLength* parameter is the length of the Data parameter in bytes.

The *CfBytes* parameter is used to indicate if the connection should use the contention or contention-free channel access scheme. When *CfBytes* is zero, contention-based access will be used. When *CfBytes* is not zero, it indicates how many bytes per frame the connection would like to be allocated.

B.4.5.2.2.4 MAC_ESTABLISH.response

The MAC_ESTABLISH.response is passed to the MAC layer to respond with a MAC_ESTABLISH.indication.

The semantics of this primitive are as follows:

MAC_ESTABLISH.response{ConHandle, Answer, Data, DataLength}

The *ConHandle* parameter is the same as the one that was received in the MAC_ESTABLISH.indication.

The *Answer* parameter is used to notify the MAC of the action to be taken for this connection establishment. This parameter may have one of the values in Table B.28.

The *Data* parameter is a general purpose buffer to be interchanged for the negotiation between the remote CL and the local CL. This parameter is received in the CON.DATA field of the connection response.

The *DataLength* parameter is the length of the Data parameter in bytes.

Data may be passed to the caller even when the connection is rejected, i.e., Answer has the value 1. The data may then optionally contain more information as to why the connection was rejected.

Table B.28 – Values of the *Answer* parameter in MAC_ESTABLISH.response primitive

Answer	Description
Accept = 0	The connection establishment is accepted.
Reject = 1	The connection establishment is rejected.

B.4.5.2.2.5 MAC_ESTABLISH.confirm

The MAC_ESTABLISH.confirm is passed from the MAC layer as the remote answer to a MAC_ESTABLISH.request.

The semantics of this primitive are as follows:

MAC_ESTABLISH.confirm{ConHandle, Result, EUI-48, Type, Data, DataLength}

The *ConHandle* is a unique identifier to uniquely identify the connection being indicated. It has a valid meaning only in the MAC SAP, used to have a reference to this connection between different primitives. The value is only valid if the *Result* parameter is 0.

The *Result* parameter indicates the result of the connection establishment process. It may have one of the values in Table B.29.

The *EUI-48* parameter indicates which device on the Subnetwork wishes to establish a connection.

The *Type* parameter is an identifier used to define the type of the Convergence layer that should be used for this connection. This parameter is 1 byte long and it is received in the CON.TYPE field of the connection request

The *Data* parameter is a general purpose buffer to be interchanged for the negotiation between the remote CL and the local CL. This parameter is received in the CON.DATA field of the connection response.

The *DataLength* parameter is the length of the Data parameter in bytes.

Data may be passed to the caller even when the connection is rejected, i.e., Result has the value 1. The data may then optionally contain more information as to why the connection was rejected.

Table B.29 – Values of the *Result* parameter in MAC_ESTABLISH.confirm primitive

Result	Description
Success = 0	The connection establishment was successful.
Reject = 1	The connection establishment failed because it was rejected by the remote Node.
Timeout = 2	The connection establishment process timed out.
No bandwidth = 3	There is insufficient available bandwidth to accept this contention-free connection.
No Such Device = 4	A device with the destination address cannot be found.
Redirect failed =5	The Base Node attempted to perform a redirect which failed.
Not Registered = 6	The Service Node is not registered.
No More LCIDs = 7	All available LCIDs have been allocated.

B.4.5.2.3 MAC_RELEASE

B.4.5.2.3.1 General

The MAC_RELEASE primitives are used to release a connection.

B.4.5.2.3.2 MAC_RELEASE.request

The MAC_RELEASE.request is a primitive used to initiate the release process of a connection.

The semantics of this primitive are as follows:

MAC_RELEASE.request{ConHandle}

The *ConHandle* parameter specifies the connection to be released. This handle is the one that was obtained during the MAC_ESTABLISH primitives.

B.4.5.2.3.3 MAC_RELEASE.indication

The MAC_RELEASE.indication is a primitive used to indicate that a connection is being released. It may be released because of a remote operation or because of a connectivity problem.

The semantics of this primitive are as follows:

MAC_RELEASE.indication{ConHandle, Reason}

The *ConHandle* parameter specifies the connection being released. This handle is the one that was obtained during the MAC_ESTABLISH primitives.

The *Reason* parameter may have one of the values given in Table B.30.

Table B.30 – Values of the *Reason* parameter in MAC_RELEASE.indication primitive

Reason	**Description**
Success = 0	The connection release was initiated by a remote service.
Error = 1	The connection was released because of a connectivity problem.

B.4.5.2.3.4 MAC_RELEASE.response

The MAC_RELEASE.response is a primitive used to respond to a connection release process.

The semantics of this primitive are as follows:

MAC_RELEASE.response{ConHandle, Answer}

The *ConHandle* parameter specifies the connection being released. This handle is the one that was obtained during the MAC_ESTABLISH primitives.

The *Answer* parameter may have one of the values given in Table B.31 This parameter may not have the value "*Reject* = 1" because a connection release process cannot be rejected.

Table B.31 – Values of the *Answer* parameter in MAC_RELEASE.response primitive

Answer	**Description**
Accept = 0	The connection release is accepted.

After sending the MAC_RELEASE.response the ConHandle is no longer valid and should not be used.

B.4.5.2.3.5 MAC_RELEASE.confirm

The MAC_RELEASE.confirm primitive is used to confirm that the connection release process has finished.

The semantics of this primitive are as follows:

MAC_RELEASE.confirm{ConHandle, Result}

The *ConHandle* parameter specifies the connection released. This handle is the one that was obtained during the MAC_ESTABLISH primitives.

The *Result* parameter may have one of the values given in Table B.32

Table B.32 – Values of the *Result* parameter in MAC_RELEASE.confirm primitive

Result	**Description**
Success = 0	The connection release was successful.
Timeout = 2	The connection release process timed out.
Not Registered = 6	The Service Node is no longer registered.

After the reception of the MAC_RELEASE.confirm the ConHandle is no longer valid and should not be used.

B.4.5.2.4 MAC_JOIN

B.4.5.2.4.1 General

The MAC_JOIN primitives are used to join to a broadcast or multicast connection and allow the reception of such packets.

B.4.5.2.4.2 MAC_JOIN.request

The MAC_JOIN.request primitive is used:

- By all Nodes: to join broadcast traffic of a specific CL and start receiving these packets
- By Service Nodes: to join a particular multicast group
- By Base Node: to invite a Service Node to join a particular multicast group.

Depending on which device makes the join-request, this SAP can have two different variants. First variant shall be used on Base Nodes and second on Service Nodes:

The semantics of this primitive are as follows:

MAC_JOIN.request{Broadcast, ConHandle, EUI-48, Type, Data, DataLength}

MAC_JOIN.request(Broadcast, Type, Data, Datalength}

The *Broadcast* parameter specifies whether the JOIN operation is being performed for a broadcast connection or for a multicast operation. It should be 1 for a broadcast operation and 0 for a multicast operation. In case of broadcast operation, EUI-48, Data, DataLength are not used.

ConHandle indicates the handle to be used with for this multicast join. In case of first join request for a new multicast group, ConHandle will be set to 0. For any subsequent EUI additions to an existing multicast group, ConHandle will serve as index to respective multicast group.

The EUI-48 parameter is used by the Base Node to specify the address of the Node to which this join request will be addressed. The MAC will internally transfer this to an address used by the MAC layer. When the CL of a Service Node initiates the request, it uses the EUI-48 00:00:00:00:00:00.

The Type parameter defines the type of the Convergence layer that will send/receive the data packets. This parameter is 1 byte long and will be transmitted in the MUL.TYPE field of the join request.

The Data parameter is a general purpose buffer to be interchanged for the negotiation between the remote CL and the local CL. This parameter is received in the MUL.DATA field of the connection request. In case the CL supports several multicast groups, this Data parameter will be used to uniquely identify the group

The DataLength parameter is the length of the Data parameter in bytes.

If Broadcast is 1, the MAC will immediately issue a MAC_JOIN.confirm primitive since it does not need to perform any end-to-end operation. For a multicast operation the MAC_JOIN.confirm is only sent once signalling with the uplink Service Node/Base Node is complete.

B.4.5.2.4.3 MAC_JOIN.confirm

The MAC_JOIN.confirm primitive is received to confirm that the MAC_JOIN.request operation has finished.

The semantics of this primitive are as follows:

MAC_JOIN.confirm{ConHandle, Result}

The *ConHandle* is a unique identifier to uniquely identify the connection being indicated. It has a valid meaning only in the MAC SAP, used to have a reference to this connection between different primitives. The value is only valid if the *Result* parameter is 0. When the MAC receives packets on this connection, they will be passed upwards using the MAC_DATA.indication primitive with this *ConHandle*.

The Result parameter indicates the result of multicast group join process. It may have one of the values given in Table B.33.

Table B.33 – Values of the *Result* parameter in MAC_JOIN.confirm primitive

Result	**Description**
Success = 0	The connection establishment was successful.
Reject = 1	The connection establishment failed because it was rejected by the upstream Service Node/Base Node.
Timeout = 2	The connection establishment process timed out.

B.4.5.2.4.4 MAC_JOIN.indication

On the Base Node, the MAC_JOIN.indication is passed from the MAC layer to indicate that a multicast group join was initiated by a Service Node. On a Service Node, it is used to indicate that the Base Node is inviting to join a multicast group.

Depending on device type, this primitive shall have two variants. The first variant below shall be used in Base Nodes and the second variant is for Service Nodes:

MAC_JOIN.indication{ConHandle, EUI-48, Type, Data, DataLength}

MAC_JOIN.indication(ConHandle, Type, Data, Datalength}

The *ConHandle* is a unique identifier interchanged to uniquely identify the multicast group being indicated. It has a valid meaning only in the MAC SAP, used to have a reference to this connection between different primitives.

The *EUI-48* parameter indicates which device on the Subnetwork wishes to establish a connection.

The *Type* parameter is an identifier used to define the type of the Convergence layer that should be used for this request. This parameter is 1 byte long and it is received in the MUL.TYPE field of the connection request.

The *Data* parameter is a general purpose buffer to be interchanged for the negotiation between the remote CL and the local CL. This parameter is received in the MUL.DATA field of the connection request.

The *DataLength* parameter is the length of the Data parameter in bytes.

B.4.5.2.4.5 MAC_JOIN.response

The MAC_JOIN.response is passed to the MAC layer to respond with a MAC_JOIN.indication. Depending on device type, this primitive could have either of the two forms given below. The first one shall be used in Service Node and the second on in Base Node implementations.

The semantics of this primitive are as follows:

MAC_JOIN.response{ConHandle, Answer}

MAC_JOIN.response (ConHandle, EUI, Answer)

The *ConHandle* parameter is the same as the one that was received in the MAC_JOIN.indication.

EUI is the EUI-48 of Service Node that requested the multicast group join.

The *Answer* parameter is used to notify the MAC of the action to be taken for this join request. This parameter may have one of the values depicted below.

Table B.34 – Values of the *Answer* parameter in MAC_ESTABLISH.response primitive

Answer	Description
Accept = 0	The multicast group join is accepted.
Reject = 1	The multicast group join is rejected.

B.4.5.2.5 MAC_LEAVE

B.4.5.2.5.1 General

The MAC_LEAVE primitives are used to leave a broadcast or multicast connection.

B.4.5.2.5.2 MAC_LEAVE.request

The MAC_LEAVE.request primitive is used to leave a multicast or broadcast traffic. Depending on device type, this primitive could have either of the two forms given below. The first one shall be used in Service Node and the second on in Base Node implementations.

The semantics of this primitive are as follows:

MAC_LEAVE.request{ConHandle}

MAC_LEAVE.request{ConHandle, EUI}

The *ConHandle* parameter specifies the connection to be left. This handle is the one that was obtained during the MAC_JOIN primitives.

EUI is the EUI-48 of Service Node to remove from multicast group.

B.4.5.2.5.3 MAC_LEAVE.confirm

The MAC_LEAVE.confirm primitive is received to confirm that the MAC_LEAVE.request operation has finished.

The semantics of this primitive are as follows:

MAC_LEAVE.confirm{ConHandle, Result}

The *ConHandle* parameter specifies the connection released. This handle is the one that was obtained during the MAC_JOIN primitives.

The *Result* parameter may have one of the values in Table B.35.

Table B.35 – Values of the *Result* parameter in MAC_LEAVE.confirm primitive

Result	Description
Success = 0	The connection leave was successful.
Timeout = 2	The connection leave process timed out.

After the reception of the MAC_LEAVE.confirm, the ConHandle is no longer valid and should not be used.

B.4.5.2.5.4 MAC_LEAVE.indication

The MAC_LEAVE.indication primitive is used to leave a multicast or broadcast traffic. Depending on device type, this primitive could have either of the two forms given below. The first one shall be used in Service Node and the second on in Base Node implementations.

The semantics of this primitive are as follows:

MAC_LEAVE.indication{ConHandle}

MAC_LEAVE.indication{ConHandle, EUI}

The ConHandle parameter is the same as that received in MAC_JOIN.confirm or MAC_JOIN.indication. This handle is the one that was obtained during the MAC_JOIN primitives.

EUI is the EUI-48 of Service Node to remove from multicast group.

B.4.5.3 Base node signalling primitives

B.4.5.3.1 General

This clause specifies MAC-SAP primitives that are only available in the Base Node.

B.4.5.3.2 MAC_REDIRECT.response

The MAC_REDIRECT.response primitive is used to answer to a MAC_ESTABLISH.indication and redirects the connection from the Base Node to another Service Node on the Subnetwork.

The semantics of this primitive are as follows:

MAC_REDIRECT.reponse{ConHandle, EUI-48, Data, DateLength}

The *ConHandle* is the one passed in the MAC_ESTABLISH.indication primitive to which it is replying.

EUI-48 indicates the Service Node to which this connection establishment should be forwarded. The Base Node should perform a direct connection setup between the source of the connection establishment and the Service Node indicated by *EUI-48*.

The *Data* parameter is a general purpose buffer to be interchanged for the negotiation between the remote CL and the Base Node CL. This parameter is received in the CON.DATA field of the connection request.

The *DataLength* parameter is the length of the Data parameter in bytes.

Once this primitive has been used, the ConHandle is no longer valid.

B.4.5.4 Service and base nodes data primitives

B.4.5.4.1 General

The following clauses describe how a Service Node or Base Node passes data between the Convergence layer and the MAC layer.

B.4.5.4.2 MAC_DATA.request

The MAC_DATA.request primitive is used to initiate the transmission process of data over a connection.

The semantics of the primitive are as follows:

MAC_DATA.request{ConHandle, Data, DataLength, Priority}

The *ConHandle* parameter specifies the connection to be used for the data transmission. This handle is the one that was obtained during the connection establishment primitives.

The *Data* parameter is a buffer of octets that contains the CL data to be transmitted through this connection.

The *DataLength* parameter is the length of the *Data* parameter in octets.

Priority indicates the priority of the data to be sent when using the CSMA access scheme, i.e., the parameter only has meaning when the connection was established with CfBytes = 0.

B.4.5.4.3 MAC_DATA.confirm

The MAC_DATA.confirm primitive is used to confirm that the transmission process of the data has completed.

The semantics of the primitive are as follows:

MAC_DATA.confirm{ConHandle, Data, Result}

The *ConHandle* parameter specifies the connection that was used for the data transmission. This handle is the one that was obtained during the connection establishment primitives.

The *Data* parameter is a buffer of octets that contains the CL data that where to be transmitted through this connection.

The Result parameter indicates the result of the transmission. This can take one of the values given in Table B.36.

Table B.36 – Values of the *Result* parameter in MAC_DATA.confirm primitive

Result	Description
Success = 0	The send was successful.
Timeout = 2	The send process timed out.

B.4.5.4.4 MAC_DATA.indication

The MAC_DATA.indication primitive notifies the reception of data through a connection to the CL.

The semantics of the primitive are as follows:

MAC_DATA.indication{ConHandle, Data, DataLength}

The *ConHandle* parameter specifies the connection where the data was received. This handle is the one that was obtained during the connection establishment primitives.

The *Data* parameter is a buffer of octets that contains the CL data received through this connection.

The *DataLength* parameter is the length of the *Data* parameter in octets.

B.4.5.5 MAC layer management entity SAPs

B.4.5.5.1 General

The following primitives are all optional.

The aim is to allow an external management entity to control Registration and Promotion of the Service Node, demotion and Unregistration of a Service Node. The MAC layer would normally perform this automatically; however, in some situations/applications it could be advantageous if this could be externally controlled. Indications are also defined so that an external entity can monitor the status of the MAC.

B.4.5.5.2 MLME_REGISTER

B.4.5.5.2.1 General

The MLME_REGISTER primitives are used to perform Registration and to indicate when Registration has been performed.

B.4.5.5.2.2 MLME_REGISTER.request

The MLME_REGISTER.request primitive is used to trigger the Registration process to a Subnetwork through a specific Switch Node. This primitive may be used for enforcing the selection of a specific Switch Node that may not necessarily be used if the selection is left automatic. The Base Node MLME function does not export this primitive.

The semantics of the primitive could be either of the following:

MLME_REGISTER.request{ }

Invoking this primitive without any parameter simply invokes the Registration process in MAC and leaves the selection of the Subnetwork and Switch Node to MAC algorithms. Using this primitive enables the MAC to perform fully automatic Registration if such a mode is implemented in the MAC.

MLME_REGISTER.request{SNA}

The SNA parameter specifies the Subnetwork to which Registration should be performed. Invoking the primitive in this format commands the MAC to register only to the specified Subnetwork.

MLME_REGISTER.request{SID}

The *SID* parameter is the SID (Switch Identifier) of the Switch Node through which Registration needs to be performed. Invoking the primitive in this format commands the MAC to register only to the specified Switch Node.

B.4.5.5.2.3 MLME_REGISTER.confirm

The MLME_REGISTER.confirm primitive is used to confirm the status of completion of the Registration process that was initiated by an earlier invocation of the corresponding request primitive. The Base Node MLME function does not export this primitive.

The semantics of the primitive are as follows:

MLME_REGISTER.confirm{Result, SNA, SID}

The *Result* parameter indicates the result of the Registration. This can take one of the values given in Table B.37.

Table B.37 – Values of the *Result* parameter in MLME_REGISTER.confirm primitive

Result	Description
Done = 0	Registration to specified SNA through specified Switch is completed successfully.
Timeout = 2	Registration request timed out.
Rejected = 1	Registration request is rejected by Base Node of specified SNA.
NoSNA = 8	Specified SNA is not within range.
NoSwitch = 9	Switch Node with specified EUI-48 is not within range.

The *SNA* parameter specifies the Subnetwork to which Registration is performed. This parameter is of significance only if *Result=0.*

The *SID* parameter is the SID (Switch Identifier) of the Switch Node through which Registration is performed. This parameter is of significance only if *Result=0.*

B.4.5.5.2.4 MLME_REGISTER.indication

The MLME_REGISTER.indication primitive is used to indicate a status change in the MAC. The Service Node is now registered to a Subnetwork.

The semantics of the primitive are as follows:

MLME_REGISTER.indication{SNA, SID}

The *SNA* parameter specifies the Subnetwork to which Registration is performed.

The *SID* parameter is the SID (Switch Identifier) of the Switch Node through which Registration is performed.

B.4.5.5.3 MLME_UNREGISTER

B.4.5.5.3.1 General

The MLME_UNREGISTER primitives are used to perform deregistration and to indicate when deregistration has been performed.

B.4.5.5.3.2 MLME_UNREGISTER.request

The MLME_UNREGISTER.request primitive is used to trigger the Unregistration process. This primitive may be used by management entities if they require the Node to unregister for some reason (e.g., register through another Switch Node or to another Subnetwork). The Base Node MLME function does not export this primitive.

The semantics of the primitive are as follows:

MLME_UNREGISTER.request{}

B.4.5.5.3.3 MLME_UNREGISTER.confirm

The MLME_UNREGISTER.confirm primitive is used to confirm the status of completion of the unregister process initiated by an earlier invocation of the corresponding request primitive. The Base Node MLME function does not export this primitive.

The semantics of the primitive are as follows:

MLME_UNREGISTER.confirm{Result}

The *Result* parameter indicates the result of the Registration. This can take one of the values given in Table B.38.

Table B.38 – Values of the *Result* parameter in MLME_UNREGISTER.confirm primitive

Result	Description
Done = 0	Unregister process completed successfully.
Timeout = 2	Unregister process timed out.
Redundant = 10	The Node is already in *Disconnected* functional state and does not need to unregister.

On generation of MLME_UNREGISTER.confirm, the MAC layer shall not perform any automatic actions that may invoke the Registration process again. In such cases, it is up to the management entity to restart the MAC functionality with appropriate MLME_REGISTER primitives.

B.4.5.5.3.4 MLME_UNREGISTER.indication

The MLME_UNREGISTER.indication primitive is used to indicate a status change in the MAC. The Service Node is no longer registered to a Subnetwork.

The semantics of the primitive are as follows:

MLME_UNREGISTER.indication{}

B.4.5.5.4 MLME_PROMOTE

B.4.5.5.4.1 General

The MLME_PROMOTE primitives are used to perform promotion and to indicate when promotion has been performed.

B.4.5.5.4.2 MLME_PROMOTE.request

The MLME_PROMOTE.request primitive is used to trigger the promotion process in a Service Node that is in a *Terminal* functional state. This primitive may be used by management entities to enforce promotion in cases where the Node's default functionality does not automatically start the process. Implementations may use such triggered promotions to improve the Subnetwork topology from time to time.

The semantics of the primitive are as follows:

MLME_PROMOTE.request{}

The value of PRO.PNA in the promotion message sent to the Base Node is undefined and implementation-specific.

B.4.5.5.4.3 MLME_PROMOTE.confirm

The MLME_PROMOTE.confirm primitive is used to confirm the status of completion of a promotion process that was initiated by an earlier invocation of the corresponding request primitive.

The semantics of the primitive are as follows:

MLME_PROMOTE.confirm{Result}

The *Result* parameter indicates the result of the Registration. This can take one of the values given in Table B.39.

Table B.39 – Values of the *Result* parameter in MLME_PROMOTE.confirm primitive

Result	Description
Done = 0	Node is promoted to Switch function successfully.
Timeout = 1	Promotion process timed out.
Rejected = 2	The Base Node rejected promotion request.
Redundant = 10	This device is already functioning as Switch Node.

B.4.5.5.4.4 MLME_PROMOTE.indication

The MLME_PROMOTE.indication primitive is used to indicate a status change in the MAC. The Service NodeService Node is now operating as a Switch.

The semantics of the primitive are as follows:

MLME_PROMOTE.indication{}

B.4.5.5.5 MLME_DEMOTE

B.4.5.5.5.1 General

The MLME_DEMOTE primitives are used to perform demotion and to indicate when demotion has been performed.

B.4.5.5.5.2 MLME_DEMOTE.request

The MLME_DEMOTE.request primitive is used to trigger a demotion process in a Service NodeService Node that is in a *Switch* functional state. This primitive may be used by management entities to enforce demotion in cases where the Node's default functionality does not automatically perform the process.

The semantics of the primitive are as follows:

MLME_DEMOTE.request{}

B.4.5.5.5.3 MLME_DEMOTE.confirm

The MLME_DEMOTE.confirm primitive is used to confirm the status of completion of a demotion process that was initiated by an earlier invocation of the corresponding request primitive.

The semantics of the primitive are as follows:

MLME_DEMOTE.confirm{Result}

The *Result* parameter indicates the result of the demotion. This can take one of the values given in Table B.40.

Table B.40 – Values of the *Result* parameter in MLME_DEMOTE.confirm primitive

Result	Description
Done = 0	Node is demoted to Terminal function successfully.
Timeout = 1	Demotion process timed out.
Redundant = 10	This device is already functioning as Terminal Node.

When a demotion has been triggered using the MLME_DEMOTE.request, the Terminal will remain demoted.

B.4.5.5.5.4 MLME_DEMOTE.indication

The MLME_DEMOTE.indication primitive is used to indicate a status change in the MAC. The Service NodeService Node is now operating as a Terminal.

The semantics of the primitive are as follows:

MLME_DEMOTE.indication{}

B.4.5.5.6 MLME_RESET

B.4.5.5.6.1 General

The MLME_RESET primitives are used to reset the MAC into a known good status.

B.4.5.5.6.2 MLME_RESET.request

The MLME_RESET.request primitive results in the flushing of all transmit and receive buffers and the resetting of all state variables. As a result of invoking of this primitive, a Service Node will transit from its present functional state to the *Disconnected* functional state.

The semantics of the primitive are as follows:

MLME_RESET.request{}

B.4.5.5.6.3 MLME_RESET.confirm

The MLME_RESET.confirm primitive is used to confirm the status of completion of a reset process that was initiated by an earlier invocation of the corresponding request primitive. On the successful completion of the reset process, the MAC entity shall restart all functions starting from the search for a subnetwork (B.4.3.1).

The semantics of the primitive are as follows:

MLME_RESET.confirm{Result}

The *Result* parameter indicates the result of the reset. This can take one of the values given below.

Table B.41 – Values of the *Result* parameter in MLME_RESET.confirm primitive

Result	**Description**
Done = 0	MAC reset completed successfully.
Failed = 1	MAC reset failed due to internal implementation reasons.

B.4.5.5.7 MLME_GET

B.4.5.5.7.1 General

The MLME_GET primitives are used to retrieve individual values from the MAC, such as statistics.

B.4.5.5.7.2 MLME_GET.request

The MLME_GET.request queries information about a given PIB attribute.

The semantics of the primitive are as follows:

MLME_GET.request{PIBAttribute}

The *PIBAttribute* parameter identifies specific attributes as listed in the *Id* fields of tables that list PIB attributes (clause B.6.2.3).

B.4.5.5.7.3 MLME_GET.confirm

The MLME_GET.confirm primitive is generated in response to the corresponding MLME_GET.request primitive.

The semantics of this primitive are as follows:

MLME_GET.confirm{status, PIBAttribute, PIBAttributeValue}

The *status* parameter reports the result of requested information and can have one of the values given in Table B.42.

Table B.42 – Values of the *status* parameter in MLME_GET.confirm primitive

Result	**Description**
Done = 0	Parameter read successfully.
Failed = 1	Parameter read failed due to internal implementation reasons.
BadAttr = 11	Specified *PIBAttribute* is not supported.

The *PIBAttribute* parameter identifies specific attributes as listed in *Id* fields of tables that list PIB attributes (clause B.6.2.3.5).

The *PIBAttributeValue* parameter specifies the value associated with a given *PIBAttribute*

B.4.5.5.8 MLME_LIST_GET

B.4.5.5.8.1 General

The MLME_LIST_GET primitives are used to retrieve a list of values from the MAC.

B.4.5.5.8.2 MLME_LIST_GET.request

The MLME_LIST_GET.request queries for a list of values pertaining to a specific class. This special class of PIB attributes are listed in clause B.6.2.3.5.

The semantics of the primitive are as follows:

MLME_LIST_GET.request{PIBListAttribute}

The *PIBListAttribute* parameter identifies a specific list that is requested by the management entity. The possible values of *PIBListAttribute* are listed in B.6.2.3.5.

B.4.5.5.8.3 MLME_LIST_GET.confirm

The MLME_LIST_GET.confirm primitive is generated in response to the corresponding MLME_LIST_GET.request primitive.

The semantics of this primitive are as follows:

MLME_LIST_GET.confirm{status, PIBListAttribute, PIBListAttributeValue}

The *status* parameter reports the result of requested information and can have one of the values given in Table B.43

Table B.43 – Values of the *status* parameter in MLME_LIST_GET.confirm primitive

Result	**Description**
Done = 0	Parameter read successfully.
Failed = 1	Parameter read failed due to internal implementation reasons.
BadAttr = 11	Specified *PIBListAttribute* is not supported.

The *PIBListAttribute* parameter identifies a specific list as listed in the *Id* field of Table B.90.

The *PIBListAttributeValue* parameter contains the actual listing associated with a given *PIBListAttribute*.

B.4.5.5.9 MLME_SET

B.4.5.5.9.1 General

The MLME_SET primitives are used to set configuration values in the MAC.

B.4.5.5.9.2 MLME_SET.request

The MLME_SET.requests information about a given PIB attribute.

The semantics of the primitive are as follows:

MLME_SET.request{PIBAttribute, PIBAttributeValue}

The *PIBAttribute* parameter identifies a specific attribute as listed in the *Id* fields of tables that list PIB attributes (clause B.6.2.3).

The *PIBAttributeValue* parameter specifies the value associated with given *PIBAttribute.*

B.4.5.5.9.3 MLME_SET.confirm

The MLME_SET.confirm primitive is generated in response to the corresponding MLME_SET.request primitive.

The semantics of this primitive are as follows:

MLME_SET.confirm{result}

The *result* parameter reports the result of requested information and can have one of the values given in Table B.44

Table B.44 – Values of the *Result* parameter in MLME_SET.confirm primitive

Result	**Description**
Done = 0	Given value successfully set for specified attribute.
Failed = 1	Failed to set the given value for specified attribute.
BadAttr = 11	Specified *PIBAttribute* is not supported.
OutofRange = 12	Specified *PIBAttributeValue* is out of acceptable range.
ReadOnly = 13	Specified PIBAttributeValue is read only.

The *PIBAttribute* parameter identifies a specific attribute as listed in the *Id* fields of tables that list PIB attributes (clause B.6.2.3).

The *PIBAttributeValue* parameter specifies the value associated with a given *PIBAttribute.*

B.4.6 MAC procedures

B.4.6.1 Registration

The initial Service Node start-up (B.4.3.1) is followed by a Registration process. A Service Node in a *Disconnected* functional state shall transmit a REG control packet to the Base Node in order to get itself included in the Subnetwork. Since no LNID or SID is allocated to a Service Node at this stage, the PKT.LNID field shall be set to all 1s and the PKT.SID field shall contain the SID of the Switch Node through which it seeks attachment to the Subnetwork.

Base Nodes may use a Registration request as an authentication mechanism. However this specification does not recommend or forbid any specific authentication mechanism and leaves this choice to implementations.

For all successfully accepted Registration requests, the Base Node shall allocate an LNID that is unique within the domain of the Switch Node through which the attachment is realized. This LNID shall be indicated in the PKT.LNID field of response (REG_RSP). The assigned LNID, in combination with the SID of the Switch Node through which the Service Node is registered, would form the NID of the registering Node.

Registration is a three-way process. The REG_RSP shall be acknowledged by the receiving Service Node with a REG_ACK message.

Figure B.43 represents a successful Registration process and Figure B.44 shows a Registration request that is rejected by the Base Node. Details on specific fields that distinguish one Registration message from the other are given in Table B.9.

The REG control packet, in all its usage variants, is transmitted unencrypted, but specified fields (REG.SNK and REG.AUK) are encrypted with context-specific encryption keys as explained in clause B.4.4.5.3. The encryption of REG.AUK in REG_RSP, its decryption at the receiving end and subsequent encrypted retransmission using a different encryption key authenticates that the REG_ACK is from the intended destination.

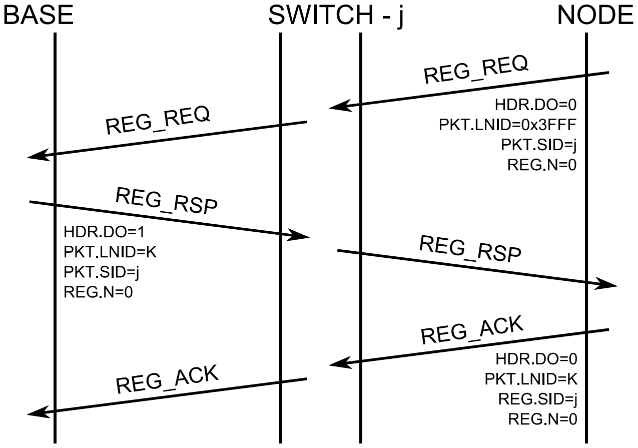


Figure B.43 – Registration process accepted

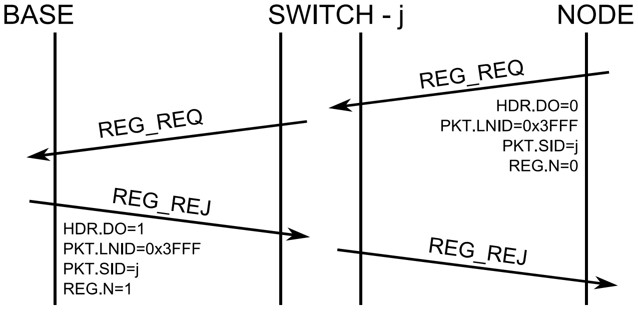


Figure B.44 – Registration process rejected

When assigning an LNID, the Base Node shall not reuse an LNID released by an unregister process until after (*macMaxCtlReTx +1) * macCtlReTxTimer*. seconds, to ensure that all retransmit packets have left the Subnetwork. Similarly, the Base Node shall not reuse an LNID freed by the Keep-Alive process until T_{keep_alive} seconds have passed, using the last known acknowledged T_{keep_alive} value, or if larger, the last unacknowledged $T_{keep\_alive,}$ for the Service Node using the LNID.

During network startup where the whole network is powered on at once, there will be considerable contention for the medium. It is recommended, but optional, that randomness is added to the first transmission of REQ_REQ and all subsequent retransmissions. A random delay of maximum duration of 10% of *macCtlReTxTimer* may be imposed before the first REG_REQ message, and a similar random delay of up to 10% of *macCtlReTxTimer* may be added to each retransmission.

B.4.6.2 Unregistering process

At any point in time, either the Base Node or the Service Node may decide to close an existing registration. This version of the specification does not provide provision for rejecting an unregister request. The Service Node or Base Node that receives an unregister request shall acknowledge its receipt and take appropriate actions.

Following a successful unregister, a Service Node shall move back from its present functional state to a *Disconnected* functional state and the Base Node may re-use any resources that were reserved for the unregistering Node.

Figure B.45 shows a successful unregister process initiated from a Service Node and Figure B.46 shows an unregister process initiated from the Base Node. Details on specific fields that identify unregister requests in REG control packets are given in Table B.9

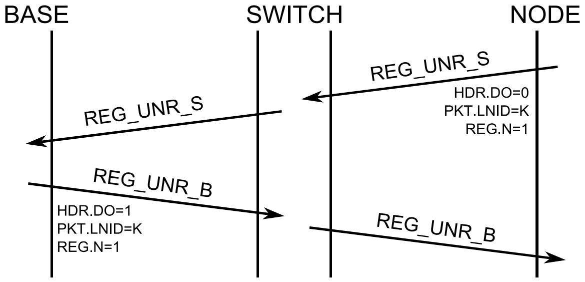


Figure B.45 – Unregistering process initiated by a terminal node

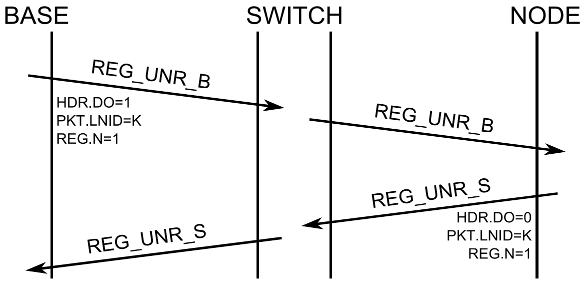


Figure B.46 – Unregistering process initiated by the base node

B.4.6.3 Promotion process

A Node that cannot reach any existing Switch may send promotion-needed frames so that a Terminal can be promoted and begin to switch. During this process, a Node that cannot reach any existing Switch may send PNPDUs so that a nearby Terminal can be promoted and begin to act as a Switch. During this process, a Terminal will receive PNPDUs and at its discretion, generate PRO_REQ control packets to the Base Node.

The Base Node examines the promotion requests during a period of time. It may use the address of the new Terminal, provided in the promotion-request packet, to decide whether or not to accept the promotion. It will decide which Node shall be promoted, if any, sending a promotion response. The other Nodes will not receive any answer to the promotion request to avoid Subnetwork saturation. Eventually, the Base Node may send a rejection if any special situation occurs. If the Subnetwork is specially preconfigured, the Base Node may send Terminal Node promotion requests directly to a Terminal Node.

When a Terminal Node requests promotion, the PRO.NSID field in the PRO_REQ_S message shall be set to all 1s. The PRO.NSID field shall contain an LSID allocated to the promoted Node in the PRO_REQ_B message. The acknowledging Switch Node shall set the PRO.NSID field in its PRO_ACK to the newly allocated LSID. This final PRO_ACK shall be used by intermediate Switch Nodes to update their switching tables as described in B.4.3.5.2.

When reusing LSIDs that have been released by a demotion process, the Base Node should not allocate the LSID until after (*macMaxCtlReTx + 1) * macCtlReTxTimer*. seconds to ensure all retransmit packets that might use that LSID have left the Subnetwork. Similarly, the Base Node shall not reuse an LNID freed by the Keep-Alive process until T_{keep_alive} seconds have passed, using the last known acknowledged T_{keep_alive} value, or if larger, the last unacknowledged $T_{keep\_alive,}$ for the Service Node using the LNID.

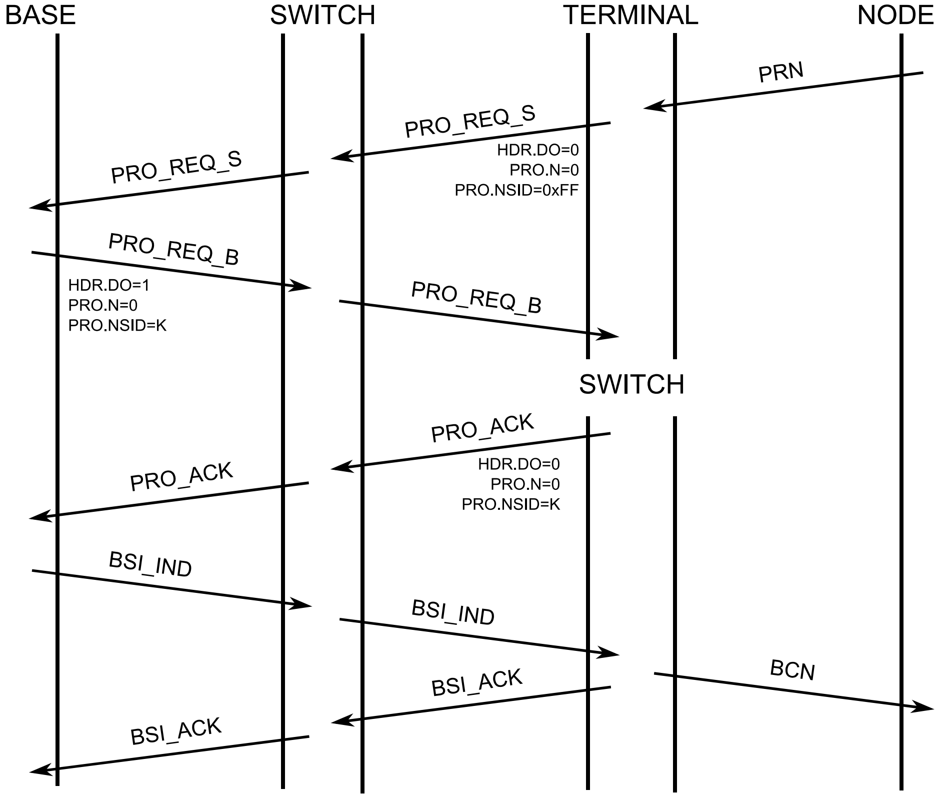


Figure B.47 – Promotion process initiated by a service node

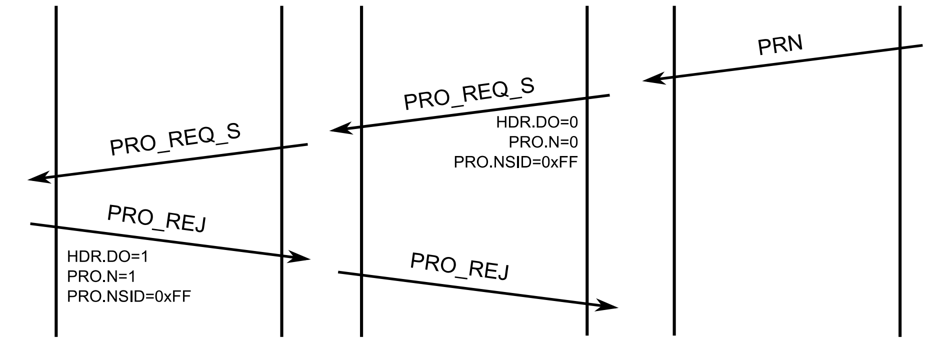


Figure B.48 – Promotion process rejected by the base node

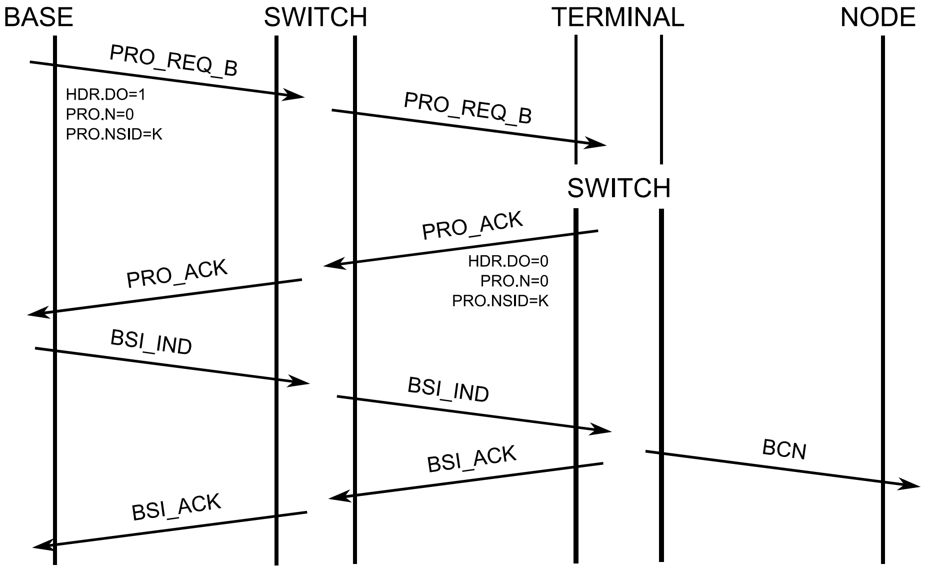


Figure B.49 – Promotion process initiated by the base node

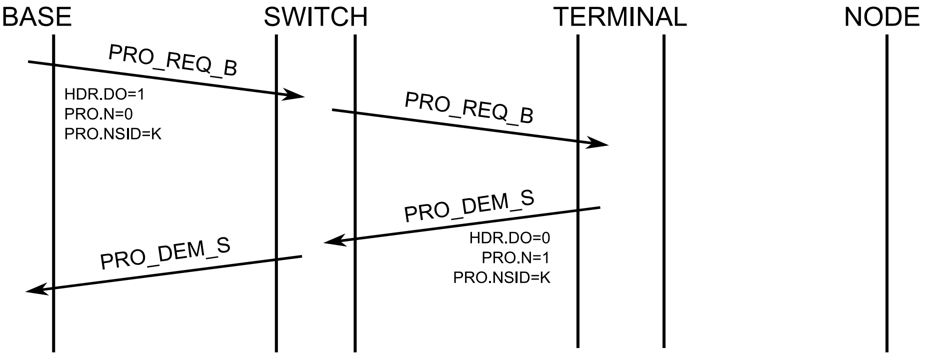


Figure B.50 – Promotion process rejected by a service node

B.4.6.4 Demotion process

The Base Node or a Switch Node may decide to discontinue a switching function at any time. The demotion process provides for such a mechanism. The PRO control packet is used for all demotion transactions.

The PRO.NSID field shall contain the SID of the Switch Node that is being demoted as part of the demotion transaction. The PRO.PNA field is not used in any demotion process transaction and its contents are not interpreted at either end.

Following the successful completion of a demotion process, a Switch Node shall immediately stop the transmission of beacons and change from a *Switch* functional state to a *Terminal* functional state. The Base Node may reallocate the LSID and Beacon Slot used by the demoted Switch after (*macMaxCtlReTx* + 1) * *macCtlReTxTimer* seconds to other Terminal Nodes requesting promotion.

The present version of this specification does not specify any explicit message to reject a demotion requested by a peer at the other end.

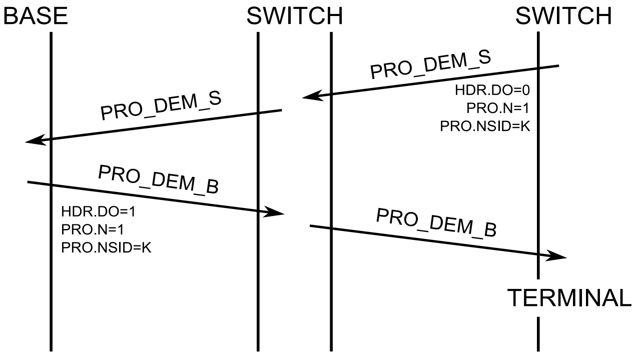


Figure B.51 – Demotion process initiated by a service node

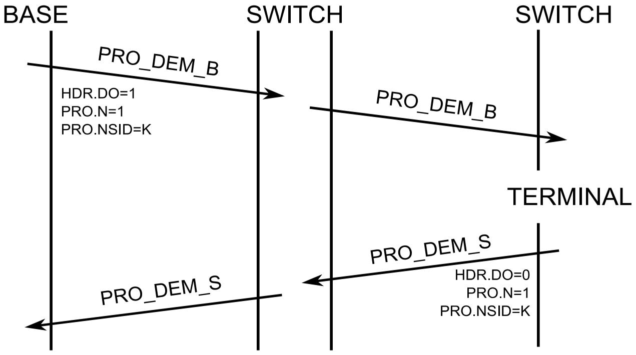


Figure B.52 – Demotion process initiated by the base node

B.4.6.5 Keep-alive process

The Keep-Alive process is used to detect when a Service Node has left the Subnetwork because of changes to the network configuration or because of fatal errors it cannot recover from.

When the Service Node receives the REG_RSP packet it uses the REG.TIME field to start a timer T_{keep_alive}. For every ALV_B it receives, it restarts this timer using the value from ALV.TIME. It should also send an ALV_S to the Base Node. If the timer ever expires, the Service Node assumes it has been unregistered by the Base Node. The message PRO_REQ does also reset the Keep-Alive timer to the PRO.TIME value.

Each switch along the path of a ALV_B message takes should keep a copy of the PRO.TIME and then ALV.TIME for each Switch Node below it in the tree. When the switch does not receive an ALV_S message from a Service Node below it for T_{keep_alive} as defined in PRO.TIME and ALV.TIME it should remove the Switch Node entry from its switch table. See clause B.4.3.5.2 for more information on the switching table. Additionally a Switch Node may use the REG.TIME and ALV.TIME to consider also every Service Node Registration status and take it into account for the switching table.

For every ALV_S or ALV_B message sent by the Base Node or Service Node, the counter ALV.TXCNT should be incremented before the message is sent. This counter is expected to wrap around. For every ALV_B or ALV_S message received by the Service Node or the Base Node the counter ALV.RXCNT should be incremented. This counter is also expected to wrap around. These two counters are placed into the ALV_S and ALV_B messages. The Base Node should keep a ALV.TXCNT and ALV.RXCNT separated counter for each Service Node. These counters are reset to zero in the Registration process.

The algorithm used by the Base Node to determine when to send ALV_B messages to registered Service Nodes and how to determine the value ALV.TIME and REG.TIME is not specified here.

B.4.6.6 Connection management

B.4.6.6.1 Connection establishment

Connection establishment works end-to-end, connecting the application layers of communicating peers. Owing to the tree topology, most connections in a Subnetwork will involve the Base Node at one end and a Service Node at the other. However, there may be cases when two Service Nodes within a Subnetwork need to establish connections. Such connections are called direct connections and are described in clause B.4.3.6.

All connection establishment messages use the CON control packet. The various control packets types and specific fields that unambiguously identify them are given in Table B.11.

Each successful connection established on the Subnetwork is allocated an LCID. The Base Node shall allocate an LCID that is unique for a given LNID.

NOTE – Either of the negotiating ends may decide to reject a connection establishment request. The receipt of a connection rejection does not amount to any restrictions on making future connection requests; it may however be advisable.

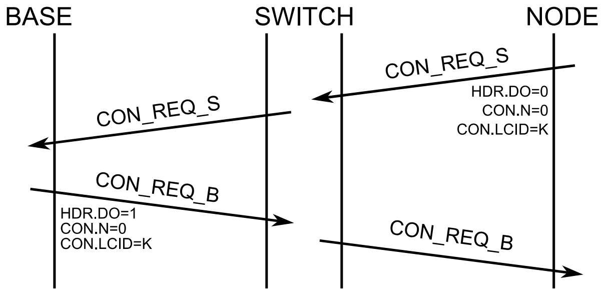


Figure B.53 – Connection establishment initiated by a service node

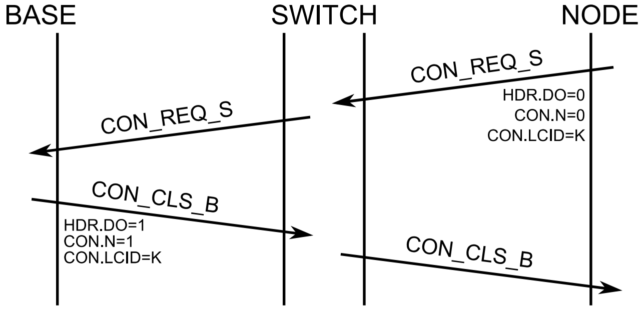


Figure B.54 – Connection establishment rejected by the base node

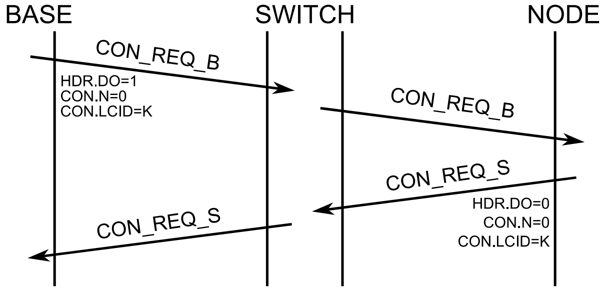


Figure B.55 – Connection establishment initiated by the base node

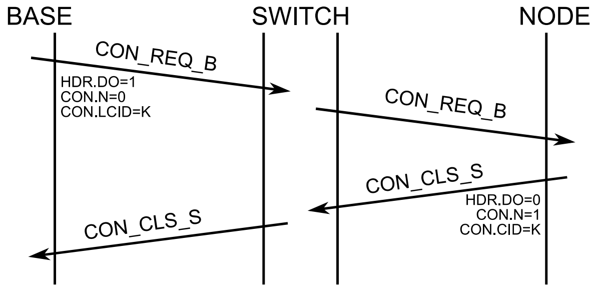


Figure B.56 – Connection establishment rejected by a service node

B.4.6.6.2 Connection closing

Either peer at both ends of a connection may decide to close the connection at any time. The CON control packet is used for all messages exchanged in the process of closing a connection. Only the CON.N field in the CON control packet is relevant in closing an active connection.

A connection closure request from one end is acknowledged by the other end before the connection is considered closed. The present version of this specification does not have any explicit message for rejecting a connection termination requested by a peer at the other end.

Figure B.57 and Figure B.58 show message exchange sequences in a connection closing process.

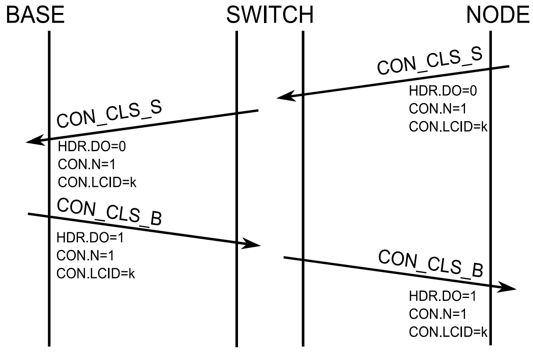


Figure B.57 – Disconnection initiated by a service node

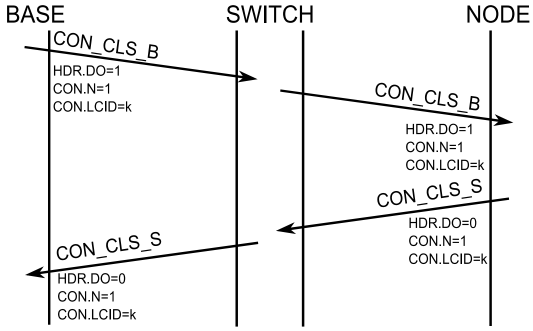


Figure B.58 – Disconnection initiated by the base node

B.4.6.7 Multicast group management

B.4.6.7.1 General

The joining and leaving of a multicast group can be initiated by the Base Node or the Service Node. The MUL control packet is used for all messages associated with multicast and the usual retransmit mechanism for control packets is used. These control messages are unicast between the Base Node and the Service Node.

B.4.6.7.2 Group join

Multicast group join maybe initiated from either the Base Node or Service Node. A device shall not start a new join procedure before an existing join procedure started by itself is completed.

Certain applications may require the Base Node to selectively invite certain Service Nodes to join a specific multicast group. In such cases, the Base Node starts a new group and invites Service Nodes as required by application.

Successful and failed group joins initiated from Base Node are shown in Figure B.59 and Figure B.60.

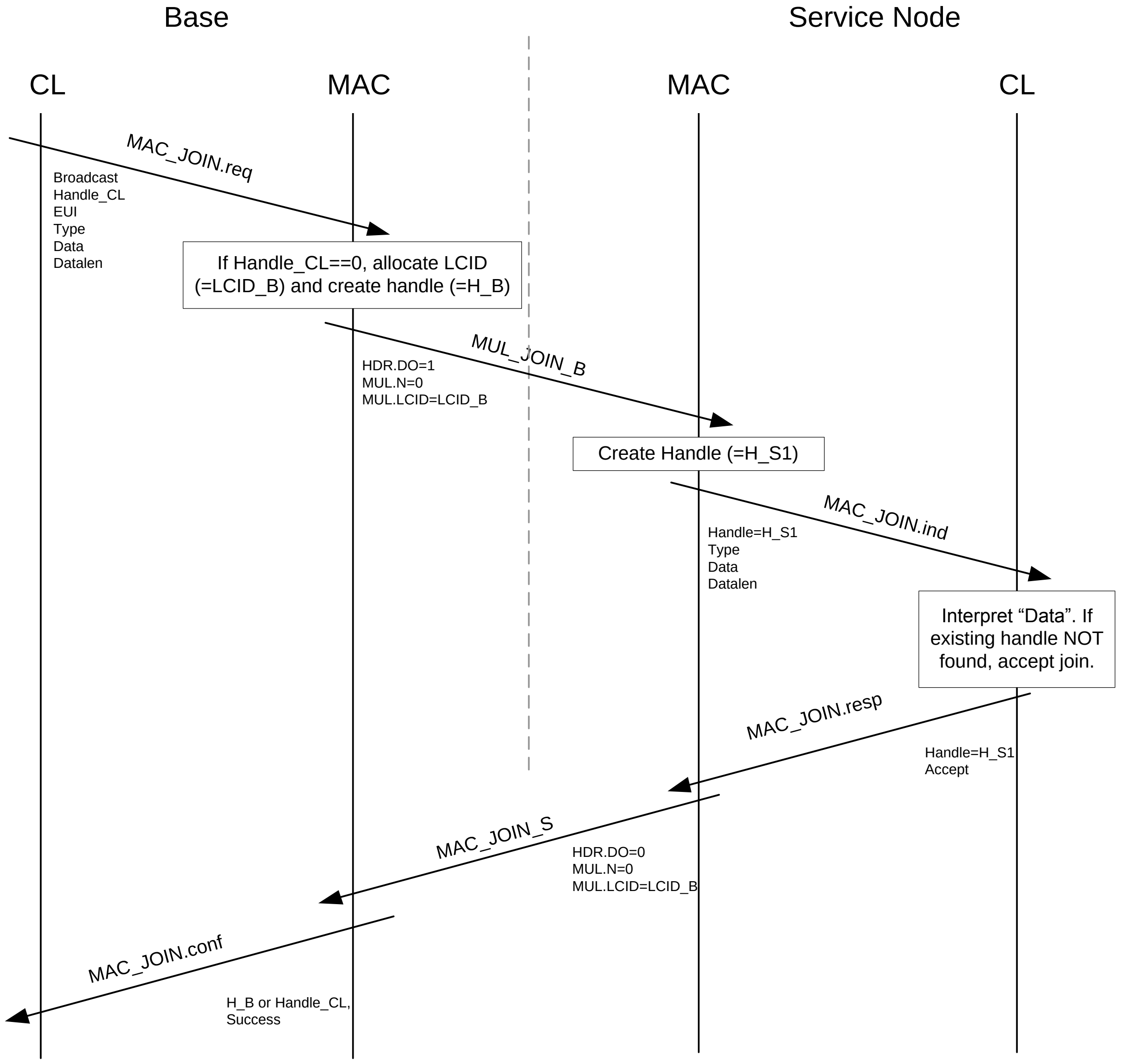


Figure B.59 – Successful group join initiated by base node

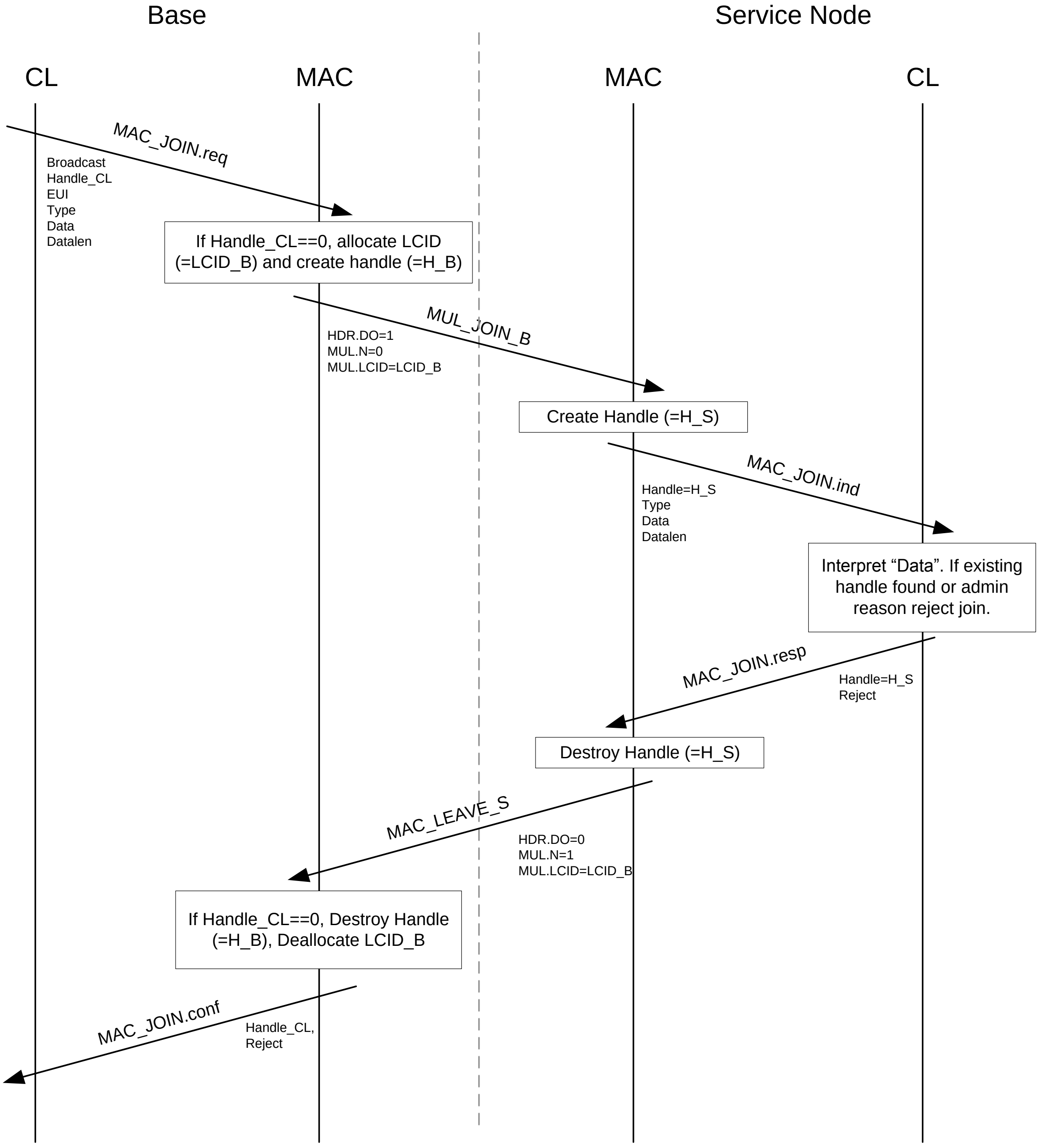


Figure B.60 – Failed group join initiated by base node

Successful and failed group joins initiated from Service Node are shown in Figure B.61 and Figure B.62.

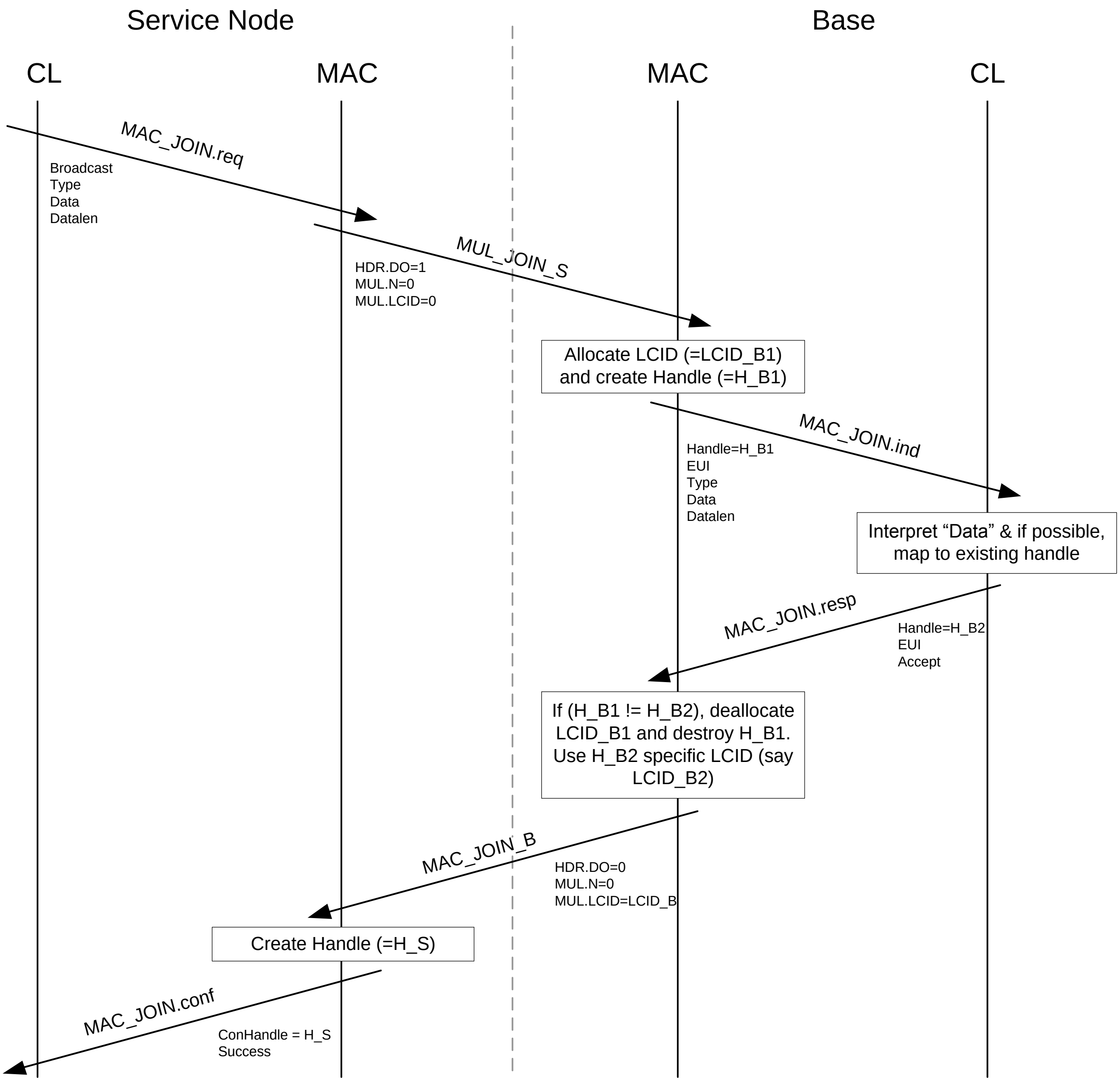


Figure B.61 – Successful group join initiated by service node

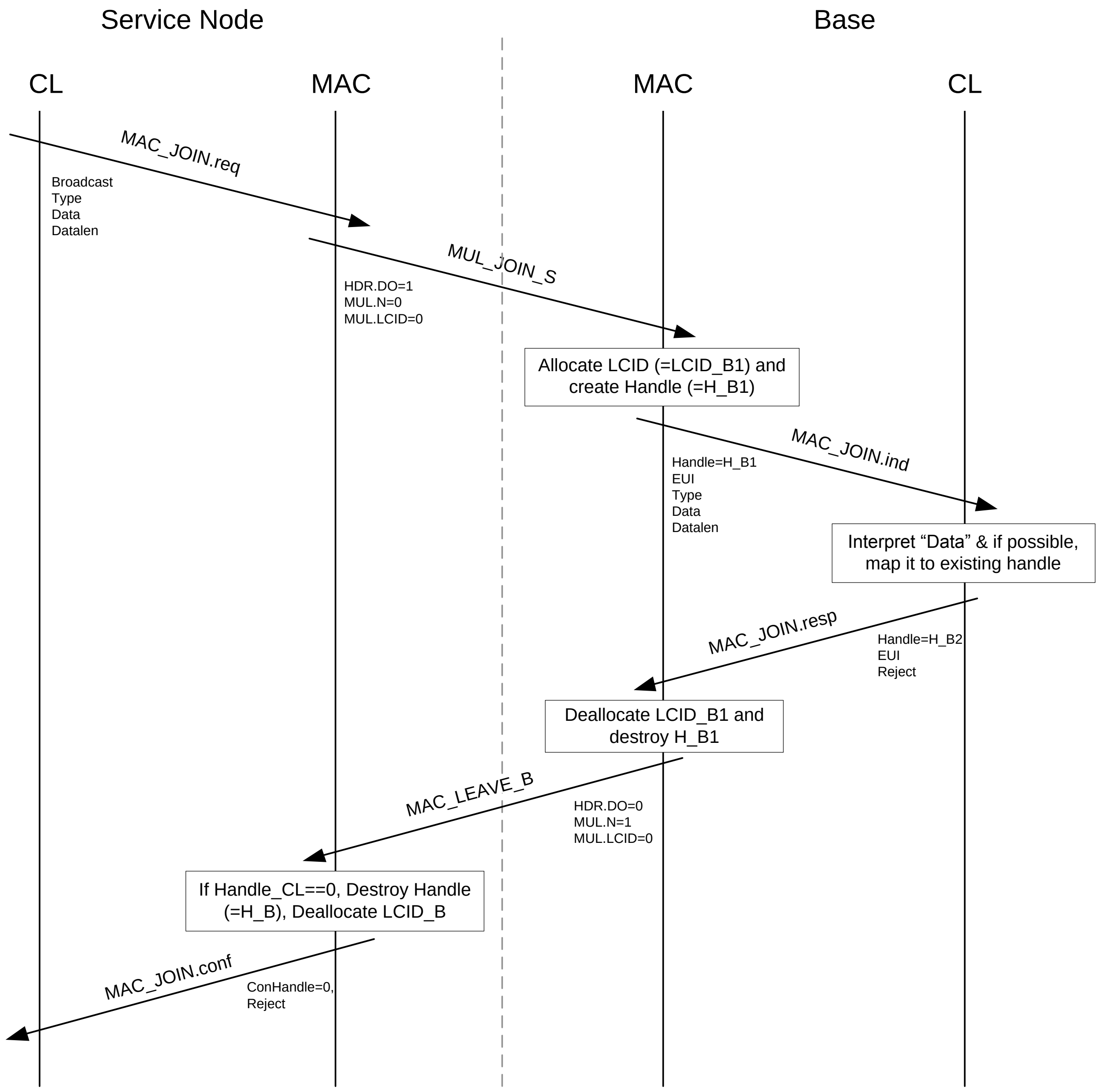


Figure B.62 – Failed group join initiated by service node

B.4.6.7.3 Group leave

Leaving a multicast group operates in the same way as connection removal. Either the Base Node or Service Node may decide to leave the group. A notable difference in the group leave process as compared to a group join is that there is no message sequence for rejecting a group leave request.

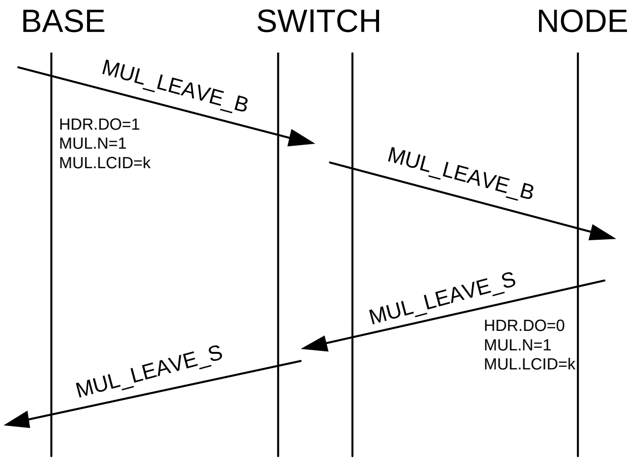


Figure B.63 – Leave initiated by the service node

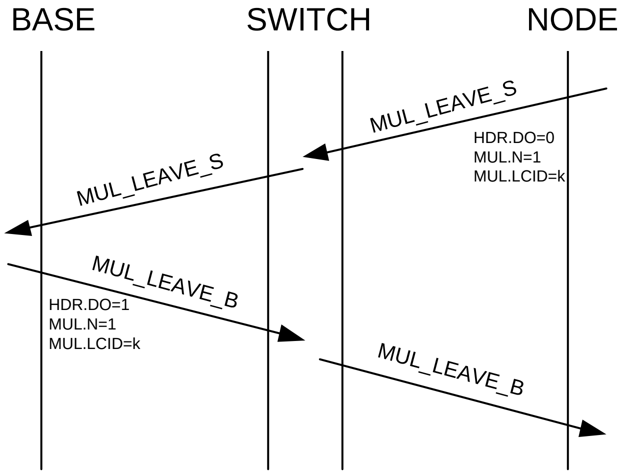


Figure B.64 – Leave initiated by the base node

B.4.6.8 PHY robustness management

B.4.6.8.1 General

The PHY layer has several parameters that affect the performance of the transmission: power transmission, modulation schema (constellation mapping and convolutional encoding). The transmitter needs feedback about the reception quality to adjust its transmission parameters. This feedback is sent using PRM control packets.

B.4.6.8.2 PHY robustness notification need detection

There are several sources of information that may be used to detect whether or not the robustness of the PHY is the right one:

- Received packets with invalid CRC.
- ARQ retransmissions.
- Control packet retransmissions.
- PRM requests sent by other Nodes to the same Switch Node (in the case of Node-to-switch notifications).
- PRM responses.

This information may be used to decide when to notify that the robustness of the PHY should be changed. This notification may be performed from a Service Node to a Switch Node and from a

Switch Node to a Service Node. For this purpose, the Base Node works as the root Switch, in exactly the same way the other Switch Nodes do.

B.4.6.8.3 PHY robustness notification

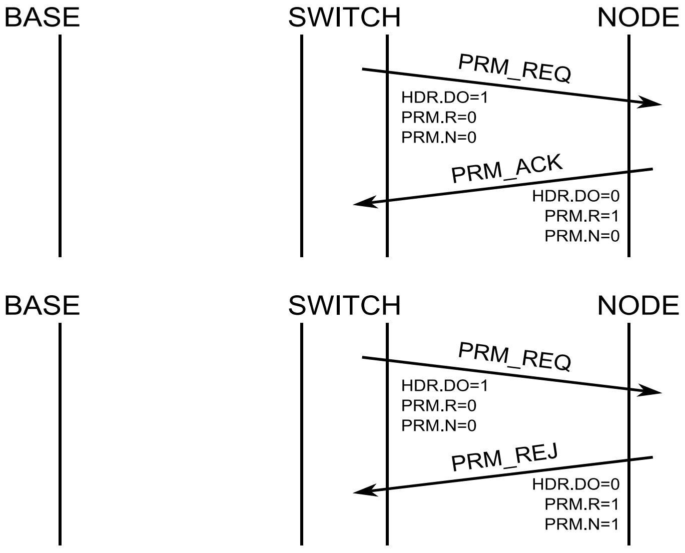


Figure B.65 – PHY robustness management requested by the switch node

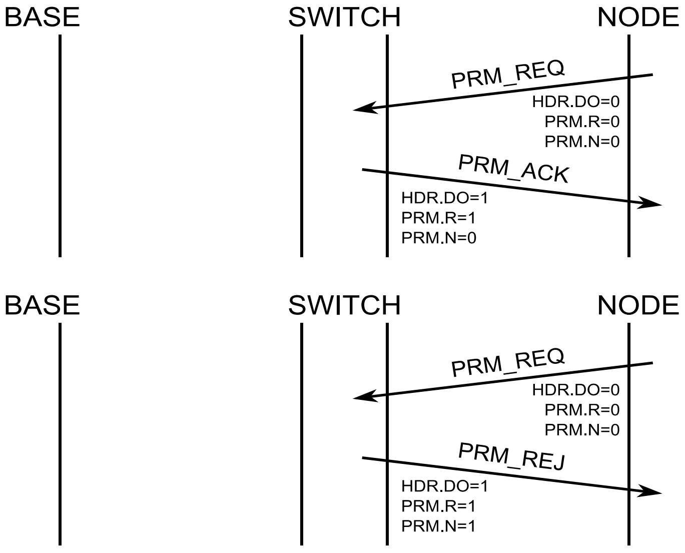


Figure B.66 – PHY robustness management requested by the service node

B.4.6.8.4 PHY robustness changing

From the PHY point of view there are several parameters that may be adjusted and which affect the transmission robustness: the transmission power and modulation parameters (convolutional encoding and constellation). As a general rule the following rules should be followed:

- **Increase robustness**: increase the power and, if it is not possible, improve the modulation scheme robustness (reducing throughput).
- **Reduce robustness**: reduce the modulation scheme robustness (increasing throughput) and, if it is not possible, reduce the transmission power.

B.4.6.9 Channel allocation and deallocation

Figure B.67 below shows a successful channel allocation sequence. All channel allocation requests are forwarded to the Base Node. Note that in order to assure a contention-free channel allocation along the entire path, the Base Node allocates non-overlapping times to intermediate Switch Nodes. In a multi-level Subnetwork, the Base Node may also reuse the allocated time at different levels. While reusing the said time, the Base Node needs to ensure that the levels that use the same time slots have sufficient separation so that there is no possible interference.

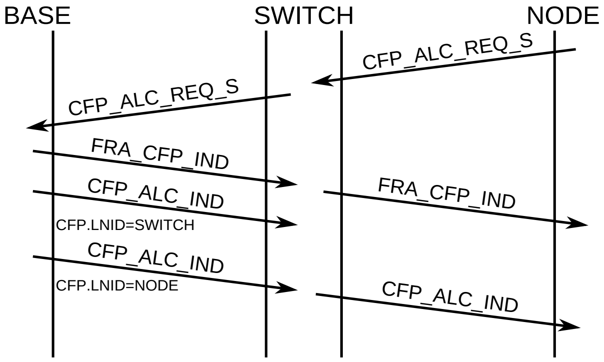


Figure B.67 – Successful allocation of CFP period

Figure B.68 below shows a channel de-allocation request from a Terminal device and the resulting confirmation from the Base Node.

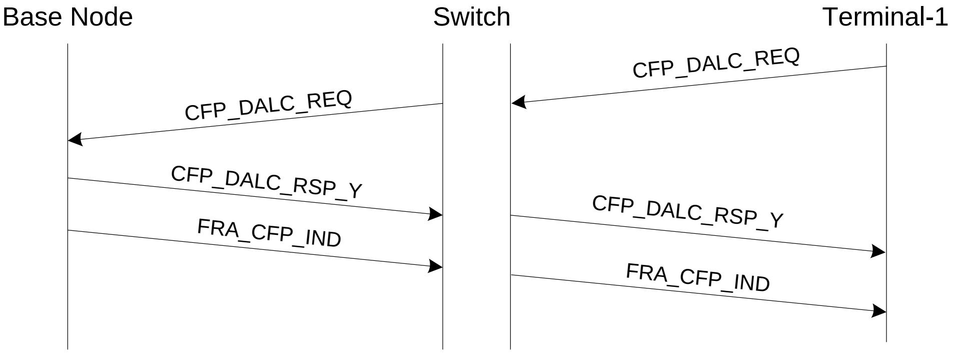


Figure B.68 – Successful channel de-allocation sequence

Figure B.69 below shows a sequence of events that may lead to a Base Node re-allocation contention-free slot to a Terminal device that already has slots allocated to it. In this example, a de-allocation request from Terminal-2 resulted in two changes: firstly, in global frame structure, this change is conveyed to the Subnetwork in the FRA_CFP_IND packet; secondly, it is specific to the time slot allocated to Terminal-1 within the CFP.

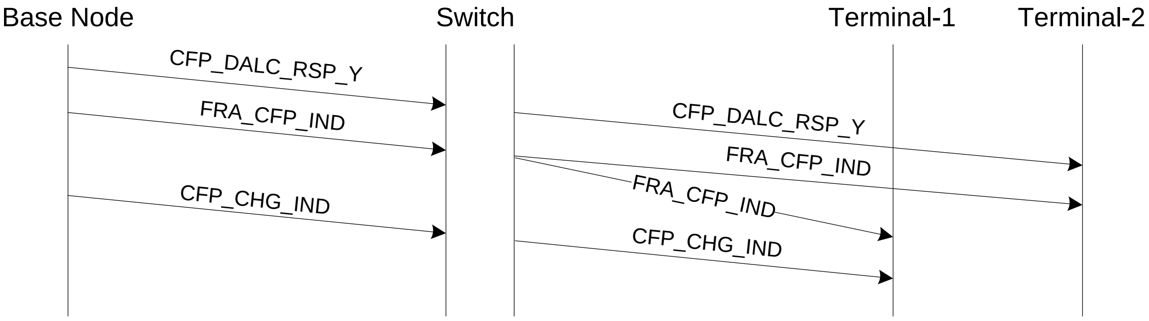


Figure B.69 – Deallocation of channel to one device results in the change of CFP allocated to another

B.4.7 Automatic repeat request (ARQ)

B.4.7.1 General

Devices complying with this specification may either implement an ARQ scheme as described in this clause or no ARQ at all. This specification provides for low-complexity Switch and Terminal devices that choose not to implement any ARQ mechanism at all.

B.4.7.2 Initial negotiation

ARQ is a connection property. During the initial connection negotiation, the originating device indicates its preference for ARQ or non-ARQ in CON.ARQ field. The responding device at the other end can indicate its acceptance or rejection of the ARQ in its response. If both devices agree to use ARQ for the connection, all traffic in the connection will use ARQ for acknowledgements, as described in clause B.4.7.3. If the responding device rejects the ARQ in its response, the data flowing through this connection will not use ARQ.

B.4.7.3 ARQ mechanism

B.4.7.3.1 General

The ARQ mechanism works between directly connected peers (original source and final destination), as long as both of them support ARQ implementation. This implies that even for a connection between the Base Node and a Terminal (connected via one or more intermediate Switch devices), ARQ works on an end-to-end basis. The behaviour of Switch Nodes in an ARQ-enabled connection is described in clause B.4.7.4. When using ARQ, a unique packet identifier is associated with each packet, to aid in acknowledgement. The packet identifier is 6 bits long and can therefore denote 64 distinct packets. ARQ windowing is supported, with a maximum window size of 32 (5 bits), as described in clause B.4.7.3.3.

B.4.7.3.2 ARQ PDU

B.4.7.3.2.1 General

The ARQ subheader is placed inside the data packets, after the packet header and before the ORIGINAL packet payload:

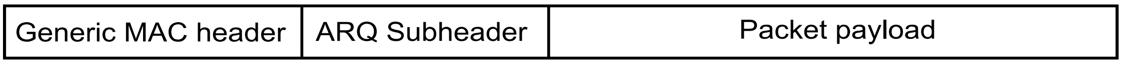


Figure B.70 – MAC Data PDU with ARQ subheader

For an ARQ PDU, the PKT.LEN field in the packet header will be set as the ARQ subheader length plus the original packet payload length. By doing this, the intermediate switching Node can correctly parse the whole PDU length without the knowledge that this PDU is ARQ-enabled, so that it can transparently relaying the ARQ PDU based on the addressing information alone.

The ARQ subheader contains a set of bytes, each byte containing different subfields. The most significant bit of each byte, the M bit, indicates if there are more bytes in the ARQ subheader.

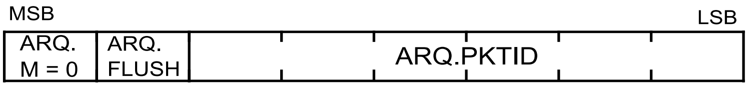


Figure B.71 – ARQ subheader only with the packet id

Figure B.71 shows an ARQ subheader with the first M bit of 0 and so the subheader is a single byte and contains only the packet ID for the transmitted packet.

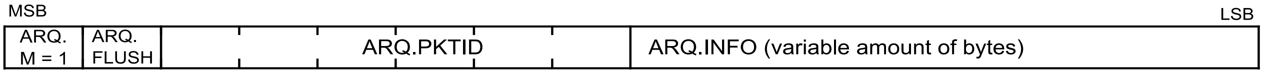


Figure B.72 – ARQ subheader with ARQ.INFO

Figure B.72 has the M bit in the first byte of the ARQ subheader set, and so the subheader contains multiple bytes. The first byte contains the packet ID of the transmitted packet and then follows the ARQ.INFO which is a list of one or more bytes, where each byte could have one of the following meanings:

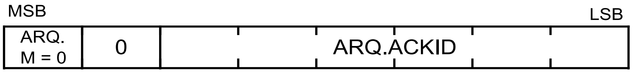


Figure B.73 – ARQ.ACK byte fields

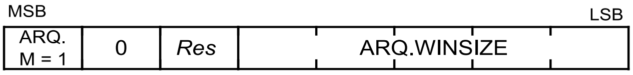


Figure B.74 – ARQ.WIN byte fields

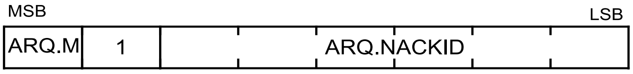


Figure B.75 – ARQ.NACK byte fields

If there are multiple packets lost, an ARQ.NACK is sent for each of them, from the first packet lost to the last packet lost. When there are several ARQ.NACK they implicitly acknowledge the packets before the first ARQ.NACK, and the packets in between the ARQ.NACKs. If an ARQ.ACK is present, it must be placed at the end of the ARQ subheader, and should reference to an ARQ.ACKID that is later than any other ARQ.NACKID, if present. If there is at least an ARQ.NACK and an ARQ.ACK they also implicitly acknowledge any packet in the middle between the last ARQ.NACKID and the ARQ.ACK.

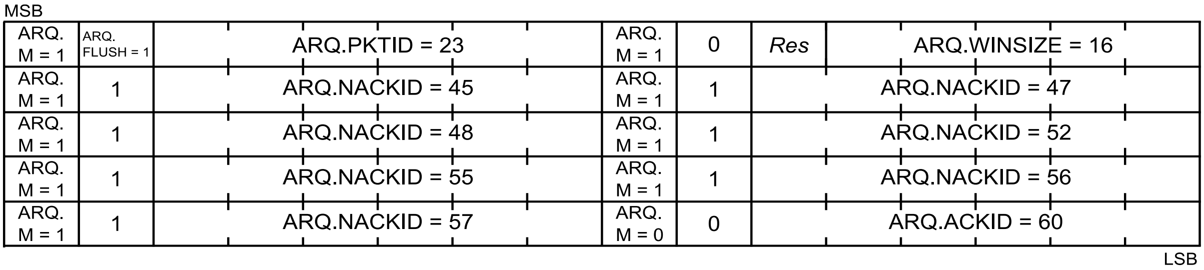
For interoperability, a device should be able to receive any well-formed ARQ subheader and should process at least the first ARQ.ACK or ARQ.NACK field.

The subfields have the following meanings as described in Table B.45

Table B.45 – ARQ fields

Field	Description
ARQ.FLUSH	ARQ.FLUSH = 1 If an ACK must be sent immediately. ARQ.FLUSH = 0 If an ACK is not needed.
ARQ.PKTID	The id of the current packet, if the packet is empty (with no data) this is the id of the packet that will be sent next.
ARQ.ACKID	The identifier with the next packet expected to be received.
ARQ.WINSIZE	The window size available from the last acknowledged packet. After a connection is established its window is 1.
ARQ.NACKID	Ids of the packets that need to be retransmitted.

B.4.7.3.2.2 ARQ subheader example



MSB

ARQ. M = 1	ARQ. FLUSH = 1	ARQ.PKTID = 23	ARQ. M = 1	0	*Res*	ARQ.WINSIZE = 16
ARQ. M = 1	1	ARQ.NACKID = 45	ARQ. M = 1	1	ARQ.NACKID = 47	
ARQ. M = 1	1	ARQ.NACKID = 48	ARQ. M = 1	1	ARQ.NACKID = 52	
ARQ. M = 1	1	ARQ.NACKID = 55	ARQ. M = 1	1	ARQ.NACKID = 56	
ARQ. M = 1	1	ARQ.NACKID = 57	ARQ. M = 0	0	ARQ.ACKID = 60	

LSB

Figure B.76 – Example of an ARQ subheader with all the fields present

In this example all the ARQ subheader fields are present. To make it understandable, since both Nodes are both transmitters and receivers, the side receiving this header will be called A and the other side transmitting B. The message has the packet ID of 23 if it contains data; otherwise the next data packet to be sent has the packet ID of 23. Since the flush bit is set it needs to be ACKed/NACKed.

B requests the retransmission of packets 45, 47, 48, 52, 55, 56 and 57. ACK = 60, so it has received packets <45, 46, 49, 50, 51, 53, 54, 58 and 59.

The window is 16 and it has received and processed up to packet 44 (first NACK = 45), so A can send all packets <= 60; that is, as well as sending the requested retransmits, it can also send packet ID = 60.

B.4.7.3.3 Windowing

A new connection between two peer devices starts with an implicit initial receiver window size of 1 and a packet identifier 0. This window size is a limiting case and the transaction (to start with) would behave like a "Stop and Wait" ARQ mechanism.

On receipt of an ARQ.WIN, the sender would adapt its window size to *ARQ.WINSIZE*. This buffer size is counted from the first packet completely ACK-ed, so if there is a NACK list and then an ACK the window size defines the number of packets from the first NACK-ed packet that could be sent. If there is just an ACK in the packet (without any NACK) the window size determines the number of packets that can be sent from that ACK.

An *ARQ.WINSIZE* value of 0 may be transmitted back by the receiver to indicate congestion at its end. In such cases, the transmitting end should wait for at least *ARQCongClrTime* before re-transmitting its data.

B.4.7.3.4 Flow control

The transmitter must manage the ACK sending algorithm by the flush bit; it is up to it having a proper ARQ communication. The receiver is only forced to send ACKs when the transmitter has sent a packet with the flush bit set, although the receiver could send more ACKs even if not forced to do it, because the flow control is only a responsibility of the transmitter.

These are the requisites to be interoperable, but the algorithm is up to the manufacturer. It is strongly recommended to piggyback data-ACK information in outgoing packets, to avoid the transmission of unnecessary packets just for ACK-ing.

B.4.7.3.5 Algorithm recommendation

No normative algorithm is specified.

B.4.7.3.6 Usage of ARQ in resource limited devices

Resource limited devices may have a low memory and simple implementation of ARQ. They may want to use a window of 1 packet. They will work as a "Stop and Wait" mechanism.

The ARQ subheader to be generated may be one of the followings:

If there is nothing to acknowledge:

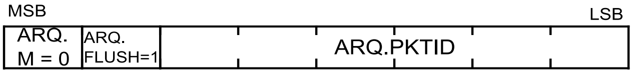


Figure B.77 – Stop and wait ARQ subheader with only packet ID

If there is something to acknowledge carrying data:

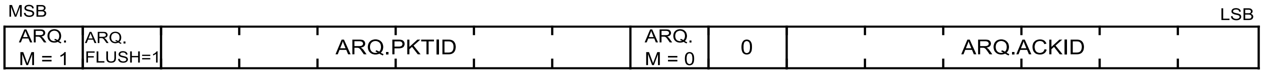


Figure B.78 – Stop and wait ARQ subheader with an ACK

If there is something to acknowledge but without any data in the packet:

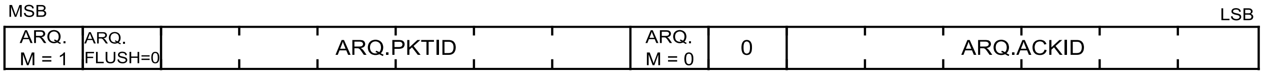


Figure B.79 – Stop and wait ARQ subheader without data and with an ACK

The ARQ.WINSIZE is not generally transmitted because the window size is already 1 by default, it only may be transmitted to handle congestion and to resume the transmission again.

B.4.7.4 ARQ packets switching

All Switch Nodes shall support transparent bridging of ARQ traffic, whether or not they support ARQ for their own transmission and reception. In this mode, Switch Nodes are not required to buffer the packets of the ARQ connections for retransmission.

Some Switch Nodes may buffer the packets of the ARQ connections, and perform retransmission in response to NACKs for these packets. The following general principles shall be followed.

- The acknowledged packet identifiers shall have end-to-end coherency.
- The buffering of packets in Switch Nodes and their retransmissions shall be transparent to the source and Destination Nodes, i.e., a Source or Destination Node shall not be required to know whether or not an intermediate Switch has buffered packets for switched data.

B.5 Convergence layer

B.5.1 Overview

Figure B.80 shows the overall structure of the Convergence layer.

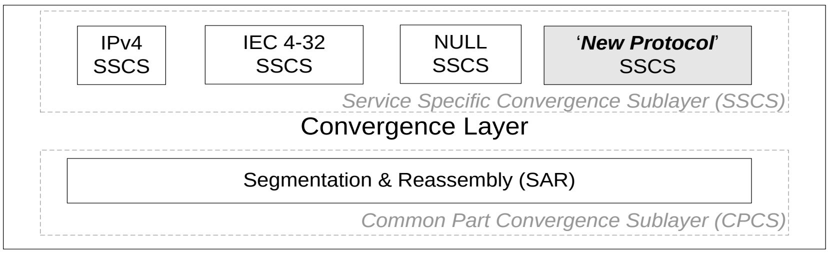

IPv4
SSCS
IEC 4-32
SSCS
NULL
SSCS
'New Protocol'
SSCS
Service Specific Convergence Sublayer (SSCS)
Convergence Layer
Segmentation & Reassembly (SAR)
Common Part Convergence Sublayer (CPCS)

Figure B.80 – Structure of the convergence layer

The Convergence layer is separated into two sublayers. The Common Part Convergence Sublayer (CPCS) provides a set of generic services. The Service Specific Convergence Sublayer (SSCS) contains services that are specific to one communication profile. There are several SSCSs, typically one per communication profile, but only one CPCS. The use of CPCS services is optional in that a certain SSCS will use the services it needs from the CPCS, and omit services which are not needed.

B.5.2 Common part convergence sublayer (CPCS)

B.5.2.1 General

This specification defines only one CPCS service: Segmentation and Reassembly (SAR).

B.5.2.2 Segmentation and reassembly (SAR)

B.5.2.2.1 General

CPCS SDUs which are larger than 'ClMTUSize-1' bytes are segmented at the CPCS. CPCS SDUs which are equal or smaller than 'ClMTUSize-1' bytes may also optionally be segmented. Segmentation means breaking up a CPCS SDU into smaller parts to be transferred by the MAC layer. At the peer CPCS, the smaller parts (segments) are put back together (i.e., reassembled) to form the complete CPCS SDU. All segments except the last segment of a segmented SDU must be the same size and at most *ClMTUSize* bytes in length. Segments may be decided to be smaller than 'ClMTUSize-1' bytes, e.g., when the channel is poor. The last segment may of course be smaller than 'ClMTUSize-1' bytes.

In order to keep SAR functionality simple, the *ClMTUSize* is a constant value for all possible modulation/coding combinations at PHY layer. The value of ClMTUSize is such that with any modulation/coding combination, it is always possible to transmit a single segment in one PPDU. Therefore, there is no need for discovering a specific MTU between peer CPCSs or modifying the SAR configuration for every change in the modulation/coding combination. In order to increase efficiency, a Service Node which supports packet aggregation may combine multiple segments into one PPDU when communicating with its peer.

Segmentation always adds a 1-byte header to each segment. The first 2 bits of SAR header identify the type of segment. The semantics of the rest of the header information then depend on the type of segment. The structure of different header types is shown in Figure B.81 and individual fields are explained in

Table B.46. Not all fields are present in each SAR header. Either SAR.NSEGS or SAR.SEQ is present, but not both.

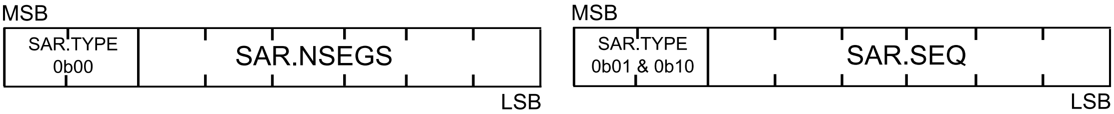


Figure B.81 – Segmentation and reassembly headers

Table B.46 – SAR header fields

Name	Length	Description
SAR.TYPE	2 bits	Type of segment. • 0b00: first segment; • 0b01: intermediate segment; • 0b10: last segment; • 0b11: reserved in this version of the specification.
SAR.NSEGS	6 bits	'Number of Segments' – 1.
SAR.SEQ	6 bits	Sequence number of segment.

Every segment (except for the first one) includes a sequence number so that the loss of a segment could be detected in reassembly. The sequence numbering shall start from zero with every new CPCS SDU. The first segment which contains a SAR.SEQ field must have SAR.SEQ = 0. All subsequent segments from the same CPCS SDU shall increase this sequence number such that the SAR.SEQ field adds one with every transmission.

The value SAR.NSEGS indicates the total number of segments, minus one. So when SAR.NSEGS = 0, the CPCS SDU is sent in one segment. SAR.NSEGS = 63 indicates there will be 64 segments to form the full CPCS SDU. When SAR.NSEGS = 0, it indicates that this first segment is also the last segment. No further segment with SAR.TYPE = 0b01 or 0b10 is to be expected for this one-segment CPCS SDU.

SAR at the receiving end shall buffer all segments and deliver only fully reassembled CPCS SDUs to the SSCS above. Should reassembly fail due to a segment not being received or too many segments being …received, etc., SAR shall not deliver any incomplete CPCS SDU to the SSCS above.

B.5.2.2.2 SAR constants

Table B.47 shows the constants for the SAR service.

Table B.47 – SAR constants

Constant	Value
ClMTUSize	256 Bytes.
ClMaxAppPktSize	Max Value (SAR.NSEGS) x ClMTUSize.

B.5.3 NULL specific service convergence sublayer (NULL SSCS)

B.5.3.1 Overview

Null SSCS provides the MAC layer with a transparent path to upper layers, being as simple as possible and minimizing overhead. It is intended for applications that do not need any special convergence capability.

The unicast and multicast connections of this SSCS shall use the SAR service, as defined in B.5.2.2. If they do not need the SAR service they shall still include the SAR header (notifying just one segment).

The CON.TYPE and MUL.TYPE (see PRIME Annex B) for unicast connections and multicast groups shall use the same type that has been already defined for the application that makes use of this Null SSCS.

B.5.3.2 Primitives

Null SSCS primitives are just a direct mapping of the MAC primitives. A full description of every primitive is avoided, because the mapping is direct and they will work as the ones of the MAC layer.

The directly mapped primitives have exactly the same parameters as the ones in the MAC layer and perform the same functionality. The set of primitives that are directly mapped are shown below.

Table B.48 – Primitive mapping between the Null SSCS primitives and the MAC layer primitives

Null SSCS mapped to …	… a MAC primitive
CL_NULL_ESTABLISH.request	MAC_ESTABLISH.request
CL_NULL_ESTABLISH.indication	MAC_ESTABLISH.indication
CL_NULL_ESTABLISH.response	MAC_ESTABLISH.response
CL_NULL_ESTABLISH.confirm	MAC_ESTABLISH.confirm
CL_NULL_RELEASE.request	MAC_RELEASE.request
CL_NULL_RELEASE.indication	MAC_RELEASE.indication
CL_NULL_RELEASE.response	MAC_RELEASE.response
CL_NULL_RELEASE.confirm	MAC_RELEASE.confirm
CL_NULL_JOIN.request	MAC_JOIN.request
CL_NULL_JOIN.indication	MAC_JOIN.indication
CL_NULL_JOIN.response	MAC_JOIN.response
CL_NULL_JOIN.confirm	MAC_JOIN.confirm
CL_NULL_LEAVE.request	MAC_LEAVE.request
CL_NULL_LEAVE.indication	MAC_LEAVE.indication
CL_NULL_LEAVE.response	MAC_LEAVE.response
CL_NULL_LEAVE.confirm	MAC_LEAVE.confirm
CL_NULL_DATA.request	MAC_DATA.request
CL_NULL_DATA.indication	MAC_DATA.indication
CL_NULL_DATA.confirm	MAC_DATA.confirm
CL_NULL_SEND.request	MAC_SEND.request
CL_NULL_SEND.indication	MAC_SEND.indication
CL_NULL_SEND.confirm	MAC_SEND.confirm

B.5.4 IPv4 specific service convergence sublayer (IPv4 SSCS)

B.5.4.1 Overview

The IPv4 SSCS provides an efficient method for transferring IPv4 packets over the OFDM TYPE 1 subnetworks. Several conventions do apply:

- A Service Node can send IPv4 packets to the Base Node or to other Service Nodes.
- It is assumed that the Base Node acts as a router between the OFDM TYPE 1 subnetwork and any other network. The Base Node could also act as a NAT. How the Base Node connects to the other networks is beyond the scope of this specification.
- In order to keep implementations simple, only one single route is supported per local IPv4 address.

- Service Nodes may use statically configured IPv4 addresses or DHCP to obtain IPv4 addresses.
- The Base Node performs IPv4 to EUI-48 address resolution. Each Service Node registers its IPv4 address and EUI-48 address with the Base Node (see clause B.5.4.2). Other Service Nodes can then query the Base Node to resolve an IPv4 address into a EUI-48 address. This requires the establishment of a dedicated connection with the Base Node for address resolution.
- The IPv4 SSCS performs the routing of IPv4 packets. In other words, the IPv4 SSCS will decide whether the packet should be sent directly to another Service Node or forwarded to the configured gateway.
- Although IPv4 is a connectionless protocol, the IPv4 SSCS is connection-oriented. Once address resolution has been performed, a connection is established between the source and destination Service Node for the transfer of IPv4 packets. This connection is maintained while traffic is being transferred and may be closed after a period of inactivity.
- The CPCS (see B.5.2) SAR sublayer shall always be present with the IPv4 Convergence layer. Generated MSDUs are at most 'ClMTUSize' bytes long and upper layer PDU messages are not expected must not to be longer than ClMaxAppPktSize.
- Optionally TCP/IPv4 headers may be compressed. Compression is negotiated as part of the connection establishment phase.
- The broadcasting of IPv4 packets is supported using the MAC broadcast mechanism.
- The multicasting of IPv4 packets is supported using the MAC multicast mechanism.

The IPv4 SSCS has a number of connection types. For address resolution there is a connection to the Base Node. For IPv4 data transfer there is one connection per Destination Node: with the Base Node that acts as the IPv4 gateway to other networks or to/with any other Node in the same subnetwork. This is shown in Figure B.82.

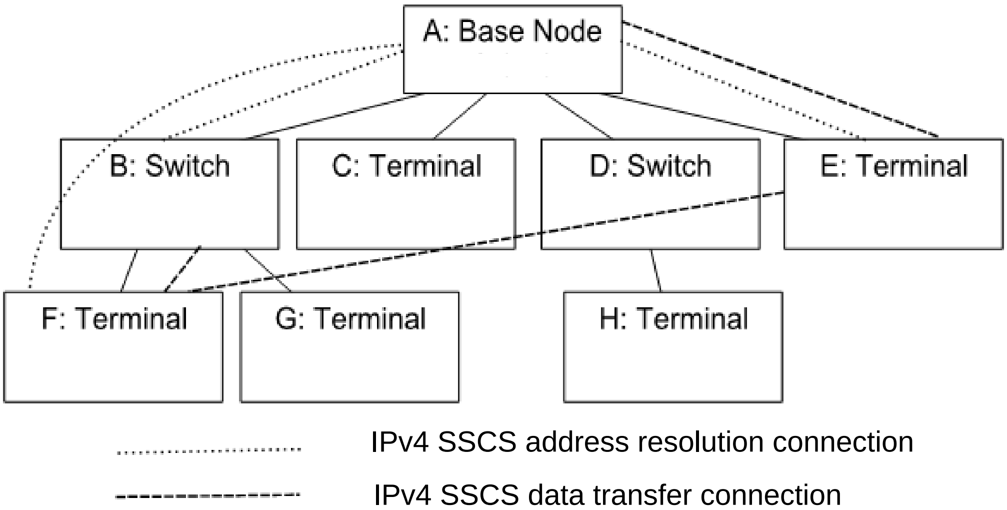


Figure B.82 – IPv4 SSCS connection example

Here, Nodes B, E and F have address resolution connections to the Base Node. Node E has a data connection to the Base Node and Node F. Node F is also has a data connection to Node B. The figure does not show broadcast and multicast connections.

B.5.4.2 Address resolution

B.5.4.2.1 General

The IPv4 layer will present the IPV4 SSCS with an IPv4 packet to be transferred. The IPV4 SSCS is responsible for determining which Service Node the packet should be delivered to using the IPv4 addresses in the packet. The IPV4 SSCS must then establish a connection to the destination if one does not already exist so that the packet can be transferred. Three classes of IPv4 addresses can be used and the following clauses describe how these addresses are resolved into EUI-48 addresses.

B.5.4.2.2 Unicast addresses

B.5.4.2.2.1 General

IPv4 unicast addresses must be resolved into unicast EUI-48 addresses. The Base Node maintains a database of IPv4 addresses and EUI-48 addresses. Address resolution then operates by querying this database. A Service Node must establish a connection to the address resolution service running on the Base Node, using the connection type value TYPE (see PRIME Annex B) TYPE_CL_IPv4_AR. No data should be passed in the connection establishment. Using this connection, the Service Node can use two mechanisms as defined in the following paragraphs.

B.5.4.2.2.2 Address registration and unregistration

A Service Node uses the AR_REGISTER_S message to register an IPv4 address and the corresponding EUI-48 address meaning request from the base node to record inside its registration table, the IPv4 address and its corresponding service node EUI-48. The Base Node will acknowledge an AR_REGISTER_B message. The Service Node may register multiple IPv4 addresses for the same EUI-48 address.

A Service Node uses the AR_DEREGISTER_S message to unregister an IPv4 address and the corresponding EUI-48 address meaning requests from the base node to delete inside its registration table, the entry corresponding to the concerned IPv4 address. The Base Node will acknowledge it with an AR_DEREGISTER_B message.

When the IPv4 address resolution connection between the Service Node and the Base Node is closed, the Base Node should remove all addresses associated to that connection.

B.5.4.2.2.3 Address lookup

A Service Node uses the AR_LOOKUP_S message to perform a lookup. The message contains the IPv4 address to be resolved. The Base Node will respond with an AR_LOOKUP_B message that contains an error code and, if there is no error, the EUI-48 address associated with the IPv4 address. If the Base Node has multiple entries in its database for the same IPv4 address, the possible returned EUI-48 address is undefined.

B.5.4.2.3 Broadcast address

IPv4 broadcast address 255.255.255.255 maps to a MAC broadcast connection with LCID equal to LCI_CL_IPv4_BROADCAST. All IPv4 broadcast packets will be sent to this connection. When an IPv4 broadcast packet is received on this connection, the IPv4 address should be examined to determine if it is a broadcast packet for the subnetwork in which the Node has an IPv4 address. Only broadcast packets from member subnets should be passed up the IPv4 protocol stack.

B.5.4.2.4 Multicast addresses

Multicast IPv4 addresses are mapped to an OFDM TYPE 1 MAC multicast connection by the Base Node using an address resolution protocol.

To join a multicast group, AR_MCAST_REG_S is sent from the Service Node to the Base Node with the IPv4 multicast address. The Base Node will reply with an AR_MCAST_REG_B that contains the LCID value assigned to the said multicast address. However, the Base Node may also allocate other LCIDs which are not in use if it so wishes. The Service Node can then join a multicast group (see B.4.6.7.2) for the given LCID to receive IPv4 multicast packets. These LCID values can be reused so that multiple IPv4 destination multicast addresses can be seen on the same LCID. To leave the multicast group, AR_MCAST_UNREG_S is sent from the Service Node to the Base Node with the IPv4 multicast address. The Base Node will acknowledge it with an AR_MCAST_UNREG_B message.

When a Service Node wants to send an IPv4 multicast datagram, it just uses the appropriate LCID. If the Service Node has not joined the multicast group, it needs first to learn the LCID to be used. The process with AR_MCAST_REG_{S|B} messages as described above can be used. While IPv4 multicast packets are still being sent, the Service Node remains registered to the multicast group. T_{mcast_reg} after the last IPv4 multicast datagram was sent, the Service Node should unregister from the multicast group, by means of AR_MCAST_UNREG_{S|B} messages. The nominal value of T_{mcast_reg} is 10 minutes; however, other values may be used.

B.5.4.2.5 Retransmission of address resolution packets

The connection between the Service Node and the Base Node for address resolution is not reliable if the MAC ARQ is not used. The Service Node is responsible for making retransmissions if the Base Node does not respond in one second. It is not considered an error when the Base Node receives the same registration requests multiple times or is asked to remove a registration that does not exist. These conditions can be the result of retransmissions.

B.5.4.3 IPv4 packet transfer

For packets to be transferred, a connection needs to be established between source and Destination Nodes. The IPV4 SSCS will examine each IPv4 packet to determine the destination EUI-48 address. If a data connection to the destination already exists, the packet is sent. To establish this, IPv4 SSCS keeps a table for each connection, with information shown in Table B.49 (see RFC 1144). To use this table, it is first necessary to determine if the IPv4 destination address is in the local subnetwork or if a gateway has to be used. The netmask associated with the local IPv4 address is used to determine this. If the IPv4 destination address is not in the local Subnetwork, the address of the default gateway is used instead of the destination address when the table is searched.

Table B.49 – IPV4 SSCS table entry

Parameter	Description
CL_IPv4_Con.Remote_IP	Remote IPv4 address of this connection.
CL_IPv4_Con.ConHandle	MAC Connection handle for the connection.
CL_IPv4_Con.LastUsed	Timestamp of last packet received/transmitted.
CL_IPv4_Con.HC	Header Compression scheme being used.
CL_IPv4_CON.RxSeq	Next expected Receive sequence number.
CL_IPv4_CON.TxSeq	Sequence number for next transmission.

The IPV4 SSCS may close a connection when it has not been used for an implementation-defined time period. When the connection is closed the entry for the connection is removed at both ends of the connection.

When a connection to the destination does not exist, more work is necessary. The address resolution service is used to determine the EUI-48 address of the remote IPv4 address if it is local or the gateway associated with the local address if the destination address is in another subnetwork. When the Base Node replies with the EUI-48 address of the destination Service Node, a MAC connection is established to the remote device. The TYPE value of this connection is

TYPE_CL_IPv4_UNICAST. The data passed in the request message is defined in clause B.5.4.7.4. The local IPv4 address is provided so that the remote device can add the new connection to its cache of connections for sending data in the opposite direction. The use of Van Jacobson Header Compression is also negotiated as part of the connection establishment. Once the connection has been established, the IPv4 packet can be sent.

When the packet is addressed to the IPv4 broadcast address, the packet has to be sent using the MAC broadcast service. When the IPV4 SSCS is opened, a broadcast connection is established for transferring all broadcast packets. The broadcast IPv4 packet is simply sent to this connection. Any packet received on this broadcast connection is passed to the IPv4 protocol stack.

B.5.4.4 Segmentation and reassembly

The IPV4 SSCS should support IPv4 packets with an MTU of 1500 bytes. This requires the use of SAR (see B.5.2.2).

B.5.4.5 Header compression

Van Jacobson TCP/IP Header Compression is an optional feature in the IPv4 SSCS. The use of VJ compression is negotiated as part of the connection establishment phase of the connection between two Service Nodes.

VJ compression is designed for use over a point-to-point link layer that can inform the decompressor when packets have been corrupted or lost. When there are errors or lost packets, the decompressor can then resynchronize with the compressor. Without this resynchronization process, erroneous packets will be produced and passed up the IPv4 stack.

The MAC layer does not provide the facility of detecting lost packets or reporting corrupt packets. Thus, it is necessary to add this functionality in the IPV4 SSCS. The IPV4 SSCS maintains two sequence numbers when VJ compression is enabled for a connection. These sequence numbers are 8 bits in size. When transmitting an IPv4 packet, the CL_IPv4_CON.TxSeq sequence number is placed in the packet header, as shown in clause B.5.4.3. The sequence number is then incremented. Upon reception of a packet, the sequence number in the received packet is compared against CL_IPv4_CON.RxSeq. If they differ, TYPE_ERROR, as defined in RFC 1144, is passed to the decompressor. The CL_IPv4_CON.RxSeq value is always updated to the value received in the packet header.

Header compression should never be negotiated for broadcast or multicast packets.

B.5.4.6 Quality of service mapping

The OFDM TYPE 1 MAC specifies that the contention-based access mechanism supports 4 priority levels (1-4). Level 1 is used for MAC signalling messages, but not exclusively so.

IPv4 packets include a TOS field in the header to indicate the QoS the packet would like to receive. Three bits of the TOS indicate the IP Precedence. The following table specifies how the IP Precedence is mapped into the OFDM TYPE 1 MAC priority.

Table B.50 – Mapping IPv4 precedence to OFDM TYPE 1 MAC priority

IP precedence	MAC priority
000 – Routine	4
001 – Priority	4
010 – Immediate	3
011 – Flash	3
100 – Flash Override	2
101 – Critical	2
110 – Internetwork Control	1
111 – Network Control	1

B.5.4.7 Packet formats and connection data

B.5.4.7.1 General

This clause defines the format of IPV4 SSCS PDUs.

B.5.4.7.2 Address resolution PDUs

B.5.4.7.2.1 General

The following PDUs are transferred over the address resolution connection between the Service Node and the Base Node. The following clauses define AR.MSG values in the range of 0 to 11. All higher values are reserved for later versions of this specification.

B.5.4.7.2.2 AR_REGISTER_S

Table B.51 shows the address resolution register message sent from the Service Node to the Base Node.

Table B.51 – AR_REGISTER_S message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type. • For AR_REGISTER_S = 0.
AR.IPv4	32-bits	IPv4 address to be registered.
AR.EUI-48	48-bits	EUI-48 to be registered.

B.5.4.7.2.3 AR_REGISTER_B

Table B.52 shows the address resolution register acknowledgment message sent from the Base Node to the Service Node.

Table B.52 – AR_REGISTER_B message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type. • For AR_REGISTER_B = 1.
AR.IPv4	32-bits	Registered IPv4 address.
AR.EUI-48	48-bits	EUI-48 registered.

The AR.IPv4 and AR.EUI-48 fields are included in the AR_REGISTER_B message so that the Service Node can perform multiple overlapping registrations.

B.5.4.7.2.4 AR_UNREGISTER_S

Table B.53 shows the address resolution unregister message sent from the Service Node to the Base Node.

Table B.53 – AR_UNREGISTER_S message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type. • For AR_UNREGISTER_S = 2.
AR.IPv4	32-bits	IPv4 address to be unregistered.
AR.EUI-48	48-bits	EUI-48 to be unregistered.

B.5.4.7.2.5 AR_UNREGISTER_B

Table B.54 shows the address resolution unregister acknowledgment message sent from the Base Node to the Service Node.

Table B.54 – AR_UNREGISTER_B message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type. • For AR_UNREGISTER_B = 3.
AR.IPv4	32-bits	Unregistered IPv4 address.
AR.EUI-48	48-bits	Unregistered EUI-48.

The AR.IPv4 and AR.EUI-48 fields are included in the AR_UNREGISTER_B message so that the Service Node can perform multiple overlapping Unregistrations.

B.5.4.7.2.6 AR_LOOKUP_S

Table B.55 shows the address resolution lookup message sent from the Service Node to the Base Node.

Table B.55 – AR_LOOKUP_S message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type. • For AR_LOOKUP_S = 4.
AR.IPv4	32-bits	IPv4 address to lookup.

B.5.4.7.2.7 AR_LOOKUP_B

Table B.56 shows the address resolution lookup response message sent from the Base Node to the Service Node.

Table B.56 – AR_LOOKUP_B message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type. • For AR_LOOKUP_B = 5.
AR.IPv4	32-bits	IPv4 address looked up.
AR.EUI-48	48-bits	EUI-48 for IPv4 address.
AR.Status	8-bits	Lookup status, indicating if the address was found or an error occurred. • 0 = found, AR.EUI-48 valid; • 1 = unknown, AR.EUI-48 undefined.

The lookup may fail if the requested address has not been registered. In that case, AR.Status will have a value other than zero and the contents of AR.EUI-48 will be undefined. The lookup is only successful when AR.Status is zero. In that case, the EUI-48 field contains the resolved address.

B.5.4.7.2.8 AR_MCAST_REG_S

Table B.57 shows the multicast address resolution register message sent from the Service Node to the Base Node.

Table B.57 – AR_MCAST_REG_S message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type. • For AR_MCAST_REG_S = 8.
AR.IPv4	32-bits	IPv4 multicast address to be registered.

B.5.4.7.2.9 AR_MCAST_REG_B

Table B.58 shows the multicast address resolution register acknowledgment message sent from the Base Node to the Service Node.

Table B.58 – AR_MCAST_REG_B message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type. For AR_MCAST_REG_B = 9.
AR.IPv4	32-bits	• IPv4 multicast address registered.
Reserved	2-bits	Reserved. Should be encoded as 0.
AR.LCID	6-bits	LCID assigned to this IPv4 multicast address.

The AR.IPv4 field is included in the AR_MCAST_REG_B message so that the Service Node can perform multiple overlapping registrations.

B.5.4.7.2.10 AR_MCAST_UNREG_S

Table B.59 shows the multicast address resolution unregister message sent from the Service Node to the Base Node.

Table B.59 – AR_MCAST_UNREG_S message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type. • For AR_MCAST_UNREG_S = 10.
AR.IPv4	32-bits	IPv4 multicast address to be unregistered.

B.5.4.7.2.11 AR_MCAST_UNREG_B

Table B.60 shows the multicast address resolution unregister acknowledgment message sent from the Base Node to the Service Node.

Table B.60 – AR_MCAST_UNREG_B message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type. • For AR_MCAST_UNREG_B = 11;
AR.IPv4	32-bits	IPv4 multicast address unregistered.

The AR.IPv4 field is included in the AR_MCAST_UNREG_B message so that the Service Node can perform multiple overlapping Unregistrations.

B.5.4.7.3 IPv4 packet format

B.5.4.7.3.1 General

The following PDU formats are used for transferring IPv4 packets between Service Nodes. Two formats are defined. The first format is for when header compression is not used. The second format is for Van Jacobson Header Compression.

B.5.4.7.3.2 IPv4 packet format, no negotiated header compression

When no header compression has been negotiated, the IPv4 packet is simply sent as is, without any header.

Table B.61 – IPv4 packet format without negotiated header compression

Name	Length	Description
IPv4.PKT	n-octets	The IPv4 Packet.

B.5.4.7.3.3 IPv4 packet format with VJ header compression

With Van Jacobsen header compression, a one-octet header is needed before the IPv4 packet.

Table B.62 – IPv4 packet format with VJ header compression negotiated

Name	Length	Description
IPv4.Type	2-bits	Type of compressed packet. • IPv4.Type = 0 – TYPE_IP; • IPv4.Type = 1 – UNCOMPRESSED_TCP; • IPv4.Type = 2 – COMPRESSED_TCP; • IPv4.Type = 3 – TYPE_ERROR.
IPv4.Seq	6-bits	Packet sequence number.
IPv4.PKT	n-octets	The IPv4 Packet.

The IPv4.Type value TYPE_ERROR is never sent. It is a pseudo packet type used to tell the decompressor that a packet has been lost.

B.5.4.7.4 Connection data

B.5.4.7.4.1 General

When a connection is established between Service Nodes for the transfer of IPv4 packets, data is also transferred in the connection request packets. This data allows the negotiation of compression and notification of the IPv4 address.

B.5.4.7.4.2 Connection data from the initiator

Table B.63 shows the connection data sent by the initiator.

Table B.63 – Connection signalling data sent by the initiator

Name	Length	Description
Reserved	6-bits	Should be encoded as 0 in this version of the IPV4 SSCS protocol.
Data.HC	2-bit	Header Compression. • Data.HC = 0 – No compression requested; • Data.HC = 1 – VJ Compression requested; • Data.HC = 2, 3 – Reserved for future versions of the specification.
Data.IPv4	32-bits	IPv4 address of the initiator

If the device accepts the connection, it should copy the Data.IPv4 address into a new table entry along with the negotiated Data.HC value.

B.5.4.7.4.3 Connection data from the responder

Table B.64 shows the connection data sent in response to the connection request.

Table B.64 – Connection signalling data sent by the responder

Name	Length	Description
Reserved	6-bits	Should be encoded as zero in this version of the IPV4 SSCS protocol.
Data.HC	2-bit	Header Compression negotiated. • Data.HC = 0 – No compression permitted; • Data.HC = 1 – VJ Compression negotiated; • Data.HC = 2, 3 – Reserved.

A header compression scheme can only be used when it is supported by both Service Nodes. The responder may only set Data.HC to 0 or the same value as that received from the initiator. When the same value is used, it indicates that the requested compression scheme has been negotiated and will be used for the connection. Setting Data.HC to 0 allows the responder to deny the request for that header compression scheme or force the use of no header compression.

B.5.4.8 Service access point

B.5.4.8.1 General

This clause defines the service access point used by the IPv4 layer to communicate with the IPV4 SSCS.

B.5.4.8.2 Opening and closing the IPv4 SSCS

B.5.4.8.2.1 General

The following primitives are used to open and close the IPv4 SSCS. The IPv4 SSCS may be opened once only. The IPv4 layer may close the IPv4 SSCS when the IPv4 interface is brought down. The IPv4 SSCS will also close the IPv4 SSCS when the underlying MAC connection to the Base Node has been lost.

B.5.4.8.2.2 CL_IPv4_ESTABLISH.request

The CL_IPv4_ESTABLISH.request primitive is passed from the IPv4 layer to the IPV4 SSCS. It is used when the IPv4 layer brings the interface up.

The semantics of this primitive are as follows:

CL_IPv4_ESTABLISH.request{}

On receiving this primitive, the IPV4 SSCS will form the address resolution connection to the Base Node and join the broadcast group used for receiving/transmitting broadcast packets.

B.5.4.8.2.3 CL_IPv4_ESTABLISH.confirm

The CL_IPv4_ESTABLISH.confirm primitive is passed from the IPV4 SSCS to the IPv4 layer. It is used to indicate that the IPv4 SSCS is ready to access IPv4 packets to be sent to peers.

The semantics of this primitive are as follows:

CL_IPv4_ESTABLISH.confirm{}

Once the IPv4 SSCS has established all the necessary connections and is ready to transmit and receive IPv4 packets, this primitive is passed to the IPv4 layer. If the IPV4 SSCS encounters an error while opening, it responds with a CL_IPv4_RELEASE.confirm primitive, rather than a CL_IPv4_ESTABLISH.confirm.

B.5.4.8.2.4 CL_IPv4_RELEASE.request

The CL_IPv4_RELEASE.request primitive is used by the IPv4 layer when the interface is put down. The IPV4 SSCS closes all connections so that no more IPv4 packets are received and all resources are released.

The semantics of this primitive are as follows:

CL_IPv4_RELEASE.request{}

Once the IPV4 SSCS has released all its connections and resources it returns a CL_IPv4_RELEASE.confirm.

B.5.4.8.2.5 CL_IPv4_RELEASE.confirm

The CL_IPv4_RELEASE.confirm primitive is used by the IPv4 SSCS to indicate to the IPv4 layer that the IPv4 SSCS has been closed. This can be as a result of a CL_IPv4_RELEASE.request primitive, a CL_IPv4_ESTABLISH.request primitive, or because the MAC layer indicates the address resolution connection has been lost, or the Service Node itself is no longer registered.

The semantics of this primitive are as follows:

CL_IPv4_RELEASE.confirm{result}

The result parameter has the meanings defined in Table B.B.3.

B.5.4.8.3 Unicast address management

B.5.4.8.3.1 General

The primitives defined here are used for address management, i.e., the registration and Unregistration of IPv4 addresses associated with this IPv4 SSCS.

When there are no IPv4 addresses associated with the IPv4 SSCS, the IPv4 SSCS will only send and receive broadcast and multicast packets; unicast packets may not be sent. However, this is sufficient for BOOTP/DHCP operation to allow the device to gain an IPv4 address. Once an IPv4 address has been registered, the IPv4 layer can transmit unicast packets that have a source address equal to one of its registered addresses.

B.5.4.8.3.2 CL_IPv4_REGISTER.request

This primitive is passed from the IPv4 layer to the IPv4 SSCS to register an IPv4 address.

The semantics of this primitive are as follows:

CL_IPv4_REGISTER.request{IPv4, netmask, gateway}

The IPv4 address is the address to be registered.

The netmask is the network mask, used to mask the network number from the address. The netmask is used by the IPv4 SSCS to determine whether the packet should be delivered directly or the gateway should be used.

The gateway is an IPv4 address of the gateway to be used for packets with the IPv4 local address but the destination address is not in the same subnetwork as the local address.

Once the IPv4 address has been registered to the Base Node, a CL_IPv4_REGISTER.confirm primitive is used. If the registration fails, the CL_IPv4_RELEASE.confirm primitive will be used.

B.5.4.8.3.3 CL_IPv4_REGISTER.confirm

This primitive is passed from the IPv4 SSCS to the IPv4 layer to indicate that a registration has been successful.

The semantics of this primitive are as follows:

CL_IPv4_REGISTER.confirm{IPv4}

The IPv4 address is the address that was registered.

Once registration has been completed, the IPv4 layer may send IPv4 packets using this source address.

B.5.4.8.3.4 CL_IPv4_UNREGISTER.request

This primitive is passed from the IPv4 layer to the IPv4 SSCS to unregister an IPv4 address.

The semantics of this primitive are as follows:

CL_IPv4_UNREGISTER.request{IPv4}

The IPv4 address is the address to be unregistered.

Once the IPv4 address has been unregistered to the Base Node, a CL_IPv4_UNREGISTER.confirm primitive is used. If the unregistration fails, the CL_IPv4_RELEASE.confirm primitive will be used.

B.5.4.8.3.5 CL_IPv4_UNREGISTER.confirm

This primitive is passed from the IPv4 SSCS to the IPv4 layer to indicate that an Unregistration has been successful.

The semantics of this primitive are as follows:

CL_IPv4_UNREGISTER.confirm{IPv4}

The IPv4 address is the address that was unregistered.

Once Unregistration has been completed, the IPv4 layer may not send IPv4 packets using this source address.

B.5.4.8.4 Multicast group management

B.5.4.8.4.1 General

This clause describes the primitives used to manage multicast groups.

B.5.4.8.4.2 CL_IPv4_IGMP_JOIN.request

This primitive is passed from the IPv4 layer to the IPv4 SSCS. It contains an IPv4 multicast address that is to be joined.

The semantics of this primitive are as follows:

CL_IPv4_IGMP_JOIN.request{IPv4 }

The IPv4 address is the IPv4 multicast group that is to be joined.

When the IPv4 SSCS receives this primitive, it will arrange for IPv4 packets sent to this group to be multicast in the OFDM TYPE 1 network and receive packets using this address to be passed to the IPv4 stack. If the IPv4 SSCS cannot join the group, it uses the CL_IPv4_IGMP_LEAVE.confirm primitive. Otherwise the CL_IPv4_IGMP_JOIN.confirm primitive is used to indicate success.

B.5.4.8.4.3 CL_IPv4_IGMP_JOIN.confirm

This primitive is passed from the IPv4 SSCS to the IPv4. It contains a result status and an IPv4 multicast address that was joined.

The semantics of this primitive are as follows:

CL_IPv4_IGMP_JOIN.confirm{IPv4}

The IPv4 address is the IPv4 multicast group that was joined. The IPv4 SSCS will start forwarding IPv4 multicast packets for the given multicast group.

B.5.4.8.4.4 CL_IPv4_IGMP_LEAVE.request

This primitive is passed from the IPv4 layer to the IPv4 SSCS. It contains an IPv4 multicast address to be left.

The semantics of this primitive are as follows:

CL_IPv4_IGMP_LEAVE.request{IPv4}

The IPv4 address is the IPv4 multicast group to be left. The IPv4 SSCS will stop forwarding IPv4 multicast packets for this group and may leave the OFDM TYPE 1 MAC multicast group.

B.5.4.8.4.5 CL_IPv4_IGMP_LEAVE.confirm

This primitive is passed from the IPv4 SSCS to the IPv4. It contains a result status and an IPv4 multicast address that was left.

The semantics of this primitive are as follows:

CL_IPv4_IGMP_LEAVE.confirm{IPv4, Result}

The IPv4 address is the IPv4 multicast group that was left. The IPv4 SSCS will stop forwarding IPv4 multicast packets for the given multicast group.

The Result takes a value from Table B.B.3.

This primitive can be used by the IPv4 SSCS as a result of a CL_IPv4_IGMP_JOIN.request, CL_IPv4_IGMP_LEAVE.request or because of an error condition resulting in the loss of the OFDM TYPE 1 MAC multicast connection.

B.5.4.8.5 Data transfer

B.5.4.8.5.1 General

The following primitives are used to send and receive IPv4 packets.

B.5.4.8.5.2 CL_IPv4_DATA.request

This primitive is passed from the IPv4 layer to the IPv4 SSCS. It contains one IPv4 packet to be sent.

The semantics of this primitive are as follows:

CL_IPv4_DATA.request{IPv4_PDU}

The IPv4_PDU is the IPv4 packet to be sent.

B.5.4.8.5.3 CL_IPv4_DATA.confirm

This primitive is passed from the IPv4 SSCS to the IPv4 layer. It contains a status indication and an IPv4 packet that has just been sent.

The semantics of this primitive are as follows:

CL_IPv4_DATA.confirm{IPv4_PDU, Result}

The IPv4_PDU is the IPv4 packet that was to be sent.

The Result value indicates whether the packet was sent or an error occurred. It takes a value from Table B.B.3.

B.5.4.8.5.4 CL_IPv4_DATA.indicate

This primitive is passed from the IPv4 SSCS to the IPv4 layer. It contains an IPv4 packet that has just been received.

The semantics of this primitive are as follows:

CL_IPv4_DATA.indicate{IPv4_PDU }

The IPv4_PDU is the IPv4 packet that was received.

B.5.5 IEC 61334-4-32 specific service convergence sublayer (IEC 61334-4-32 SSCS)

B.5.5.1 General

For all the service required, the IEC 61334-4-32 SSCS supports the DL_DATA primitives as defined in the IEC 61334-4-32 standard. IEC 61334-4-32 should be read at the same time as this clause, which is not standalone text.

B.5.5.2 Overview

The IEC 61334-4-32 SSCS provides convergence functions for applications that use IEC 61334-4-32 services. Implementations conforming to this SSCS shall offer all LLC basic and management services as specified in IEC 61334-4-32 (1996-09 Edition), subsections 2.2.1 and 2.2.3. Additionally, the IEC 61334-4-32 SSCS specified in this clause provides extra services that help mapping this connection-less IEC 61334-4-32 LLC protocol to the connection-oriented nature of MAC.

- A Service Node can only exchange data with the Base Node and not with other Service Nodes. This meets all the requirements of IEC 61334-4-32, which has similar restrictions.
- Each IEC 61334-4-32 SSCS session establishes a dedicated OFDM TYPE 1 MAC connection for exchanging unicast data with the Base Node.
- The Service Node SSCS session is responsible for initiating this connection to the Base Node. The Base Node SSCS cannot initiate a connection to a Service Node.
- Each IEC 61334-4-32 SSCS listens to an OFDM TYPE 1 broadcast MAC connection dedicated to the transfer of IEC 61334-4-32 broadcast data from the Base Node to the Service Nodes. This broadcast connection is used when applications in the Base Node using IEC 61334-4-32 services make a transmission request with the Destination_address used for broadcast or the broadcast SAP functions are used. When there are multiple SSCS sessions within a Service Node, one OFDM TYPE 1 broadcast MAC connection is shared by all the SSCS sessions.
- A CPCS session is always present with a IEC 61334-4-32 SSCS session. The SPCS sublayer functionality is as specified in clause B.5.2.2. Thus, the MSDUs generated by IEC 61334-4-32 SSCS are always less than *ClMTUSize* bytes and application messages shall not be longer than *ClMaxAppPktSize*.

B.5.5.3 Address allocation and connection establishment

Each 4-32 connection will be identified with the "Application unique identifier" that will be communicating through this 4-32 connection. It is the scope of the communication profile based on these lower layers to define the nature and rules for, this unique identifier. As long as the specification of the 4-32 Convergence layer concerns this identifier will be called the "Device Identifier".

The protocol stack as defined in IEC 61334 defines a Destination address to identify each device in the network. This Destination address is specified beyond the scope of the IEC 61334-4-32 document. However, it is used by the document. So that OFDM TYPE 1 devices can make use of the 4-32 layer, this Destination address is also required and is specified here. For more information about this Destination address, please see IEC 61334-4-1 section 4.3, MAC Addresses.

The Destination address has a scope of one OFDM TYPE 1 Subnetwork. The Base Node 4-32 SSCP layer is responsible for allocating these addresses dynamically and associating the Device Identifier of the Service Nodes SSCP session device with the allocated Destination address, according to the IEC-61334-4-1 standard. The procedure is as follows:

When the Service Node IEC 61334-4-32 SSCS session is opened by the application layer, it passes the Device Identifier of the device. The IEC 61334-4-32 SSCS session then establishes its unicast connection to the Base Node. This unicast connection uses the OFDM TYPE 1 MAC TYPE value TYPE_CL_432, as defined in Table B.B.1. The connection request packet sent from the Service Node to the Base Node contains a data parameter. This data parameter contains the Device Identifier. The format of this data is specified in clause B.5.5.4.2.

On receiving this connection request at the Base Node, the Base Node allocates a unique Subnetwork Destination address to the Service Nodes SSCS session. The Base Node sends back an OFDM TYPE 1 MAC connection response packet that contains a data parameter. This data parameter contains the allocated Destination address and the address being used by the Base Node itself. The format of this data parameter is defined in clause B.5.5.4.2. A 4-32 CL SAP primitive is used in the Base Node to indicate this new Service Node SSCS session mapping of Device Identifier and Destination_address to the 4-32 application running in the Base Node.

On receiving the connection establishment and the Destination_address passed in the OFDM TYPE 1 MAC connection establishment packet, the 4-32 SSCS session confirms to the application that the Convergence layer session has been opened and indicates the Destination_address allocated to the

Service Node SSCS session and the address of the Base Node. The Service Node also opens a OFDM TYPE 1 MAC broadcast connection with LCID equal to LCI_CL_432_BROADCAST, as defined in Table B.B.2, if no other SSCS session has already opened such a broadcast connection This connection is used to receive broadcast packets sent by the Base Node 4-32 Convergence layer to all Service Node 4-32 Convergence layer sessions.

If the Base Node has allocated all its available Destination_addresses, due to the exhaustion of the address space or implementation limits, it should simply reject the connection request from the Service Node. The Service Node may try to establish the connection again. However, to avoid overloading the OFDM TYPE 1 Subnetwork with such requests, it should limit such connection establishments to one attempt per minute when the Base Node rejects a connection establishment.

When the unicast connection between a Service Node and the Base Node is closed (e.g., because the Convergence layer on the Service Node is closed or the OFDM TYPE 1 MAC level connection between the Service Node and the Base Node is lost), the Base Node will deallocate the Destination_address allocated to the Service Node SSCS session. The Base Node will use a 4-32 CL SAP (CL_432_Leave.indication) primitive to indicate the deallocation of the Destination_address to the 4-32 application running on the Base Node

B.5.5.4 Connection establishment data format

B.5.5.4.1 General

As described in clause B.5.5.3, the MAC OFDM TYPE 1 connection data is used to transfer the Device Identifier to the Base Node and the allocated Destination_address to the Service Node SSCS session. This clause describes the format used for this data.

B.5.5.4.2 Service node to base node

The Service Node session passes the Device Identifier to the Base Node as part of the connection establishment request. The format of this message is shown in Table B.65.

Table B.65 – Connection signalling data sent by the service node

Name	Length	Description
Data.SN	n-Octets	Device Identifier. "COSEM logical device name" of the "Management logical device" of the DLMS/COSEM device as specified in the DLMS/COSEM, which will be communicating through this 4-32 connection.

B.5.5.4.3 Base Node to Service Node

The Base Node passes the allocated Destination_address to the Service Node session as part of the connection establishment request. It also gives its own address to the Service Node. The format of this message is shown in Table B.66.

Table B.66 – Connection signalling data sent by the base node

Name	Length	Description
Reserved	4-bits	Reserved. Should be encoded as zero in this version of the specification.
Data.DA	12-bits	Destination_address allocated to the Service Node.
Reserved	4-bits	Reserved. Should be encoded as zero in this version of the specification.
Data.BA	12-bits	Base_address used by the Base Node.

B.5.5.5 Packet format

The packet formats are used as defined in IEC 61334-4-32, clause 4, LLC Protocol Data Unit Structure (LLC_PDU).

B.5.5.6 Service access point

B.5.5.6.1 Opening and closing the convergence layer at the service node

B.5.5.6.1.1 CL_432_ESTABLISH.request

This primitive is passed from the application to the 4-32 Convergence layer. It is used to open a Convergence layer session and initiate the process of registering the Device Identifier with the Base Node and the Base Node allocating a Destination_address to the Service Node session.

The semantics of this primitive are as follows:

CL_432_ESTABLISH.request{ DeviceIdentifier }

The Device Identifier is that of the device to be registered with the Base Node.

If the Device Identifier is registered and the Convergence layer session is successfully opened, the primitive CL_432_ESTABLISH.confirm is used. If an error occurs the primitive CL_432_RELEASE.confirm is used.

B.5.5.6.1.2 CL_432_ESTABLISH.confirm

This primitive is passed from the 4-32 Convergence layer to the application. It is used to confirm the successful opening of the Convergence layer session and that data may now be passed over the Convergence layer.

The semantics of this primitive are as follows:

CL_432_ESTABLISH.confirm{ DeviceIdentifier, Destination_address, Base_address }

The Device Identifier is used to identify which CL_432_ESTABLISH.request this CL_432_ESTABLISH.confirm is for.

The Destination_address is the address allocated to the Service Node 4-32 session by the Base Node.

The Base_address is the address being used by the Base Node.

B.5.5.6.1.3 CL_432_RELEASE.request

This primitive is passed from the application to the 4-32 Convergence layer. It is used to close the Convergence layer and release any resources it may be holding.

The semantics of this primitive are as follows:

CL_432_RELEASE.request{Destination_address}

The Destination_address is the address allocated to the Service Node 4-32 session which is to be closed.

The Convergence layer will use the primitive CL_432_RELEASE.confirm when the Convergence layer session has been closed.

B.5.5.6.1.4 CL_432_RELEASE.confirm

This primitive is passed from the 4-32 Convergence layer to the application. The primitive tells the application that the Convergence layer session has been closed. This could be because of a CL_432_RELEASE.request or because an error has occurred, forcing the closure of the Convergence layer session.

The semantics of this primitive are as follows:

CL_432_RELEASE.confirm{Destination_address, result}

The Handle identifies the session which has been closed.

The result parameter has the meanings defined in Table B.B.3.

B.5.5.6.2 Opening and closing the convergence layer at the base node

No service access point primitives are defined at the Base Node for opening or closing the Convergence layer. None are required since the 4-32 application in the Base Node does not need to pass any information to the 4-32 Convergence layer in the Base Node.

B.5.5.6.3 Base node indications

B.5.5.6.3.1 General

The following primitives are used in the Base Node 4-32 Convergence layer to indicate events to the 4-32 application in the Base Node. They indicate when a Service Node session has joined or left the network.

B.5.5.6.3.2 CL_432_JOIN.indicate

CL_432_JOIN.indicate{ Device Identifier, Destination_address}

The Device Identifier is that of the device connected to the Service Node that has just joined the network.

The Destination_address is the address allocated to the Service Node by the Base Node.

B.5.5.6.3.3 CL_432_LEAVE.indicate

CL_432_LEAVE.indicate{Destination_address}

The Destination_address is the address of the Service Node session that just left the network.

B.5.5.6.4 Data transfer primitives

The data transfer primitives are used as defined in IEC 61334-4-32, sections 2.2, 2.3, 2.4 and 2.11, LLC Service Specification. As stated earlier, OFDM Type 1 432 SSCS make the use of IEC 61334-4-32 DL_Data service (.req, .conf, .ind) for carrying out all the data involved during data transfer.

B.5.6 IPv6 service-specific convergence sublayer (IPv6 SSCS)

B.5.6.1 Overview

B.5.6.1.1 General

The IPv6 convergence layer provides an efficient method for transferring IPv6 packets over the PRIME network.

A Service Node can pass IPv6 packets to the Base Node or directly to other Service Nodes.

By default, the Base Node acts as a router between the PRIME subnet and the backbone network. All the Base Nodes must have at least this connectivity capability. Any other node inside the Subnetwork can also act as a gateway. The Base Node could also act as a NAT router. However given the abundance of IPv6 addresses this is not expected. How the Base Node connects to the backbone is beyond the scope of this standard.

B.5.6.1.2 IPv6 unicast addressing assignment

- IPv6 Service Nodes (and Base Nodes) shall support the standard IPv6 protocol, as described in RFC 2460.

- IPv6 Service Nodes (and Base Nodes) shall support the standard IPv6 addressing architecture, as described in [IETF RFC 4291].
- IPv6 Service Nodes (and Base Nodes) shall support global unicast IPv6 addresses, link-local IPv6 addresses and multicast IPv6 addresses, as described in [IETF RFC 4291].
- IPv6 Service Nodes (and Base Nodes) shall support automatic address configuration using stateless address configuration [RFC 2462]. They may also support automatic address configuration using stateful address configuration [RFC 3315] and they may support manual configuration of IPv6 addresses. The decision of which address configuration scheme to use is deployment specific.
- Service Node shall support DHCPv6 client, when Base Nodes have to support DHCPv6 server as described in RFC 3315 for stateless address configuration

B.5.6.1.3 Address management in PRIME Subnetwork

Packets are routed in PRIME Subnetwork according to the node identifier NID. Node identifier is a combination of Service Node's LNID and SID (see clause B.4.2). The Base Node is responsible of assigning LNID to Service Nodes. During the registration process which leads to a LNID assignment to the related Service Node, the Base Node registers the Service Node EUI-48, and the assigned LNID together with SID.

At the convergence layer level, addressing is performed using the EUI-48 of the related Service Node. The role of the convergence sublayer is to resolve the IPv6 address into EUI-48 of the Service Node. This is done using the address resolution service set of the Base Node.

B.5.6.1.4 Role of the base node

At the convergence sublayer level, the Base Node maintains a table containing all the IPv6 unicast addresses and the EUI-48 related to them. One of the roles of the Base Node is to perform IPv6 to EUI-48 address resolution. Each Service Node belonging to the Subnetwork managed by the Base Node, registers its IPv6 address and EUI-48 address with the Base Node. Other Service Nodes can then query the Base Node to resolve an IPv6 address into a EUI-48 address. This requires the establishment of a dedicated connection to the Base Node for address resolution, which is shared by both IPv4 and IPv6 address resolution.

Optionally UDP/IPv6 headers may be compressed. Compression is negotiated as part of the connection establishment phase. Currently one header compression technique is described in the present specification that is used for transmission of IPv6 packets over IEEE 802.15.4 networks, as defined in [SP 800-57]. This is also known as LOWPAN_IPHC1.

The multicasting of IPv6 packets is supported using the MAC multicast mechanism.

B.5.6.2 IPv6 convergence layer

B.5.6.2.1 Overview

B.5.6.2.1.1 General

The convergence layer has a number of connection types. For address resolution there is a connection to the Base Node. For IPv6 data transfer there is one connection per destination node: the Base Node that acts as the IPv6 gateway to the outside world or another node in the same Subnetwork. This is shown in Figure B.83.

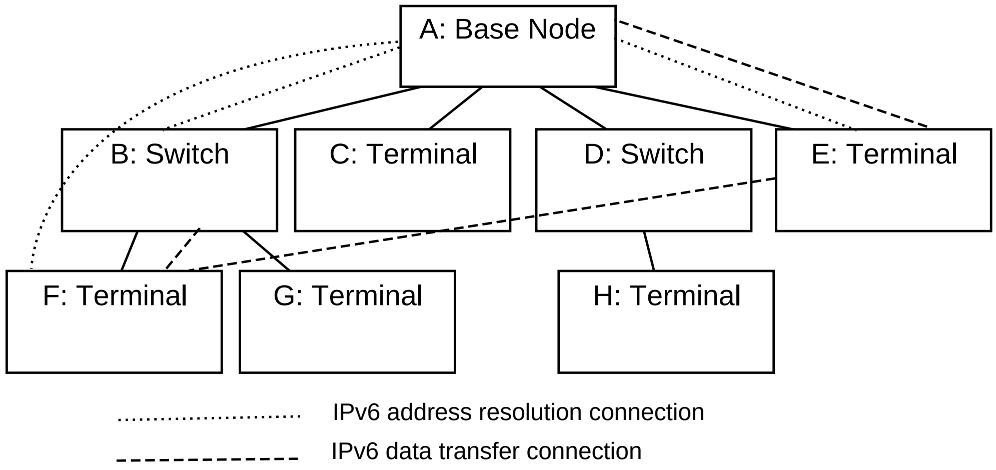


Figure B.83 – IPv6 SSCS connection example

Here, nodes B, E and F have address resolution connections to the Base Node. Node E has a data connection to the Base Node and node F. Node F is also has a data connection to node B. The figure does not show broadcast-traffic and multicast-traffic connections.

B.5.6.2.1.2 Routing in the subnetwork

Routing IPv6 packets is the scope of the Convergence layer. In other words, the convergence layer will decide whether the packet should be sent directly to another Service Node or forwarded to the configured gateway depending on the IPv6 destination address.

Although IPv6 is a connectionless protocol, the IPv6 convergence layer is connection-oriented. Once address resolution has been performed, a connection is established between the source and destination Service Nodes for the transfer of IP packets. This connection is maintained all the time the traffic is being transferred and may be removed after a period of inactivity.

B.5.6.2.1.3 SAR

The CPCS sublayer shall always be present with the IPv6 convergence layer allowing segmentation and reassembly facilities. The SAR sublayer functionality is given in clause B.5.2. Thus, the MSDUs generated by the IPv6 convergence layer are always less than ClMTUSize bytes and application messages are expected to be no longer than ClMaxAppPktSize.

B.5.6.3 IPv6 address configuration

B.5.6.3.1 Overview

The Service Nodes may use statically configured IPv6 addresses, link local addresses, stateless or stateful auto-configuration according to RFC 2462, or DHCPv6 to obtain IPv6 addresses. All the Nodes shall support the unicast link local address, in addition with other configured addresses below, and multicast addresses, if ever the node belong to multicast groups.

B.5.6.3.2 Interface identifier

In order to make use of stateless address auto configuration and link local addresses it is necessary to define how the Interface identifier, as defined in [IETF RFC 4291], is derived. Each PRIME node has a unique EUI-48. This EUI-48 is converted into an EUI-64 in the same way as for Ethernet networks as defined in RFC 2464. This EUI-64 is then used as the Interface Identifier.

B.5.6.3.3 IPv6 link local address configuration

The IPv6 Link local address of a PRIME interface is formed by appending the Interface Identifier as defined above to the Prefix FE80::/64.

B.5.6.3.4 Stateless address configuration

An IPv6 address prefix used for stateless auto configuration, as defined in [IETF RFC 4862], of a PRIME interface shall have a length of 64 bits. The IPv6 prefix is obtained by the Service Nodes from the Base Node via Router Advertisement messages, which are send periodically or on request by the Base Node.

B.5.6.3.5 Stateful address configuration

An IPv6 address can be alternatively configured using DHCPv6, as described in RFC 3315. DHCPv6 can provide a device with addresses assigned by a DHCPv6 server and other configuration information, which are carried in options.

B.5.6.3.6 Multicast address

IPv6 Service Nodes (and Base Nodes) shall support the multicast IPv6 addressing, as described in [IETF RFC 4291], clause 2.7.

B.5.6.3.7 Address resolution

B.5.6.3.7.1 Overview

The IPv6 layer will present the convergence layer with an IPv6 packet to be transferred. The convergence layer is responsible for determining which Service Node the packet should be delivered to, using the IPv6 addresses in the packet. The convergence layer shall then establish a connection to the destination if one does not already exist so that the packet can be transferred. Two classes of IPv6 addresses can be used and the following clause describes how these addresses are resolved into PRIME EUI-48 addresses. It should be noted that IPv6 does not have a broadcast address. However broadcasting is possible using multicast all nodes addresses.

B.5.6.3.7.2 Unicast address

B.5.6.3.7.2.1 General

IPv6 unicast addresses shall be resolved into PRIME unicast EUI-48 addresses. The Base Node maintains a central database Node of IPv6 addresses and EUI-48 addresses. Address resolution functions are performed by querying this database. The Service Node shall establish a connection to the address resolution service running on the Base Node, using the TYPE value TYPE_CL_IPv6_AR. No data should be passed in the connection establishment signalling. Using this connection, the Service Node can use two mechanisms as defined in the present specification.

B.5.6.3.7.2.2 Address registration and deregistration

A Service Node uses the AR_REGISTERv6_S message to register an IPv6 address and the corresponding EUI-48 address. The Base Node will acknowledge an AR_REGISTERv6_B message. The Service Node may register multiple IPv6 addresses for the same EUI-48.

A Service Node uses the AR_UNREGISTERv6_S message to unregister an IPv6 address and the corresponding EUI-48 address. The Base Node will acknowledge with an AR_UNREGISTERv6_B message.

When the address resolution connection between the Service Node and the Base Node is closed, the Base Node should remove all addresses associated with that connection.

B.5.6.3.7.2.3 Address lookup

A Service Node uses the AR_LOOKUPv6_S message to perform a lookup. The message contains the IPv6 address to be resolved. The Base Node will respond with an AR_LOOKUPv6_B message that contains an error code and, if there is no error, the EUI-48 associated with the IPv6 address. If the Base Node has multiple entries in its database Node for the same IPv6 address, the possible EUI-48 returned is undefined.

It should be noted that, for the link local addresses, due to the fact that the EUI-48 can be obtained from the IPv6 address, the lookup can simply return this value by extracting it from the IPv6 address.

B.5.6.3.7.3 Multicast address

Multicast IPv6 addresses are mapped to connection handles (ConnHandle) by the Convergence Layer.

To join a multicast group, CL uses the MAC_JOIN.request primitive with the IPv6 address specified in the data field. A corresponding MAC_JOIN.Confirm primitive will be generated by the MAC after completion of the join process. The MAC_Join.Confirm primitive will contain the result (success/failure) and the corresponding ConnHandle to be used by the CL. The MAC layer will handle the transfer of data for this connection using the appropriate LCIDs. To leave the multicast group, the CL at the service node shall use the MAC-LEAVE.Request{ConnHandle} primitive.

To send an IPv6 multicast packet, the CL will simply send the packet to the group, using the allocated ConnHandle. The ConnHandle is maintained while there are more packets to be sent. However, after Tmcast_reg seconds of not sending an IPv6 multicast packet to the group, the node should release the ConnHandle by using the MAC-LEAVE.Request primitive. The nominal value of Tmcast_reg is 10 minutes; however, other values may be used.

B.5.6.3.7.4 Retransmission of address resolution packets

The connection between the Service Node and the Base Node for address resolution is not reliable. The MAC ARQ is not used. The Service Node is responsible for making retransmissions if the Base Node does not respond in one second. It is not considered an error when the Base Node receives the same registration requests multiple times or is asked to remove a registration that does not exist. These conditions can be the result of retransmissions.

B.5.6.4 IPv6 packet transfer

For packets to be transferred, a connection needs to be established between the source and destination nodes. The IPv6 convergence layer will examine each IP packet to determine the destination EUI-48 address. If a connection to the destination has already been established, the packet is simply sent. To establish this, the convergence layer keeps a table for each connection it has with information shown in Table B.67. To use this table, it is first necessary to determine if the remote address is in the local subnet or if ever a gateway has to be used. The netmask associated with the local IP address is used to determine this. If the destination address is not in the local Subnetwork, the address of the gateway is used instead of the destination address when the table is searched.

Table B.67 – IPv6 convergence layer table entry

Parameter	Description
CL_IPv6_Con.Remote_IP	Remote IP address of this connection
CL_IPv6_Con.ConHandle	MAC Connection handle for the connection
CL_IPv6_Con.LastUsed	Timestamp of last packet received/transmitted
CL_IPv6_Con.HC	Header Compression scheme being used

The convergence layer may close a connection when it has not been used for an implementation-defined time period. When the connection is closed the entry for the connection is removed at both ends of the connection.

When a connection to the destination does not exist, more work is necessary. The address resolution service is used to determine the EUI-48 address of the remote IP address if it is local or the gateway associated with the local address if the destination address is in another subnet. When the Base Node replies with the EUI-48 address of the destination Service Node, a MAC connection is established to the remote device. The TYPE value of this connection is TYPE_CL_IPv6_UNICAST. The data passed in the request message is defined in clause B.5.6.8.3. The local IP address is provided so that the remote device can add the new connection to its cache of connections for sending data in the opposite direction. The use of header compression is also negotiated as part of the connection establishment. Once the connection has been established, the IP packet can be sent.

B.5.6.5 Segmentation and reassembly

The IPv6 convergence layer should support IPv6 packets with an MTU of 1500 bytes. This requires the use of the common part convergence sublayer segmentation and reassembly service.

B.5.6.6 Compression

Any PRIME device shall be capable of LOWPAN_IPHC IPv6 header compression/decompression. It may also be also capable of performing UDP compression/decompression. Thus UDP/IPv6 compression is negotiated.

No negotiation can take place for multicast packet. Nodes can only make use of mandatory compression capabilities.

Depending of the type of IPv6 address carried by the packet and the capabilities which are negotiated between the nodes involved in the data exchanges, IPv6 header compression is performed.

All the Service Nodes and the Base Node shall support IPv6 Header Compression using source and destination Addresses stateless compression as defined in [SP 800-57]. Source and destination IPv6 addresses using stateful compression and IPv6 Next header compression are negotiable.

B.5.6.7 Quality of service mapping

The PRIME MAC specifies that the contention-based access mechanism supports 4 priority levels (1-4). Level 1 is used for MAC signalling messages, but not exclusively so.

IPv6 packets include a Traffic Class field in the header to indicate the QoS the packet would like to receive. This traffic class can be used in the same way that IPv4 TOS (see [IEEE 802.3]). That is, three bits of the TOS indicate the IP Precedence. The following table specifies how the IP Precedence is mapped into the PRIME MAC priority.

Table B.68 – Mapping IPv6 precedence to PRIME MAC priority

IP Precedence	MAC Priority
000 – Routine	3
001 – Priority	3
010 – Immediate	2
011 – Flash	2
100 – Flash Override	1
P101 – Critical	1
110 – Internetwork Control	0
111 – Network Control	0

NOTE – At the MAC layer level the priority as stated in the Packet header field is the value assigned in this table minus 1, as the range of PKT.PRIO field is from 0 to 3.

B.5.6.8 Packet formats and connection data

B.5.6.8.1 Overview

This clause defines the format of convergence layer PDUs.

B.5.6.8.2 Address resolution PDU

B.5.6.8.2.1 General

The following PDUs are transferred over the address resolution connection between the Service Node and the Base Node. The following clauses define a number of AR.MSG values. All other values are reserved for later versions of this standard.

B.5.6.8.2.2 AR_REGISTERv6_S

Table B.69 shows the address resolution register message sent from the Service Node to the Base Node.

Table B.69 – AR_REGISTERv6_S message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type • For AR_REGISTERv6_S = 16
AR.IPv6	128-bits	IPv6 address to be registered
AR.EUI-48	48-bits	EUI-48 to be registered

B.5.6.8.2.3 AR_REGISTERv6_B

Table B.70 shows the address resolution register acknowledgment message sent from the Base Node to the Service Node.

Table B.70 – AR_REGISTERv6_B message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type • For AR_REGISTERv6_B = 17
AR.IPv6	128-bits	IPv6 address registered
AR.EUI-48	48-bits	EUI-48 registered

The AR.IPv6 and AR.EUI-48 fields are included in the AR_REGISTERv6_B message so that the Service Node can perform multiple overlapping registrations.

B.5.6.8.2.4 AR_UNREGISTERv6_S

Table B.71 shows the address resolution unregister message sent from the Service Node to the Base Node.

Table B.71 – AR_UNREGISTERv6_S message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type • For AR_UNREGISTERv6_S = 18
AR.IPv6	128-bits	IPv6 address to be unregistered
AR.EUI-48	48-bits	EUI-48 to be unregistered

B.5.6.8.2.5 AR_UNREGISTERv6_B

Table B.72 shows the address resolution unregister acknowledgment message sent from the Base Node to the Service Node.

Table B.72 – AR_UNREGISTERv6_B message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type • For AR_UNREGISTERv6_B = 19
AR.IPv6	128-bits	IPv6 address unregistered
AR.EUI-48	48-bits	EUI-48 unregistered

The AR.IPv6 and AR.EUI-48 fields are included in the AR_UNREGISTERv6_B message so that the Service Node can perform multiple overlapping unregistrations.

B.5.6.8.2.6 AR_LOOKUPv6_S

Table B.73 shows the address resolution lookup message sent from the Service Node to the Base Node.

Table B.73 – AR_LOOKUPv6_S message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type • For AR_LOOKUPv6_S = 20
AR.IPv6	128-bits	IPv6 address to lookup

B.5.6.8.2.7 AR_LOOKUPv6_B

Table B.74 shows the address resolution lookup response message sent from the Base Node to the Service Node.

Table B.74 – AR_LOOKUPv6_B message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type • For AR_LOOKUPv6_B = 21
AR.IPv6	128-bits	IPv6 address looked up
AR.EUI-48	48-bits	EUI-48 for IPv6 address
AR.Status	8-bits	Lookup status, indicating if the address was found or an error occurred. • 0 = found, AR.EUI-48 valid. • 1 = unknown, AR.EUI-48 undefined

The lookup may fail if the requested address has not been registered. In that case, AR.Status will have a value equal to 1, and the contents of AR.EUI-48 will be undefined. The lookup is only successful when AR.Status is zero. In that case, the EUI-48 field contains the resolved address.

B.5.6.8.2.8 AR_MCAST_REGv6_S

Table B.75 shows the multicast address resolution register message sent from the Service Node to the Base Node.

Table B.75 – AR_MCAST_REGv6_S message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type - For AR_MCAST_REGv6_S = 24
AR.IPv6	128-bits	IPv6 multicast address to be registered

B.5.6.8.2.9 AR_MCAST_REGv6_B

Table B.76 shows the multicast address resolution register acknowledgment message sent from the Base Node to the Service Node.

Table B.76 – AR_MCAST_REGv6_B message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type - For AR_MCAST_REGv6_B = 25
AR.IPv6	128-bits	IPv6 multicast address registered
Reserved	2-bits	Reserved. Should be encoded as 0.
AR.LCID	6-bits	LCID assigned to this IPv6 multicast address

The AR.IPv6 field is included in the AR_MCAST_REGv6_B message so that the Service Node can perform multiple overlapping registrations.

B.5.6.8.2.10 AR_MCAST_UNREGv6_S

Table B.77 shows the multicast address resolution unregister message sent from the Service Node to the Base Node.

Table B.77 – AR_MCAST_UNREGv6_S message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type - For AR_MCAST_UNREGv6_S = 26
AR.IPv6	128-bits	IPv6 multicast address to be unregistered

B.5.6.8.2.11 AR_MCAST_UNREGv6_B

Table B.78 shows the multicast address resolution unregister acknowledgment message sent from the Base Node to the Service Node.

Table B.78 – AR_MCAST_UNREGv6_B message format

Name	Length	Description
AR.MSG	8-bits	Address Resolution Message Type - For AR_MCAST_UNREGv6_B = 27
AR.IPv6	128-bits	IPv6 multicast address unregistered

The AR.IPv6 field is included in the AR_MCAST_UNREGv6_B message so that the Service Node can perform multiple overlapping unregistrations.

B.5.6.8.3 IPv6 packet format

B.5.6.8.3.1 General

The following PDU formats are used for transferring IPv6 packets between Service Nodes.

B.5.6.8.3.2 No negotiated header compression

When no header compression take place, the IP packet is simply sent as it is, without any header.

Table B.79 – IPv6 packet format without negotiated header compression

Name	Length	Description
IPv6.PKT	n-octets	The IPv6 Packet

B.5.6.8.3.3 Header compression

When LOWPAN_IPHC1 header compression takes place, and the next header compression is negotiated, the UDP/IPv6 packet is sent as shown in Table B.80.

Table B.80 – UDP/IPv6 packet format with LOWPAN_IPHC1 header compression and LOWPAN_NHC

Name	Length	Description
IPv6.IPHC	2-octet	Dispatch + LOWPAN_IPHC encoding. With bit 5=1 indicating that the next is compressed, using LOWPAN_NHC format
IPv6.ncIPv6	n.m-octets	Non-Compressed IPv6 fields (or elided)
IPv6.HC_UDP	1-octet	Next header encoding
IPv6.ncUDP	n.m-octets	Non-Compressed UDP fields
Padding	0.m-octets	Padding to byte boundary
IPv6.DATA	n-octets	UDP data

Note that these fields are not necessarily aligned to byte boundaries. For example the IPv6.ncIPv6 field can be any number of bits. The IPv6.IPHC_UDP field follows directly afterwards, without any padding. Padding is only applied at the end of the complete compressed UDP/IPv6 header such that the UDP data is byte aligned.

When the IPv6 packet contains data other than UDP the following packet format is used as shown in Table B.81.

Table B.81 – IPv6 packet format with LOWPAN_IPHC negotiated header compression

Name	Length	Description
IPv6.IPHC	2-octet	HC encoding. Bits 5 contain 0 indicating the next header byte is not compressed.
IPv6.ncIPv6	n.m-octets	Non-Compressed IPv6 fields
Padding	0.m-octets	Padding to byte boundary
IPv6.DATA	n-octets	IP Data

B.5.6.8.4 Connection data

B.5.6.8.4.1 Overview

When a connection is established between Service Nodes for the transfer of IP packets, data is also transferred in the connection request packets. This data allows the negotiation of compression and notification of the IP address.

B.5.6.8.4.2 Connection data from the initiator

Table B.82 shows the connection data sent by the initiator.

Table B.82 – IPv6 connection signalling data sent by the initiator

Name	Length	Description
Reserved	6-bits	Should be encoded as zero in this version of the convergence layer protocol
Data.HCNH	2-bit	Header Compression negotiated • Data.HC = 0 – No compression requested • Data.HC = 1 – LOWPAN_NH • Data.HC = 2 – stateful address compression. • Data.HC = 3 – LOWPAN_NH and stateful address compression.
Data.IPv6	128-bits	IPv6 address of the initiator

If the device accepts the connection, it should copy the Data.IPv6 address into a new table entry along with the negotiated Data.HC value.

B.5.6.8.4.3 Connection data from the responder

Table B.83 shows the connection data sent in response to the connection request.

Table B.83 – IPv6 connection signalling data sent by the responder

Name	Length	Description
Reserved	6-bits	Should be encoded as zero in this version of the convergence layer protocol
Data.HC	2-bit	Header Compression negotiated • Data.HC = 0 – No compression requested: NOTE – When stateless address compression is used all nodes shall support it. When the stateless address compression is not used then the node notify by this value, its compression capability. Data.HC = 1 – LOWPAN_NH • Data.HC = 2 – stateful address compression. • Data.HC = 3 – LOWPAN_NH and stateful address compression.

All nodes support stateless address compression.

The next header compression scheme and stateful address compression can only be used when it is supported by both Service Nodes. The responder may only set Data.HC to the same value as that received from the initiator or a value lower than the one received. When the same value is used, it indicates that the requested compression scheme has been negotiated and will be used for the connection. Setting Data.HC to lower value allows the responder to deny the request for that header compression scheme.

B.5.6.9 Service access point

B.5.6.9.1 Overview

This clause defines the service access point used by the IPv6 layer to communicate with the IPv6 convergence layer.

B.5.6.9.2 Opening and closing the convergence layer

The following primitives are used to open and close the convergence layer. The convergence layer may be opened once only. The IPv6 layer may close the convergence layer when the IPv6 interface is brought down. The convergence layer will also close the convergence layer when the underlying MAC connection to the Base Node has been lost.

B.5.6.9.2.1 CL_IPv6_Establish.request

The CL_IPv6_ESTABLISH.request primitive is passed from the IPv6 layer to the IPv6 convergence layer. It is used when the IPv6 layer brings the interface up.

The semantics of this primitive are as follows:

CL_IPv6_ESTABLISH.request{}

On receiving this primitive, the convergence layer will form the address resolution connection to the Base Node.

B.5.6.9.2.2 CL_IPv6_Establish.confirm

The CL_IPv6_ESTABLISH.confirm primitive is passed from the IPv6 convergence layer to the IPv6 layer. It is used to indicate that the convergence layer is ready to access IPv6 packets to be sent to peers.

The semantics of this primitive are as follows:

CL_IPv6_ESTABLISH.confirm{}

Once the convergence layer has established all the necessary connections and is ready to transmit and receive IPv6 packets, this primitive is passed to the IPv6 layer. If the convergence layer encounters an error while opening, it responds with a CL_IPv6_RELEASE.confirm primitive, rather than a CL_IPv6_ESTABLISH.confirm.

B.5.6.9.2.3 CL_IPv6_Release.request

The CL_IPv6_RELEASE.request primitive is used by the IPv6 layer when the interface is put down. The convergence layer closes all connections so that no more IPv6 packets are received and all resources are released.

The semantics of this primitive are as follows:

CL_IPv6_RELEASE.request{}

Once the convergence layer has released all its connections and resources it returns a CL_IPv6_RELEASE.confirm.

B.5.6.9.2.4 CL_IPv6_Release.confirm

The CL_IPv6_RELEASE.confirm primitive is used by the IPv6 convergence layer to indicate to the IPv6 layer that the convergence layer has been closed. This can be as a result of a CL_IPv6_RELEASE.request primitive, a CL_IPv6_ESTABLISH.request primitive, or because the

MAC layer indicates the address resolution connection has been lost, or the Service Node itself is no longer registered.

The semantics of this primitive are as follows:

CL_IPv6_RELEASE.confirm{result}

The result parameter has the meanings defined in Table B.112.

B.5.6.9.3 Unicast address management

B.5.6.9.3.1 General

The primitives defined here are used for address management, i.e., the registration and unregistration of IPv6 addresses associated with this convergence layer.

When there are no IPv6 addresses associated with the convergence layer, the convergence layer will only send and receive multicast packets; unicast packets may not be sent. However, this is sufficient for various address discovery protocols to be used to gain an IPv6 address. Once an IPv6 address has been registered, the IPv6 layer can transmit unicast packets that have a source address equal to one of its registered addresses.

B.5.6.9.3.2 CL_IPv6_Register.request

This primitive is passed from the IPv6 layer to the IPv6 convergence layer to register an IPv6 address.

The semantics of this primitive are as follows:

CL_IPv6_REGISTER.request{ipv6, netmask, gateway}

The ipv6 address is the address to be registered.

The netmask is the network mask, used to mask the network number from the address. The netmask is used by the convergence layer to determine whether the packet should deliver directly or the gateway should be used.

The IPv6 address of the gateway, to which packets with destination address that are not in the same subnet as the local address are to be sent.

Once the IPv6 address has been registered to the Base Node, a CL_IPv6_REGISTER.confirm primitive is used. If the registration fails, the CL_IPv6_RELEASE.confirm primitive will be used.

B.5.6.9.3.3 CL_IPv6_Register.confirm

This primitive is passed from the IPv6 convergence layer to the IPv6 layer to indicate that a registration has been successful.

The semantics of this primitive are as follows:

CL_IPv6_REGISTER.confirm{ipv6}

The ipv6 address is the address that was registered.

Once registration has been completed, the IPv6 layer may send IPv6 packets using this source address.

B.5.6.9.3.4 CL_IPv6_Unregister.request

This primitive is passed from the IPv6 layer to the IPv6 convergence layer to unregister an IPv6 address.

The semantics of this primitive are as follows:

CL_IPv6_UNREGISTER.request{ipv6}

The ipv6 address is the address to be unregistered.

Once the IPv6 address has been unregistered to the Base Node, a CL_IPv6_UNREGISTER.confirm primitive is used. If the registration fails, the CL_IPv6_RELEASE.confirm primitive will be used.

B.5.6.9.3.5 Unregister.confirm

This primitive is passed from the IPv6 convergence layer to the IPv6 layer to indicate that an unregistration has been successful.

The semantics of this primitive are as follows:

CL_IPv6_UNREGISTER.confirm{ipv6}

The IPv6 address is the address that was unregistered.

Once unregistration has been completed, the IPv6 layer may not send IPv6 packets using this source address.

B.5.6.9.4 Multicast group management

B.5.6.9.4.1 General

This clause describes the primitives used to manage multicast groups.

B.5.6.9.4.2 CL_IPv6_MUL_Join.request

This primitive is passed from the IPv6 layer to the IPv6 convergence layer. It contains an IPv6 multicast address that is to be joined.

The semantics of this primitive are as follows:

CL_IPv6_MUL_JOIN.request{IPv6 }

The IPv6 address is the IPv6 multicast group that is to be joined.

When the convergence layer receives this primitive, it will arrange for IP packets sent to this group to be multicast in the PRIME network and receive packets using this address to be passed to the IPv6 stack. If the convergence layer cannot join the group, it uses the CL_IPv6_MUL_LEAVE.confirm primitive. Otherwise the CL_IPv6_MUL_JOIN.confirm primitive is used to indicate success.

B.5.6.9.4.3 CL_IPv6_MUL_Join.confirm

This primitive is passed from the IPv6 convergence layer to the IPv6. It contains a result status and an IPv6 multicast address that was joined.

The semantics of this primitive are as follows:

CL_IPv6_MUL_JOIN.confirm{IPv6}

The IPv6 address is the IPv6 multicast group that was joined. The convergence layer will start forwarding IPv6 multicast packets for the given multicast group.

B.5.6.9.4.4 CL_IPv6_MUL_Leave.request

This primitive is passed from the IPv6 layer to the IPv6 convergence layer. It contains an IPv6 multicast address to be left.

The semantics of this primitive are as follows:

CL_IPv6_MUL_LEAVE.request{IPv6}

The IPv6 address is the IPv6 multicast group to be left. The convergence layer will stop forwarding IPv6 multicast packets for this group and may leave the PRIME MAC multicast group.

B.5.6.9.4.5 CL_IPv6_MUL_Leave.confirm

This primitive is passed from the IPv6 convergence layer to the IPv6. It contains a result status and an IPv6 multicast address that was left.

The semantics of this primitive are as follows:

CL_IPv6_MUL_LEAVE.confirm{IPv6, Result}

The IPv6 address is the IPv6 multicast group that was left. The convergence layer will stop forwarding IPv6 multicast packets for the given multicast group.

The Result takes a value from Table B.B.3.

This primitive can be used by the convergence layer as a result of a CL_IPv6_MUL_JOIN.request, CL_IPv6_MUL_LEAVE.request or because of an error condition resulting in the loss of the PRIME MAC multicast connection.

B.5.6.9.5 Data transfer

B.5.6.9.5.1 General

The following primitives are used to send and receive IPv6 packets.

B.5.6.9.5.2 CL_IPv6_DATA.request

This primitive is passed from the IPv6 layer to the IPv6 convergence layer. It contains one IPv6 packet to be sent.

The semantics of this primitive are as follows:

CL_IPv6_DATA.request{IPv6_PDU}

The IPv6_PDU is the IPv6 packet to be sent.

B.5.6.9.5.3 CL_IPv6_DATA.confirm

This primitive is passed from the IPv6 convergence layer to the IPv6 layer. It contains a status indication and an IPv6 packet that has just been sent.

The semantics of this primitive are as follows:

CL_IPv6_DATA.confirm{IPv6_PDU, Result}

The IPv6_PDU is the IPv6 packet that was to be sent.

The Result value indicates whether the packet was sent or an error occurred. It takes a value from Table B.B.3.

B.5.6.9.5.4 CL_IPv6_DATA.indicate

This primitive is passed from the IPv6 convergence layer to the IPv6 layer. It contains an IPv6 packet that has just been received.

The semantics of this primitive are as follows:

CL_IPv6_DATA.indicate{IPv6_PDU }

The IPv6_PDU is the IPv6 packet that was received.

B.6 Management plane

B.6.1 Introduction

This chapter specifies the Management Plane functionality. The picture below highlights the position of Management Plan in overall protocol architecture

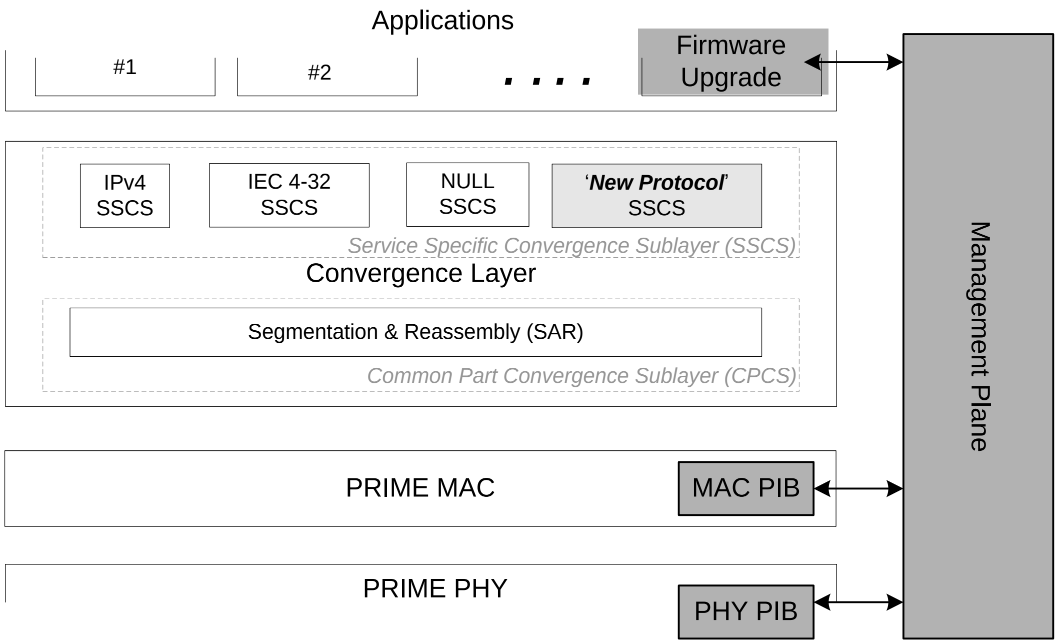


Figure B.84 – Management plane. Introduction

All nodes shall implement the management plane functionality enumerated in this clause. Management plane enables a local or remote control entity to perform actions on a Node.

Present version of this specification enumerates management plane functions for Node management and firmware upgrade. Future versions may include additional management functions.

- To enable access to management functions on a Service Node, Base Node shall open a management connection after successful completion of registration (refer to B.6.4).
- The Base Node may open such a connection either immediately on successful registration or sometime later.
- Unicast management connection shall be identified with CON.TYPE = TYPE_CL_MGMT.
- Multicast management connections can also exist. At the time of writing of this document, multicast management connection shall only be used for firmware upgrade.
- There shall be no broadcast management connection.
- In case Service Node supports ARQ connections, the Base Node shall preferentially try to open an ARQ connection for management functions.
- Management plane functions shall use NULL SSCS as specified in clause B.5.3.

B.6.2 Node management

B.6.2.1 General

Node management is accomplished through a set of attributes. Attributes are defined for both PHY and MAC layers. The set of these management attributes is called PLC Information Base (PIB). Some attributes are read-only while others are read-write.

PIB Attribute identifiers are 16 bit values. This allows for up to 65535 PIB Attributes to be specified.

- PIB Attribute identifier values from 0 to 32767 are open to be standardized. No proprietary attributes may have identifiers in this range.

- Values in the range 32768 to 65535 are open for vendor specific usage.

PIB Attributes identifiers in standard range (0 to 32767) that are not specified in this version are reserved for future use.

NOTE – PIB attribute tables below indicate type of each attribute. For integer types the size of the integer has been specified in bits. An implementation may use a larger integer for an attribute; however, it must not use a smaller size.

B.6.2.2 PHY PIB attributes

B.6.2.2.1 General

The PHY layer implementation in each device may optionally maintain a set of attributes which provide detailed information about its working. The PHY layer attributes are part of the PLC Information Base (PIB).

B.6.2.2.2 Statistical attributes

The PHY may provide statistical information for management purposes. Next table lists the statistics that PHY should make available to management entities across the PLME_GET primitive. The Id field in this table is the service parameter of the PLME_GET primitive specified in clause B.3.10.4/ [ITU-T G.9955].

Table B.84 – PHY read-only variables that provide statistical information

Attribute Name	Size (in bits)	Id	Description
phyStatsCRCIncorrectCount	16	0x00A0	Number of bursts received on the PHY layer for which the CRC was incorrect.
phyStatsCRCFailCount	16	0x00A1	Number of bursts received on the PHY layer for which the CRC was correct, but the *Protocol* field of PHY header had an invalid value. This count would reflect number of times corrupt data was received and the CRC calculation failed to detect it.
phyStatsTxDropCount	16	0x00A2	Number of times when PHY layer received new data to transmit (PHY_DATA.request) and had to either overwrite on existing data in its transmit queue or drop the data in new request due to full queue.
phyStatsRxDropCount	16	0x00A3	Number of times when PHY layer received new data on the channel and had to either overwrite on existing data in its receive queue or drop the newly received data due to full queue.
phyStatsRxTotalCount	32	0x00A4	Total number of PPDUs correctly decoded. Useful for PHY layer test cases, to estimate the FER.
phyStatsBlkAvgEvm	16	0x00A5	Exponential moving average of the EVM over the past 16 PPDUs, as returned by the PHY_SNR primitive. Note that the PHY_SNR primitive returns a 3-bit number in dB scale. So first each 3-bit dB number is converted to linear scale (number k goes to 2^(k/2)), yielding a 7 bit number with 3 fractional bits. The result is just accumulated over 16 PPDUs and reported.

Table B.84 – PHY read-only variables that provide statistical information

Attribute Name	Size (in bits)	Id	Description

phyEmaSmoothing	8	0x00A8	Smoothing factor divider for values that are updated as exponential moving average (EMA). Next value is
			`Vnext = S*NewSample+(1-S)*Vprev`
			Where
			`S=1/(2^phyEMASmoothing).`

B.6.2.2.3 Implementation attributes

It is possible to implement PHY functions conforming to this specification in multiple ways. The multiple implementation options provide some degree of unpredictability for MAC layers. PHY implementations may optionally provide specific information on parameters which are of interest to MAC across the PLME_GET primitive. A list of such parameters which maybe queried across the PLME_GET primitives by MAC is provided in Table B.85 – All of the attributes listed in Table B.85 – are implementation constants and shall not be changed.

Table B.85 – PHY read-only parameters, providing information on specific implementation

Attribute Name	Size (in bits)	Id	Description
phyTxQueueLen	10	0x00B0	Number of concurrent MPDUs that the PHY transmit buffers can hold.
phyRxQueueLen	10	0x00B1	Number of concurrent MPDUs that the PHY receive buffers can hold.
phyTxProcessingDelay	20	0x00B2	Time elapsed from the instance when data is received on MAC-PHY communication interface to the time when it is put on the physical channel. This shall not include communication delay over the MAC-PHY interface. Value of this attribute is in unit of microseconds.
phyRxProcessingDelay	20	0x00B3	Time elapsed from the instance when data is received on physical channel to the time when it is made available to MAC across the MAC-PHY communication interface. This shall not include communication delay over the MAC-PHY interface. Value of this attribute is in unit of microseconds.
phyAgcMinGain	8	0x00B4	Minimum gain for the AGC <= 0dB.
phyAgcStepValue	3	0x00B5	Distance between steps in dB <= 6dB.
phyAgcStepNumber	8	0x00B6	Number of steps so that phyAgcMinGain +((phyAgcStepNumber – 1) * phyAgcStepValue) >= 21dB.

B.6.2.3 MAC PIB attributes

B.6.2.3.1 General

NOTE – Note that the "M"(Mandatory) column in the tables below specifies if the PIB attributes are mandatory for all devices (both Service Node and Base Node, specified as "All"), only for Service Nodes ("SN"), only for Base Nodes ("BN") or not mandatory at all ("No").

B.6.2.3.2 MAC variable attributes

MAC PIB variables include the set of PIB attributes that influence the functional behaviour of an implementation. These attributes may be defined external to the MAC, typically by the management entity and implementations may allow changes to their values during normal running, i.e., even after the device start-up sequence has been executed.

An external management entity can have access to these attributes through the MLME_GET (B.4.5.5.7) and MLME_SET (B.4.5.5.9) set of primitives. The Id field in the following table would be the *PIBAttribute* that needs to be passed MLME SAP while working on these parameters.

Table B.86 – Table of MAC read-write variables

Attribute Name	Id	Type	M	Valid Range	Description	Def.
macMinSwitchSearchTime	0x0010	Integer8	No	16 – 32 seconds	Minimum time for which a Service Node in *Disconnected* status should scan the channel for Beacons before it can broadcast PNPDU. This attribute is not maintained in Base Nodes.	24
macMaxPromotionPdu	0x0011	Integer8	No	1 – 4	Maximum number of PNPDUs that may be transmitted by a Service Node in a period of *macPromotionPduTxPeriod* seconds. This attribute is not maintained in Base Node.	2
macPromotionPduTxPeriod	0x0012	Integer8	No	2 – 8 seconds	Time quantum for limiting a number of PNPDUs transmitted from a Service Node. No more than *macMaxPromotionPdu* may be transmitted in a period of *macPromotionPduTxPeriod* seconds.	5
macBeaconsPerFrame	0x0013	Integer8	BN	1 – 5	Maximum number of beacon-slots that may be provisioned in a frame. This attribute is maintained in Base Nodes.	5

Table B.86 – Table of MAC read-write variables

Attribute Name	Id	Type	M	Valid Range	Description	Def.
macSCPMaxTxAttempts	0x0014	Integer8	No	2 – 5	Number of times the CSMA algorithm would attempt to transmit requested data when a previous attempt was withheld due to PHY indicating channel busy.	5
macCtlReTxTimer	0x0015	Integer8	No	2 – 20 seconds	Number of seconds for which a MAC entity waits for acknowledgement of receipt of MAC control packet from its peer entity. On expiry of this time, the MAC entity may retransmit the MAC control packet.	15
macMaxCtlReTx	0x0018	Integer8	No	3 – 5	Maximum number of times a MAC entity will try to retransmit an unacknowledged MAC control packet. If the retransmit count reaches this maximum, the MAC entity shall abort further attempts to transmit the MAC control packet.	3
macEMASmoothing	0x0019	Integer8	All	0 – 7	Smoothing factor divider for values that are updated as exponential moving average (EMA). Next value is V*next* = S*NewSample+ (1–S)*V*prev* Where S=1/(2^macEMASmoothing).	3

Table B.87 – Table of MAC read-only variables

Attribute Name	Id	Type	M	Valid Range	Description	Def.
macSCPRBO	0x0016	Integer8	No	1 – 15 symbols	Random backoff period for which an implementation should delay the start of channel-sensing iterations when attempting to transmit data in SCP. This is a 'read-only' attribute.	–
macSCPChSenseCount	0x0017	Integer8	No	2 – 5	Number of times for which an implementation has to perform channel-sensing. This is a 'read-only' attribute.	–

B.6.2.3.3 Functional attributes

Some PIB attributes belong to the functional behaviour of MAC. They provide information on specific aspects. A management entity can only read their present value using the MLME_GET primitives. The value of these attributes cannot be changed by a management entity through the MLME_SET primitives.

The Id field in the table below would be the *PIBAttribute* that needs to be passed MLME_GET SAP for accessing the value of these attributes.

Table B.88 – Table of MAC read-only variables that provide functional information

Attribute Name	Id	Type	M	Valid Range	Description
macLNID	0x0020	Integer16	SN	0 – 16383	LNID allocated to this Node at time of its registration.
MacLSID	0x0021	Integer8	SN	0 – 255	LSID allocated to this Node at time of its promotion. This attribute is not maintained if a Node is in a *Terminal* functional state.
MacSID	0x0022	Integer8	SN	0 – 255	SID of the Switch Node through which this Node is connected to the Subnetwork. This attribute is not maintained in a Base Node.
MacSNA	0x0023	EUI-48	SN		Subnetwork address to which this Node is registered. The Base Node returns the SNA it is using.
MacState	0x0024	Enumerate	SN		Present functional state of the Node.
				0	DISCONNECTED.
				1	TERMINAL.
				2	SWITCH.
				3	BASE.
MacSCPLength	0x0025	Integer16	SN		The SCP length, in symbols, in present frame.

Table B.88 – Table of MAC read-only variables that provide functional information

Attribute Name	Id	Type	M	Valid Range	Description
MacNodeHierarchyLevel	0x0026	Integer8	SN	0 – 63	Level of this Node in Subnetwork hierarchy.
MacBeaconSlotCount	0x0027	Integer8	SN	0 – 7	Number of beacon-slots provisioned in present frame structure.
macBeaconRxSlot	0x0028	Integer8	SN	0 – 7	Beacon Slot on which this device's Switch Node transmits its beacon. This attribute is not maintained in a Base Node.
MacBeaconTxSlot	0x0029	Integer8	SN	0 – 7	Beacon Slot in which this device transmits its beacon. This attribute is not maintained in Service Nodes that are in a *Terminal* functional state.
MacBeaconRxFrequency	0x002A	Integer8	SN	0 – 31	Number of frames between receptions of two successive beacons. A value of 0x0 indicates beacons are received in every frame. This attribute is not maintained in Base Node.
MacBeaconTxFrequency	0x002B	Integer8	SN	0 – 31	Number of frames between transmissions of two successive beacons. A value of 0x0 indicates beacons are transmitted in every frame. This attribute is not maintained in Service Nodes that are in a *Terminal* functional state.

Table B.88 – Table of MAC read-only variables that provide functional information

Attribute Name	Id	Type	M	Valid Range	Description

MacCapabilities	0x002C	Integer16	All	Bitmap	Bitmap of MAC capabilities of a given device. This attribute shall be maintained on all devices. Bits in sequence of right-to-left shall have the following meaning:
					Bit 0: Switch Capable;
					Bit 1: Packet Aggregation;
					Bit 2: Contention Free Period;
					Bit 3: Direct connection;
					Bit 4: Multicast;
					Bit 5: PHY Robustness Management;
					Bit 6: ARQ;
					Bit 7: Reserved for future use;
					Bit 8: Direct Connection Switching;
					Bit 9: Multicast Switching Capability;
					Bit 10: PHY Robustness Management Switching Capability;
					Bit11: ARQ Buffering Switching Capability;
					Bits 12 to 15: Reserved for future use.

B.6.2.3.4 Statistical attributes

The MAC layer shall provide statistical information for management purposes. Table B.89 lists the statistics MAC shall make available to management entities across the MLME_GET primitive.

The Id field in table below would be the *PIBAttribute* that needs to be passed MLME_GET SAP for accessing the value of these attributes.

Table B.89 – Table of MAC read-only variables that provide statistical information

Attribute Name	Id	M	Type	Description
macTxDataPktCount	0x0040	No	Integer32	Count of successfully transmitted MSDUs.
MacRxDataPktCount	0x0041	No	Integer32	Count of successfully received MSDUs whose destination address was this Node.
MacTxCtrlPktCount	0x0042	No	Integer32	Count of successfully transmitted MAC control packets.
MacRxCtrlPktCount	0x0043	No	Integer32	Count of successfully received MAC control packets whose destination address was this Node.
MacCSMAFailCount	0x0044	No	Integer32	Count of failed CSMA transmitted attempts.
MacCSMAChBusy Count	0x0045	No	Integer32	Count of number of times this Node had to back off SCP transmission due to channel busy state.

B.6.2.3.5 MAC list attributes

MAC layer shall make certain lists available to the management entity across the MLME_LIST_GET primitive. These lists are given in Table B.90. Although a management entity can read each of these lists, it cannot change the contents of any of them.

The Id field in table below would be the *PIBListAttribute* that needs to be passed MLME_LIST_GET primitive for accessing the value of these attributes.

Table B.90 – Table of read-only lists made available by MAC layer through management interface

List Attribute Name	Id	M	Description		
macListRegDevices	0x0050	BN	List of registered devices. This list is maintained by the Base Node only. Each entry in this list shall comprise the following information.		
			Entry Element	**Type**	**Description**
			regEntryID	EUI-48	EUI-48 of the registered Node.
			regEntryLNID	Integer16	LNID allocated to this Node.
			regEntryState	TERMINAL =1, SWITCH=2	Functional state of this Node.
			regEntryLSID	Integer16	SID allocated to this Node.
			regEntrySID	Integer16	SID of Switch through which this Node is connected.
			regEntryLevel	Interger8	Hierarchy level of this Node.

			regEntryTCap	Integer8	Bitmap of MAC Capabilities of Terminal functions in this device.
					Bits in sequence of right-to-left shall have the following meaning:
					Bit 0: Switch capable;
					Bit 1: Packet Aggregation;
					Bit 2: Contention Free Period;
					Bit 3: Direct connection;
					Bit 4: Multicast;
					Bit 5: PHY Robustness Management;
					Bit 6: ARQ;
					Bit 7: Reserved for future use.

Table B.90 – Table of read-only lists made available by MAC layer through management interface

List Attribute Name	Id	M	Description		

			regEntrySwCap	Integer8	Bitmap of MAC Switching capabilities of this device
					Bits in sequence of right-to-left shall have the following meaning:
					Bit 0: Direct Connection Switching Capability;
					Bit 1: Multicast switching;
					Bit 2: PHY Robustness Management Switching Capability;
					Bit 3: ARQ Buffering Switching Capability;
					Bit 4 to 7: Reserved for future use.

macListActiveConn	0x0051	BN	List of active non-direct connections. This list is maintained by the Base Node only.		
			Entry Element	**Type**	**Description**
			connEntrySID	Integer16	SID of Switch through which the Service Node is connected.
			connEntryLNID	Integer16	NID allocated to Service Node.
			connEntryLCID	Integer8	LCID allocated to this connection.
			connEntryID	EUI-48	EUI-48 of Service Node.
macListMcast Entries	0x0052	No	List of entries in multicast switching table. This list is not maintained by Service Nodes in a *Terminal* functional state.		
			Entry Element	**Type**	**Description**
			mcastEntryLCID	Integer8	LCID of the multicast group.
			mcastEntryMembers	Integer16	Number of child Nodes (including the Node itself) that are members of this group.
macListSwitchTable	0x0053	SN	List the Switch table. This list is not maintained by Service Nodes in a *Terminal* functional state.		
			Entry Element	**Type**	**Description**
			stblEntryLSID	Integer16	SID of attached Switch Node.

Table B.90 – Table of read-only lists made available by MAC layer through management interface

List Attribute Name	Id	M	Description		
macListDirectConn	0x0054	No	List of direct connections that are active. This list is maintained only in the Base Node.		
			Entry Element	**Type**	**Description**
			dconnEntrySrcSID	Integer16	SID of Switch through which the source Service Node is connected.
			dconEntrySrcLNID	Integer16	NID allocated to the source Service Node.
			dconnEntrySrcLCID	Integer8	LCID allocated to this connection at the source.
			dconnEntrySrcID	EUI-48	EUI-48 of source Service Node.
			dconnEntryDstSID	Integer16	SID of Switch through which the destination Service Node is connected.
			dconnEntryDstLNID	Integer16	NID allocated to the destination Service Node.
			dconnEntryDstLCID	Integer8	LCID allocated to this connection at the destination.
			dconnEntryDstID	EUI-48	EUI-48 of destination Service Node.
			dconnEntryDSID	Integer16	SID of Switch that is the direct Switch.
			dconnEntryDID	EUI-48	EUI-48 of direct switch.
macListDirectTable	0x0055	No	List the direct Switch table		
			Entry Element	**Type**	**Description**
			dconnEntrySrcSID	Integer16	SID of Switch through which the source Service Node is connected.
			dconEntrySrcLNID	Integer16	NID allocated to the source Service Node.
			dconnEntrySrcLCID	Integer8	LCID allocated to this connection at the source.
			dconnEntryDstSID	Integer16	SID of Switch through which the destination Service Node is connected.
			dconnEntryDstLNID	Integer16	NID allocated to the destination Service Node.
			dconnEntryDstLCID	Integer8	LCID allocated to this connection at the destination.
			dconnEntryDID	EUI-48	EUI-48 of direct switch.

Table B.90 – Table of read-only lists made available by MAC layer through management interface

List Attribute Name	Id	M	Description		
macListAvailable Switches	0x0056	SN	List of Switch Nodes whose beacons are received.		
			Entry Element	**Type**	**Description**
			slistEntrySNA	EUI-48	EUI-48 of the Subnetwork.
			slistEntryLSID	Integer16	SID of this Switch.
			slistEntryLevel	Integer8	Level of this Switch in Subnetwork hierarchy.
			slistEntryRxLvl	Integer8 EMA	Received signal level for this Switch.
			slistEntryRxSNR	Integer8 EMA	Signal to Noise Ratio for this Switch.
macListPhyComm	0x0057	All	List of PHY communication parameters. This table is maintained in every Node. For Terminal Nodes it contains only one entry for the Switch the Node is connected through. For other Nodes is contains also entries for every directly connected child Node.		
			Entry Element	**Type**	**Description**
			phyCommEUI	EUI-48	EUI-48 of the other device.
			phyCommTxPwr	Integer8	Tx power of GPDU packets send to the device.
			phyCommTxCod	Integer8	Tx coding of GPDU packets send to the device.
			phyCommRxCod	Integer8	Rx coding of GPDU packets received from the device.
			phyCommRxLvl	Integer8 EMA	Rx power level of GPDU packets received from the device.
			phyCommSNR	Integer8 EMA	SNR of GPDU packets received from the device.
			phyCommTxPwrMod	Integer8	Number of times the Tx power was modified.
			phyCommTxCodMod	Integer8	Number of times the Tx coding was modified.
			phyCommRxCodMod	Integer8	Number of times the Rx coding was modified.

B.6.2.3.6 Action PIB attributes

Some of the conformance tests require triggering certain actions on Service Nodes. The following table lists the set of action attributes that need to be supported by all implementations.

Table B.91 – Action PIB attributes

Attribute Name		Id	M	Size (in bits)	Description
MACActionTxData		0x0060	SN	8	Total number of PPDUs correctly decoded. Useful for PHY layer to estimate FER.
MACActionConnClose		0x0061	SN	8	Trigger to close one of the open connections.
MACActionRegReject		0x0062	SN	8	Trigger to reject incoming registration request.
MACActionProReject		0x0063	SN	8	Trigger to reject incoming promotion request.
MACActionUnregister		0x0064	SN	8	Trigger to unregister from the subnetwork.

B.6.2.4 Application PIB attributes

The following PIB attributes are used for general administration and maintenance of an OFDM TYPE 1 compliant device. These attributes do not affect the communication functionality, but enable easier administration.

These attributes shall be supported by both Base Node and Service Node devices.

Table B.92 – Applications PIB attributes

Attribute Name	Size (in bits)	Id	Description
AppFwVersion	128	0x0075	Textual description of firmware version running on device.
AppVendorId	16	0x0076	PRIME Alliance assigned unique vendor identifier.
AppProductId	16	0x0077	Vendor assigned unique identifier for specific product.

B.6.3 Firmware upgrade

B.6.3.1 General

The present clause specifies firmware upgrade. Devices supporting OFDM TYPE 1 may have several firmware inside them, at least one supporting the Application itself, and the one related to the OFDM TYPE 1 protocol. Although it is possible that the application can perform the firmware upgrade of all the firmware images of the device, for instance DLMS/COSEM image transfer, using COSEM image transfer object, supporting OFDM TYPE 1 firmware upgrade is mandatory in order to process to OFDM TYPE 1 firmware upgrade independently of the application.

B.6.3.2 Requirements and features

This clause specifies the firmware upgrade application, which is unique and mandatory for Base Nodes and Service Nodes.

The most important features of the Firmware Upgrade mechanism are listed below. See following chapters for more information. The FU mechanism:

- Shall be a part of management plane and therefore use the NULL SSCS, as specified in clause B.5.3.

- Is able to work in unicast (default mode) and multicast (optional mode). The control messages are always sent using unicast connections, whereas data can be transmitted using both unicast and multicast. No broadcast should be used to transmit data.
- May change the data packet sizes according to the channel conditions. The packet size will not be changed during the download process.
- Is able to request basic information to the Service Nodes at any time, such as device model, firmware version and FU protocol version.
- Shall be abortable at any time.
- Shall check the integrity of the downloaded FW after completing the reception. In case of failure, the firmware upgrade application shall request a new retransmission.
- The new firmware shall be executed in the Service Nodes only if they are commanded to do so. The FU application shall have to be able to set the moment when the reset takes place.
- Must be able to reject the new firmware after a "test" period and switch to the old version. The duration of this test period has to be fixed by the FU mechanism.

B.6.3.3 General description

B.6.3.3.1 General

The Firmware Upgrade mechanism is able to work in unicast and multicast modes. All control messages are sent using unicast connections, whereas the data can be sent via unicast (by default) or multicast (only if supported by the manufacturer). Note that in order to ensure correct reception of the FW when Service Nodes from different vendors are upgraded, data packets shall not be sent via broadcast. Only unicast and multicast are allowed. A Node will reply only to messages sent via unicast. See chapter B.6.3.5 for a detailed description of the control and information messages used by the FU mechanism.

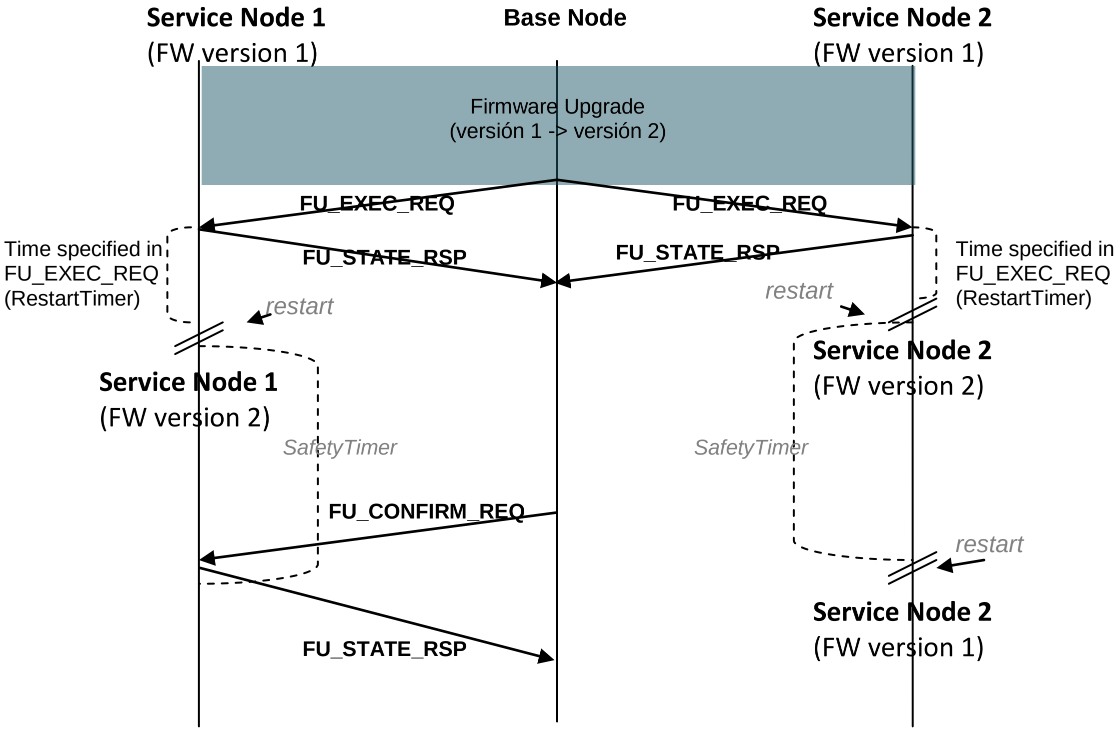
The unicast and multicast connections are set up by the Base Node. In case of supporting multicast, the Base Node shall request the Nodes from a specific vendor to join a specific multicast group, which is exclusively created to perform the firmware upgrade and is removed after finishing it.

As said before, it is up to the vendor to use unicast or multicast for transmitting the data. In case of unicast data transmission, please note that the use of ARQ is an optional feature. Some examples showing the traffic between the Base Node and the Service Nodes in unicast and multicast are provided in B.6.3.6.

After completing the firmware download, each Service Node is committed by the Base Node to perform an integrity check on it. The firmware download will be restarted if the firmware image results to be corrupt. In other case, the Service Nodes will wait until they are commanded by the Base Node to execute the new firmware.

The FU mechanism can setup the instant when the recently downloaded firmware is executed on the Service Nodes. Thus, the Base Node can choose to restart all Nodes at the same time or in several steps. After restart, each Service Node runs the new firmware for a time period specified by the FU mechanism. If this period expires without receiving any confirmation from the Base Node, or the Base Node decides to abort the upgrade process, the Service Nodes will reject the new firmware and switch to the old version. In any other case (a confirmation message is received) the Service Nodes will consider the new firmware as the only valid version and delete the old one.

This is done in order to leave an "open back-door" in case that the new firmware is defect or corrupt. Please note that the Service Nodes are not allowed to discard any of the stored firmware versions until the final confirmation from the Base Node arrives or until the safety time period expires. The two last firmware upgrade steps explained above are shown in B.6.3.5. See chapter B.6.3.5.3 for a detailed description of the control messages.



NOTE – In normal circumstances, both Service Nodes should either accept or reject the new firmware version. Both possibilities are shown above simultaneously for academic purposes.

Figure B.85 – Restarting the nodes and running the new firmware

B.6.3.3.2 Segmentation

The firmware image is the information to be transferred, in order to process a firmware upgrade. The size of the firmware image will be called "*ImageSize*", and is measured in bytes. This image is divided in smaller elements called pages that are easier to be transferred in packets. The "*PageSize*" may be one of the following: 32 bytes, 64 bytes, 128 bytes or 192 bytes. This implies that the number of pages in a firmware image is calculated by the following formula:

$$PageCount = \left\lceil \frac{ImageSize}{PageSize} \right\rceil + 1$$

Every page will have a size specified by *PageSize,* except the last one that will contain the remaining bytes up to *ImageSize.*

The *PageSize* is configured by the Base Node and notified during the initialization of the Firmware Upgrade process, and imposes a condition in the size of the packets being transferred by the protocol.

B.6.3.4 Firmware upgrade PIB attributes

The following PIB attributes shall be supported by Service Nodes to support the firmware download application.

Table B.93 – FU PIB attributes

Attribute Name	Size (in bits)	Id	Description
AppFwdlRunning	16	0x0070	Indicate if a firmware download is in progress or not. 0 = No firmware download; 1 = Firmware download in progress.
AppFwdlRxPkt Count	16	0x0071	Count of firmware download packets that have been received until the time of query.

B.6.3.5 State machine

B.6.3.5.1 General

A Service Node using the Firmware Upgrade service will be in one of five possible states: *Idle, Receiving, Complete, Countdown* and *Upgrade*. These states, the events triggering them and the resulting actions/output messages are detailed below.

Table B.94 – FU state machine

FU State	Description	Event	Output (or action to be performed)	Next state
Idle	The FU application is doing nothing.	Receive FU_INFO_REQ	FU_INFO_RSP.	*Idle*
		Receive FU_STATE_REQ	FU_STATE_RSP (.State = 0).	*Idle*
		Receive FU_MISS_REQ	FU_STATE_RSP (.State = 0).	*Idle*
		Receive FU_CRC_REQ	FU_STATE_RSP (.State = 0).	*Idle*
		Receive FU_INIT_REQ	FU_STATE_RSP (.State = 1).	*Receiving*
		Receive FU_DATA	(ignore).	*Idle*
		Receive FU_EXEC_REQ	FU_STATE_RSP (.State = 0).	*Idle*
		Receive FU_CONFIRM_REQ	FU_STATE_RSP (.State = 0).	*Idle*
		Receive FU_KILL_REQ	FU_STATE_RSP (.State = 0).	*Idle*
Receiving	The FU application is receiving the Firmware Image.	Complete FW received and CRC OK	(if CRC of the complete Image is OK, switch to *Complete* without sending any additional messages)	*Complete*
		Receive FU_INFO_REQ	FU_INFO_RSP.	*Receiving*
		Receive FU_STATE_REQ	FU_STATE_RSP (.State = 1).	*Receiving*
		Receive FU_MISS_REQ	FU_MISS_LIST or FU_MISS_BITMAP.	*Receiving*

Table B.94 – FU state machine

FU State	Description	Event	Output (or action to be performed)	Next state
		Receive FU_CRC_REQ	FU_CRC_RSP (FU_STATE_RSP if the Bitmap is not complete)	*Receiving*
		Receive FU_INIT_REQ	FU_STATE_RSP (.State = 1)	*Receiving*
		Receive FU_DATA	(receiving data, normal behaviour).	*Receiving*
		Receive FU_EXEC_REQ	FU_STATE_RSP (.State = 1).	*Receiving*
		Receive FU_CONFIRM_REQ	FU_STATE_RSP (.State = 1).	*Receiving*
		Receive FU_KILL_REQ	FU_STATE_RSP (.State = 0); (switch to *Idle).*	*Idle*
Complete	Upgrade completed, image integrity OK, the Service Node is waiting to reboot with the new FW version.	Receive FU_INFO_REQ	FU_INFO_RSP.	*Complete*
		Receive FU_STATE_REQ	FU_STATE_RSP (.State = 2).	*Complete*
		Receive FU_MISS_REQ	FU_STATE_RSP (.State = 2).	*Complete*
		Receive FU_CRC_REQ	FU_STATE_RSP (.State = 2).	*Complete*
		Receive FU_INIT_REQ	FU_STATE_RSP (.State = 2).	*Complete*
		Receive FU_DATA	(ignore).	*Complete*
		Receive FU_EXEC_REQ with *RestartTimer* != 0	FU_STATE_RSP (.State = 3).	*Countdown*
		Receive FU_EXEC_REQ with *RestartTimer* = 0	FU_STATE_RSP (.State = 4).	*Upgrade*
		Receive FU_CONFIRM_REQ	FU_STATE_RSP (.State = 2).	*Complete*
		Receive FU_KILL_REQ	FU_STATE_RSP (.State = 0); (switch to *Idle*).	*Idle*

Table B.94 – FU state machine

FU State	Description	Event	Output (or action to be performed)	Next state
Countdown	Waiting until *RestartTimer* expires.	*RestartTimer* expires	(switch to *Upgrade).*	*Upgrade*
		Receive FU_INFO_REQ	FU_INFO_RSP.	*Countdown*
		Receive FU_STATE_REQ	FU_STATE_RSP (.State = 3).	*Countdown*
		Receive FU_MISS_REQ	FU_STATE_RSP (.State = 3).	*Countdown*
		Receive FU_CRC_REQ	FU_STATE_RSP (.State = 3).	*Countdown*
		Receive FU_INIT_REQ	FU_STATE_RSP (.State = 3).	*Countdown*
		Receive FU_DATA	(ignore).	*Countdown*
		Receive FU_EXEC_REQ with *RestartTimer* != 0	FU_STATE_RSP (.State = 3); (update *RestartTimer* and *SafetyTimer*).	*Countdown*
		Receive FU_EXEC_REQ with *RestartTimer* = 0	FU_STATE_RSP (.State = 4); (update *RestartTimer* and *SafetyTimer*).	*Upgrade*
		Receive FU_CONFIRM_REQ	FU_STATE_RSP (.State = 3).	*Countdown*
		Receive FU_KILL_REQ	FU_STATE_RSP (.State = 0); (switch to *Idle*).	*Idle*
Upgrade	The FU mechanism reboots using the new FW image and tests it for *SafetyTimer* seconds.	*SafetyTimer* expires	FU_STATE_RSP (.State = 0); (switch to *Idle*, FW rejected).	*Idle*
		Receive FU_INFO_REQ	FU_INFO_RSP.	*Upgrade*
		Receive FU_STATE_REQ	FU_STATE_RSP (.State = 4).	*Upgrade*
		Receive FU_MISS_REQ	FU_STATE_RSP (.State = 4).	*Upgrade*
		Receive FU_CRC_REQ	FU_STATE_RSP (.State = 4).	*Upgrade*
		Receive FU_INIT_REQ	FU_STATE_RSP (.State = 4).	*Upgrade*
		Receive FU_DATA	(ignore).	*Upgrade*
		Receive FU_EXEC_REQ	FU_STATE_RSP (.State = 0).	*Upgrade*
		Receive FU_CONFIRM_REQ	FU_STATE_RSP (.State = 0); (switch to *Idle*, FW accepted).	*Idle*
		Receive FU_KILL_REQ	FU_STATE_RSP (.State = 0); (switch to *Idle*, FW rejected).	*Idle*

The state diagram is represented below. Please note that only the most relevant events are shown in the state transitions. See B.6.3.5.3 for a detailed description of each state's behaviour and the events and actions related to them. A short description of each state is provided in B.6.3.5.2.

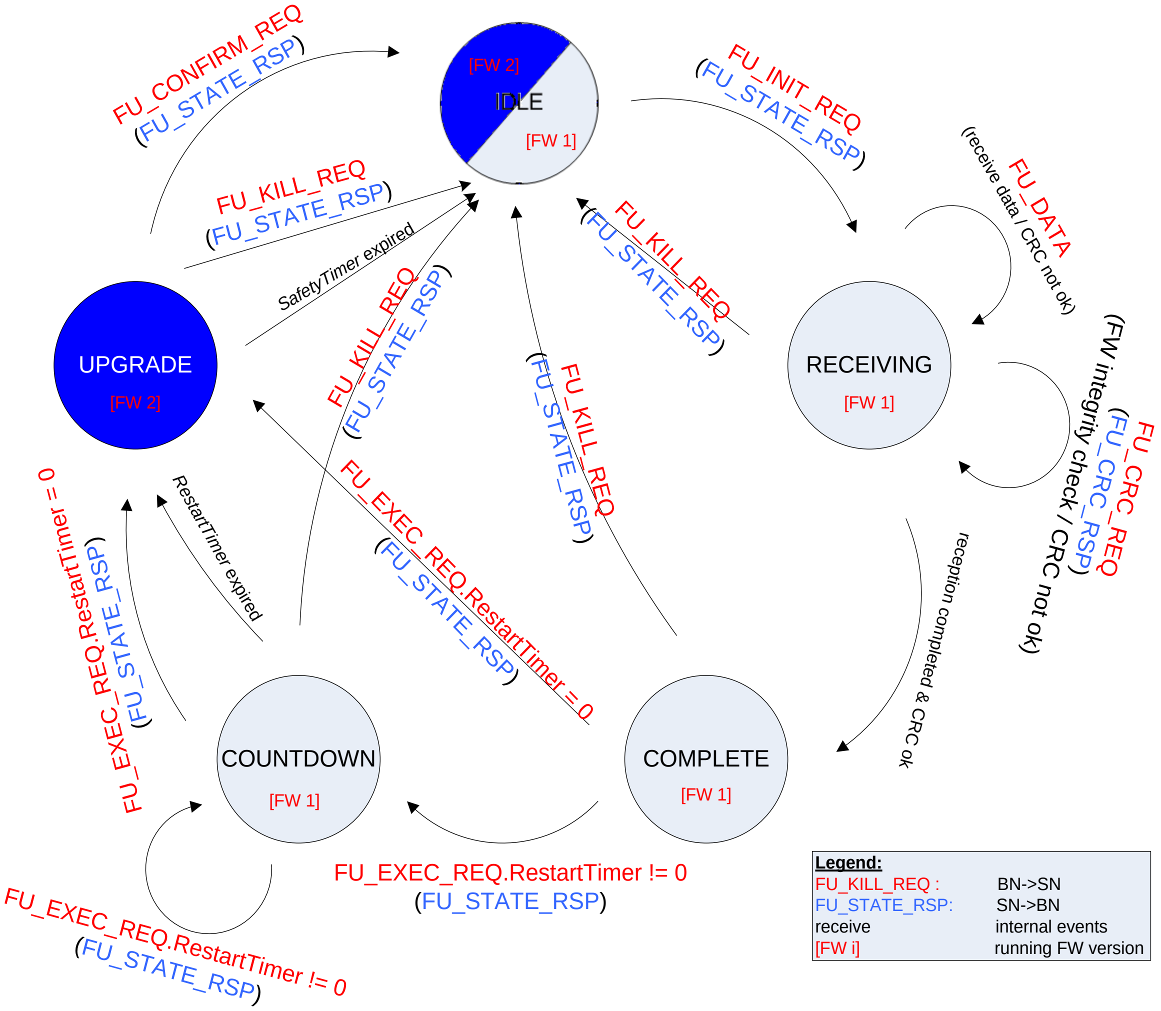


Figure B.86 – Firmware upgrade mechanism, state diagram

B.6.3.5.2 State description

B.6.3.5.2.1 Idle

The Service Nodes are in "Idle" state when they are not performing a firmware upgrade. The reception of a FU_INIT_REQ message is the only event that forces the Service Node to switch to the next state ("*Receiving*"). FU_KILL_REQ aborts the upgrade process and forces the Service Nodes to switch from any state to "*Idle*".

B.6.3.5.2.2 Receiving

The Service Nodes receive the firmware image via FU_DATA messages. Once the download is complete, the integrity of the image is checked by the Base Node using FU_CRC_REQ and the Service Node responds with FU_CRC_RSP. This final CRC on the complete FW image is mandatory. If the CRC results to be OK, the Service Node responds with FU_CRC_RSP and then switches to "*Complete*" state. If the CRC is wrong, the Service Node reports to the Base Node via FU_CRC_RSP, drops the complete FW image, updates the bitmap accordingly and waits for packet retransmission.

Please remember that the Service Node will change from "*Receiving*" to "*Complete*" state only if the complete FW has been downloaded and the CRC has been successful.

B.6.3.5.2.3 Complete

A Service Node in "*Complete*" state waits until reception of a FU_EXEC_REQ message. The Service Node may switch either to "*Countdown*" or "*Upgrade*" depending on the field *RestartTimer*, which specifies in which instant the Service Node has to reboot using the new firmware. If *RestartTimer* = 0, the Service Node immediately switches to "*Upgrade*"; else, the Service Node switches to "Countdown".

B.6.3.5.2.4 Countdown

A Service Node in "*Countdown*" state waits a period of time specified in the *RestartTimer* field of a previous FU_EXEC_REQ message. When this timer expires, it automatically switches to "*Upgrade*".

FU_EXEC_REQ can be used in "*Countdown*" state to reset *RestartTimer* and *SafetyTimer*. In this case, both timers have to be specified in FU_EXEC_REQ because both will be overwritten. Note that it is possible to force the Node to immediately switch from "*Countdown*" to "*Upgrade*" state setting *RestartTimer* to zero.

B.6.3.5.2.5 Upgrade

A Service Node in "*Upgrade*" state shall run the new firmware during a time period specified in FU_EXEC_REQ.SafetyTimer. If it does not receive any confirmation at all before this timer expires (or if it receives a FU_KILL_REQ message), the Service Node discards the new FW, reboots with the old version and switches to "*Idle*" state. In any other case it discards the old FW version and switches to "*Idle*" state.

B.6.3.5.3 Control packets

B.6.3.5.3.1 FU_INIT_REQ

The Base Node sends this packet in order to configure a Service Node for the Firmware Upgrade. If the Service Node is in "*Idle*" state, it will change its state from "*Idle*" to "*Receiving*" and will answer with FU_STATE_RSP. In any other case it will just answer sending FU_STATE_RSP.

The content of FU_INIT_REQ is shown below.

Table B.95 – Fields of FU_INIT_REQ

Field	Length	Description
Type	4 bits	0 = FU_INIT_REQ.
Version	2 bits	0 for this version of the protocol.

PageSize	2 bits	0 for a PageSize=32;
		1 for a PageSize=64;
		2 for a PageSize=128;
		3 for a PageSize=192.

ImageSize	32 bits	Size of the Firmware Upgrade image in bytes.
CRC	32 bits	CRC of the Firmware Upgrade Image. The input polynomial M(x) is formed as a polynomial whose coefficients are bits of the data being checked (the first bit to check is the highest order coefficient and the last bit to check is the coefficient of order zero). The Generator polynomial for the CRC is $G(x)=x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^{8}+x^{7}+x^{5}+x^{4}+x^{2}+x+1$. The remainder R(x) is calculated as the remainder from the division of $M(x)\cdot x^{32}$ by G(x). The coefficients of the remainder will then be the resulting CRC.

B.6.3.5.3.2 FU_EXEC_REQ

This packet is used by the Base Node to command a Service Node in "*Complete*" state to restart using the new firmware, once the complete image has been received by the Service Node. FU_EXEC_REQ specifies when the Service Node has to restart and how long the "safety" period shall be, as explained in B.6.3.5.2.5. Additionally, FU_EXEC_REQ can be used in "*Countdown*" state to reset the restart and the safety timers.

Depending on the value of *RestartTimer*, a Service Node in "*Complete*" state may change either to "*Countdown*" or to "*Upgrade*" state. In any case, the Service Node answers with FU_STATE_RSP.

In "*Countdown*" state, the Base Node can reset *RestartTimer* and *SafetyTimer* with a FU_EXEC_REQ message (both timers must be specified in the message because both will be overwritten).

The content of this packet is described below.

Table B.96 – Fields of FU_EXEC_REQ

Field	Length	Description
Type	4 bits	1 = FU_EXEC_REQ.
Version	2 bits	0 for this version of the protocol.
Reserved	2 bits	0.
RestartTimer	16 bits	0..65536 seconds; time before restarting with new FW.
SafetyTimer	16 bits	0..65536 seconds; time to test the new FW. It starts when the "*Upgrade*" state is entered.

B.6.3.5.3.3 FU_CONFIRM_REQ

This packet is sent by the Base Node to a Service Node in "*Upgrade*" state to confirm the current FW. If the Service Node receives this message, it discards the old FW version and switches to "*Idle*" state. The Service Node answers with FU_STATE_RSP when receiving this message.

In any other state, the Service Node answers with FU_STATE_RSP without performing any additional actions.

This packet contains the fields described below.

Table B.97 – Fields of FU_CONFIRM_REQ

Field	Length	Description
Type	4 bits	2 = FU_CONFIRM_REQ.
Version	2 bits	0 for this version of the protocol.
Reserved	2 bits	0.

B.6.3.5.3.4 FU_STATE_REQ

This packet is sent by the Base Node in order to get the Firmware Upgrade state of a Service Node. The Service Node will answer with FU_STATE_RSP.

This packet contains the fields described below.

Table B.98 – Fields of FU_STATE_REQ

Field	Length	Description
Type	4 bits	3 = FU_STATE_REQ.
Version	2 bits	0 for this version of the protocol.
Reserved	2 bits	0.

B.6.3.5.3.5 FU_KILL_REQ

The Base Node sends this message to terminate the Firmware Upgrade process. A Service Node receiving this message will automatically switch to "*Idle*" state and optionally delete the downloaded data. The Service Node replies sending FU_STATE_RSP.

The content of this packet is described below.

Table B.99 – Fields of FU_KILL_REQ

Field	Length	Description
Type	4 bits	4 = FU_KILL_REQ.
Version	2 bits	0 for this version of the protocol.
Reserved	2 bits	0.

B.6.3.5.3.6 FU_STATE_RSP

This packet is sent by the Service Node as an answer to FU_STATE_REQ, FU_KILL_REQ, FU_EXEC_REQ, FU_CONFIRM_REQ or FU_INIT_REQ messages received through the unicast connection. It is used to notify the Firmware Upgrade state in a Service Node.

Additionally, FU_STATE_RSP is used as default response to all events that happen in states where they are not foreseen (e.g., FU_EXEC_REQ in "*Receiving*" state, FU_INIT_REQ in "*Upgrade*"...).

This packet contains the fields described below.

Table B.100 – Fields of FU_STATE_RSP

Field	Length	Description
Type	4 bits	5 = FU_STATE_RSP.
Version	2 bits	0 for this version of the protocol.
Reserved	2 bits	0.

State	4 bits	0 for Idle;
		1 for Receiving;
		2 for Complete;
		3 for Countdown;
		4 for Upgrade;
		5 to 15 reserved for future use.

Reserved	4 bits	0.
CRC	32 bits	CRC as the one received in the CRC field of FU_INIT_REQ.
Received	32 bits	Number of received pages (this field should only be present if State is Receiving).

B.6.3.5.3.7 FU_DATA

This packet is sent by the Base Node to transfer a page of the Firmware Image to a Service Node. No answer is expected by the Base Node.

This packet contains the fields described below.

Table B.101 – Fields of FU_DATA

Field	Length	Description
Type	4 bits	6 = FU_DATA.
Version	2 bits	0 for this version of the protocol.
Reserved	2 bits	0.
PageIndex	32 bits	Index of the page being transmitted.
Reserved	8 bits	Padding byte for 16-bit devices. Set to 0 by default.
Data	*Variable*	Data of the page. The length of this data is PageSize (32, 64, 128 or 192) bytes for every page, except the last one that will have the remaining bytes of the image.

B.6.3.5.3.8 FU_MISS_REQ

This packet is sent by the Base Node to a Service Node to request information about the pages that are still to be received.

If the Service Node is in "*Receiving*" state it will answer with a FU_MISS_BITMAP or FU_MISS_LIST message. If the Service Node is in any other state it will answer with a FU_STATE_RSP.

This packet contains the fields described below.

Table B.102 – Fields of FU_MISS_REQ

Field	Length	Description
Type	4 bits	7 = FU_MISS_REQ.
Version	2 bits	0 for this version of the protocol.
Reserved	2 bits	0.
PageIndex	32 bits	Starting point to gather information about missing pages.

B.6.3.5.3.9 FU_MISS_BITMAP

This packet is sent by the Service Node as an answer to a FU_MISS_REQ. It carries the information about the pages that are still to be received.

This packet will contain the fields described below.

Table B.103 – Fields of FU_MISS_BITMAP

Field	Length	Description
Type	4 bits	8 = FU_MISS_BITMAP.
Version	2 bits	0 for this version of the protocol.
Reserved	2 bits	0.
PageIndex	32 bits	Page index of the page represented by the first bit of the bitmap. It should be the same as the *PageIndex* field in FU_MISS_REQ messages, or a posterior one. If it is posterior, it means that the pages in between are already received. In this case, if all pages after the *PageIndex* specified in FU_MISS_REQ have been received, the Service Node shall start looking from the beginning (*PageIndex* = 0).
Bitmap	*Variable*	This bitmap contains the information about the status of each page. The first bit (most significant bit of the first byte) represents the status of the page specified by *PageIndex*. The next bit represents the status of the *PageIndex+1* and so on. A '1' represents that a page is missing, a '0' represents that the page is already received. After the bit that represents the last page in the image, it is allowed to overflow including bits that represent the missing status of the page with index zero. The maximum length of this field is *PageSize* bytes.

It is up to the Service Node to decide to send this type of packet or a FU_MISS_LIST message. It is usually more efficient to transmit this kind of packets when the number of missing packets is not very low. But it is up to the implementation to transmit one type of packet or the other. The Base Node should understand both.

B.6.3.5.3.10 FU_MISS_LIST

This packet is sent by the Service Node as an answer to a FU_MISS_REQ. It carries the information about the pages that are still to be received.

This packet will contain the fields described below.

Table B.104 – Fields of FU_MISS_LIST

Field	Length	Description
Type	4 bits	9 = FU_MISS_LIST.
Version	2 bits	0 for this version of the protocol.
Reserved	2 bits	0.
PageIndexList	Variable	List of pages that are still to be received. Each page is represented by its PageIndex, coded as a 32 bit integer. These pages should be sorted in ascending order (low to high), being possible to overflow to the *PageIndex* equal to zero to continue from the beginning. The first page index should be the same as the *PageIndex* field in FU_MISS_REQ, or a posterior one. If it is posterior, it means that the pages in between are already received (by posterior it is allowed to overflow to the page index zero, to continue from the beginning). The maximum length of this field is *PageSize* bytes.

It is up to the Service Node to decide to transmit this packet type or a FU_MISS_BITMAP message. It is usually more efficient to transmit this kind of packets when the missing packets are very sparse, but it is implementation-dependent to transmit one type of packet or the other. The Base Node should understand both.

B.6.3.5.3.11 FU_INFO_REQ

This packet is sent by a Base Node to request information from a Service Node, such as manufacturer, device model, firmware version and other parameters specified by the manufacturer. The Service Node will answer with one or more FU_INFO_RSP packets.

This packet contains the fields described below.

Table B.105 – Fields of FU_INFO_REQ

Field	Length	Description
Type	4 bits	10 = FU_INFO_REQ.
Version	2 bits	0 for this version of the protocol.
Reserved	2 bits	0.
InfoIdList	Variable	List of identifiers with the information to retrieve. Each identifier is 1 byte long. The maximum length of this field is 32 bytes.

The following identifiers are defined:

Table B.106 – InfoId possible values

InfoId	Name	Description
0	Manufacturer	Universal Identifier of the Manufacturer.
1	Model	Model of the product working as Service Node.
2	Firmware	Current firmware version being executed.
128-255	*Manufacturer specific*	Range of values that are manufacturer specific.

B.6.3.5.3.12 FU_INFO_RSP

This packet is sent by a Service Node as a response to a FU_INFO_REQ message from the Base Node. A Service Node may have to send more than one FU_INFO_RSP when replying to an information request by the Base Node.

This packet contains the fields described below.

Table B.107 – Fields of FU_INFO_RSP

Field	Length	Description
Type	4 bits	11 = FU_INFO_RSP.
Version	2 bits	0 for this version of the protocol.
Reserved	2 bits	0.
InfoData	0 – 192 bytes	Data with the information requested by the Base Node. It may contain several entries (one for each requested identifier), each entry has a maximum size of 32 bytes. The maximum size of this field is 192 bytes (6 entries).

The InfoData field can contain several entries, the format of each entry is specified below.

Table B.108 – Fields of each entry of InfoData in FU_INFO_RSP

Field	Length	Description
InfoId	8 bits	Identifier of the information as specified in B.6.3.5.3.11.
Reserved	3 bits	0.
Length	5 bits	Length of the Data field (If Length is 0 it means that the specified InfoId is not supported by the specified device).
Data	0 – 30 bytes	Data with the information provided by the Service Node. Its content may depend on the meaning of the InfoId field. No value may be longer than 30 bytes.

B.6.3.5.3.13 FU_CRC_REQ

FU_CRC_REQ is sent by the Base Node to command a Service Node to perform a CRC on the complete firmware image. The CRC on the complete FW image is mandatory. The CRC specified in FU_CRC_REQ.CRC is the same as in FU_INIT_REQ.

The Service Node replies with FU_CRC_RSP if it is in "*Receiving*" state, in any other case it replies with FU_STATE_RSP. The Base Node shall not send a FU_CRC_REQ if the image download is not complete (that is, the bitmap is not complete). Should the Base Node have an abnormal behaviour and send FU_CRC_REQ before completing the FW download, the Service Node would reply with FU_STATE_RSP.

Please note that in "*Idle*" state, the CRC field from FU_STATE_RSP will be a dummy (because no FU_INIT_REQ has been received yet). The Base Node will ignore this field if the Service Node is in "*Idle*" state.

This packet contains the fields described below.

Table B.109 – Fields of FU_CRC_REQ

Field	Length	Description
Type	4 bits	12 = FU_CRC_REQ.
Version	2 bits	0 for this version of the protocol.
Reserved	2 bits	0.
SectionSize	32 bits	Size of the Firmware Upgrade Image in bytes.
CRC	32 bits	CRC of the Firmware Upgrade Image. The input polynomial M(x) is formed as a polynomial whose coefficients are bits of the data being checked (the first bit to check is the highest order coefficient and the last bit to check is the coefficient of order zero). The Generator polynomial for the CRC is $G(x)=x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^{8}+x^{7}+x^{5}+x^{4}+x^{2}+x+1$. The remainder R(x) is calculated as the remainder from the division of $M(x)\cdot x^{32}$ by G(x). The coefficients of the remainder will then be the resulting CRC.

B.6.3.5.3.14 FU_CRC_RSP

This packet is sent by the Service Node as a response to a FU_CRC_REQ message sent by the Base Node.

This packet contains the fields described below.

Table B.110 – Fields of FU_CRC_RSP

Field	Length	Description
Type	4 bits	13 = FU_CRC_RSP.
Version	2 bits	0 for this version of the protocol.
CRC_Result	1 bit	Result of the CRC: "0" check failed; "1" check OK.
Reserved	1 bit	0.

B.6.3.6 Examples

The figures below are an example of the traffic generated between the Base Node and the Service Node during the Firmware Upgrade process.

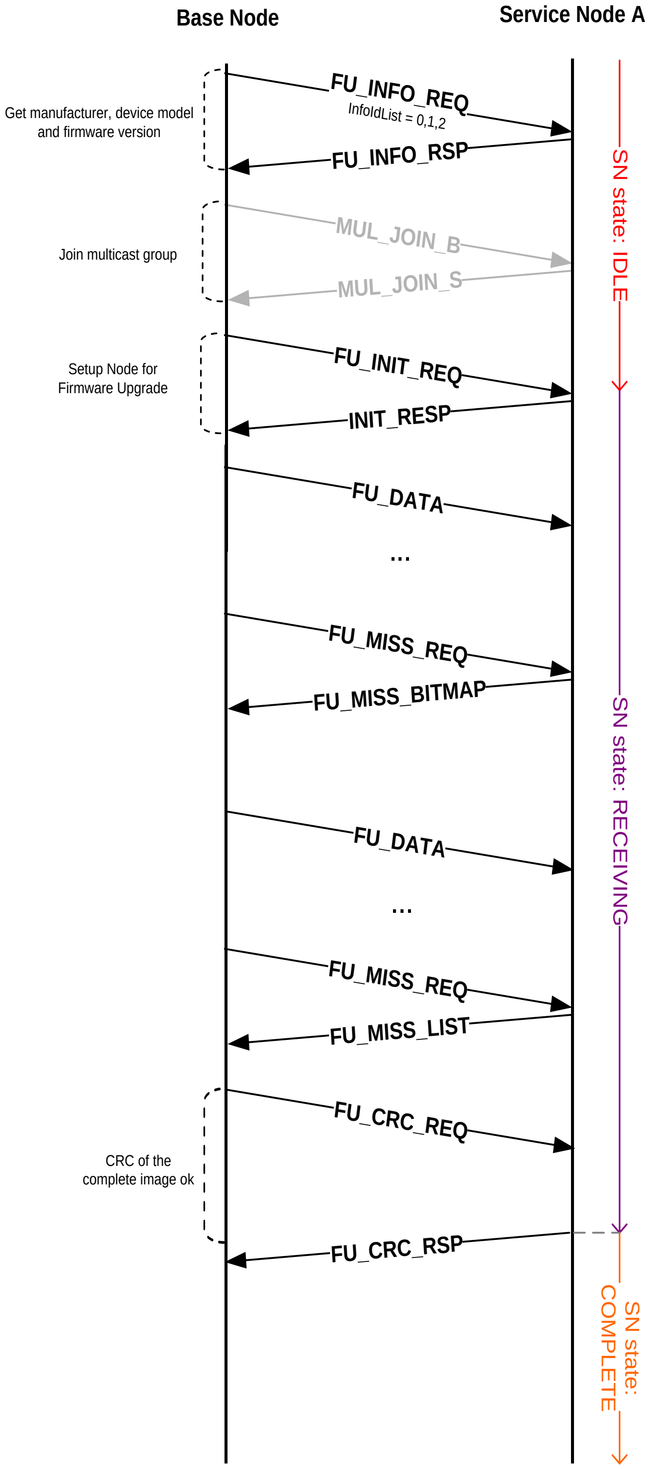


Figure B.87 – Init service node and complete FW image

Above figure shows the initialization of the process, the FW download and the integrity check of the image. In the example above, the downloaded FW image is supposed to be complete before sending the last FU_MISS_REQ. The Base Node sends it to verify its bitmap. In this example, FU_MISS_LIST has an empty *PageIndexList* field, which means that the FW image is complete.

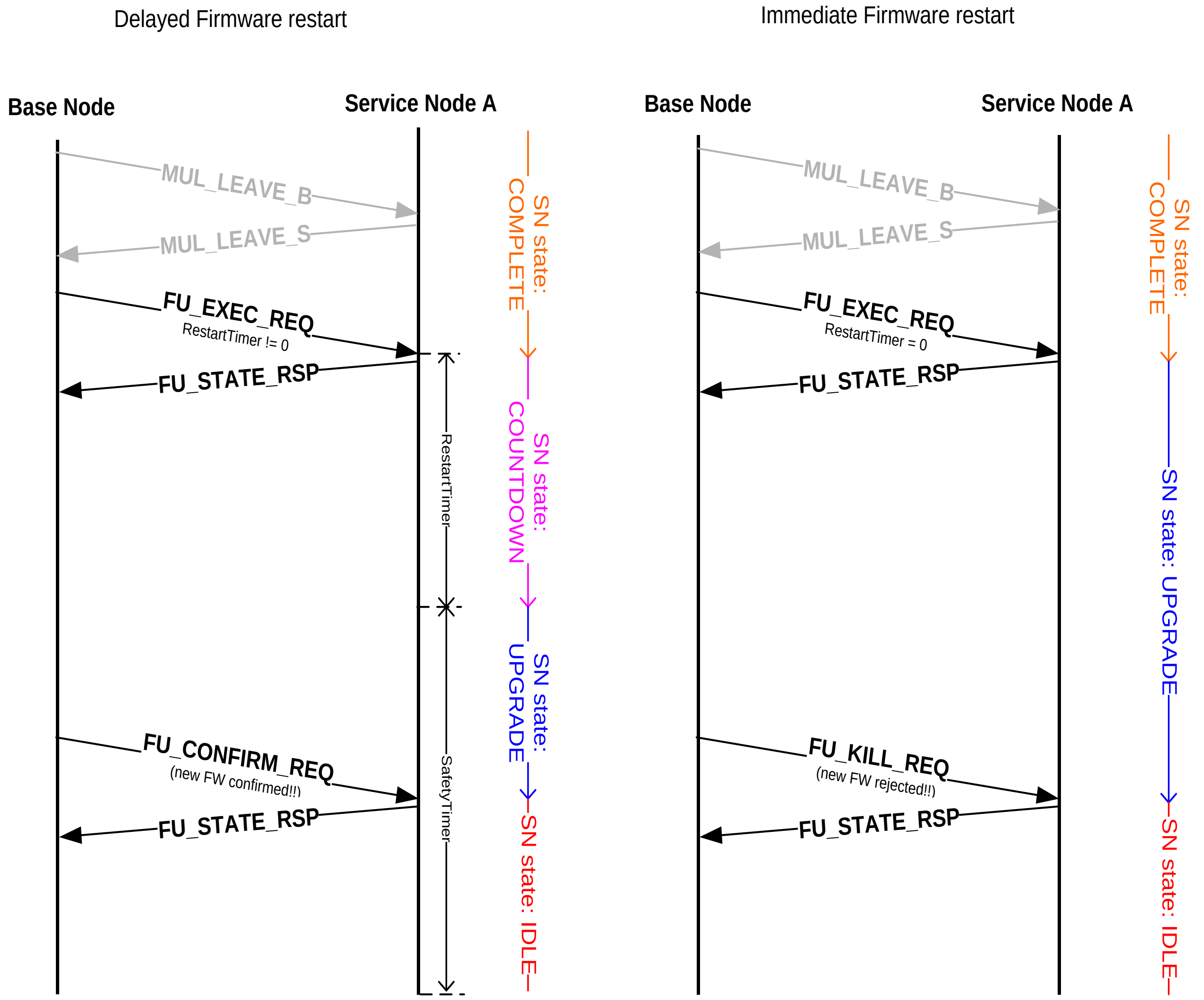


Figure B.88 – Execute upgrade and confirm/reject new FW version

Above it is shown how to proceed after completing the FW download. The Base Node commands the Service Node to reboot either immediately ("Immediate Firmware Start", *RestartTimer* = 0) or after a defined period of time ("Delayed Firmware start", *RestartTimer* != 0). After reboot, the Base Node can either confirm the recently downloaded message sending a FU_CONFIRM_REQ or reject it (sending a FU_KILL_REQ or letting the safety period expire doing nothing).

B.6.4 Management interface description

B.6.4.1 General

Management functions defined in earlier clauses shall be available over an abstract management interface specified in this clause. The management interface can be accessed over diverse media. Each physical media shall specify its own management plane communication profile over which management information is exchanged. It is mandatory for implementations to support PRIME management plane communication profile. All other "management plane communication profiles" are optional and maybe mandated by certain "application profiles" to use in specific cases.

The present version of specifications describes two communication profiles, one of which is over this specification NULL SSCS and other over serial link.

With these two communication profiles, it shall be possible to address the following use-cases:

1. Remote access of management interface over NULL SSCS. This shall enable Base Node's use as a supervisory gateway for all devices in a subnetwork.

2. Local access of management interface (over peripherals like RS232, USBSerial, etc.) in a Service Node. Local access shall fulfil cases where a coprocessor exists for supervisory control of processor or when manual access is required over local physical interface for maintenance.

Management data comprises of a 2 bytes header followed by payload information corresponding to the type of information carried in message. The header comprises of a 10 bit length field and 6 bit message_id field.

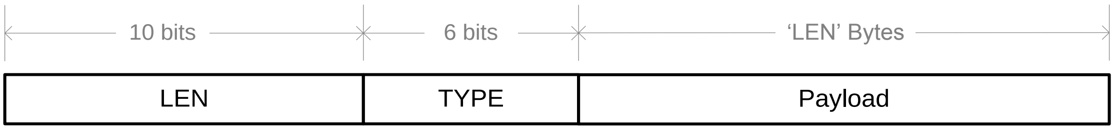


Table B.111 – Management data frame fields

Name	Length	Description
MGMT.LEN	10 bits	Length of payload data following the 2 byte header. LEN=0 implies there is no payload data following this header and the TYPE field contains all required information to perform appropriate action. NOTE – The length field maybe redundant in some communication profiles (e.g., When transmitted over OFDM TYPE 1), but is required in others. Therefore for the sake of uniformity, it is always included in management data.

MGMT.TYPE	6 bits	Type of management information carried in corresponding data. Some message_id s have standard semantics which should be respected by all OFDM TYPE 1 compliant devices while others are reserved for local use by vendors.
		0x00 – Get PIB attribute query;
		0x01 – Get PIB attribute response;
		0x02 – Set PIB attribute command;
		0x03 – Reset all PIB statistics attributes;
		0x04 – Reboot destination device;
		0x05 – Firmware upgrade protocol message;
		0x06 – 0x0F: Reserved for future use. Vendors should not use these values for local purpose;
		0x10 – 0x3F: Reserved for vendor specific use.

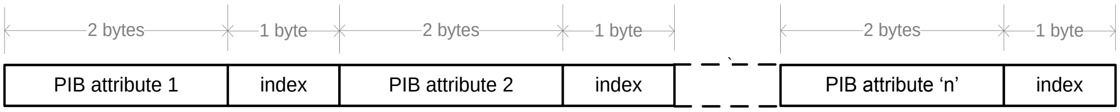
B.6.4.2 Payload format of management information

B.6.4.2.1 Get PIB attribute query

This query is issued by a remote management entity that is interested in knowing values of PIB attributes maintained on a compliant device with this specification.

The payload may comprise of a query on either a single PIB attribute or multiple attributes. For reasons of efficiency queries on multiple PIB attributes maybe aggregated in one single command. Given that the length of a PIB attribute identifier is constant, the number of attributes requested in a single command is derived from the overall MGMT.LEN field in header.

The format of payload information is shown in the following figure.



Fields of a GET request are summarized in table below:

Table B.112 – GET PIB attribute request fields

Name	Length	Description
PIB Attribute id	2 bytes	16 bit PIB attribute identifier
Index	1 byte	Index of entry to be returned for corresponding PIB Attribute id. This field is only of relevance while returning PIB list attributes. Index = 0; if PIB Attribute is not a list; Index = 1 to 255; Return list record at given index.

B.6.4.2.2 Get PIB attribute response

This data is sent out from a compliant device of this specification in response to a query of one or more PIB attributes. If a certain queried PIB attribute is not maintained on the device, it shall still respond to the query with value field containing all '1s' in the response.

The format of payload is shown in the following figure.

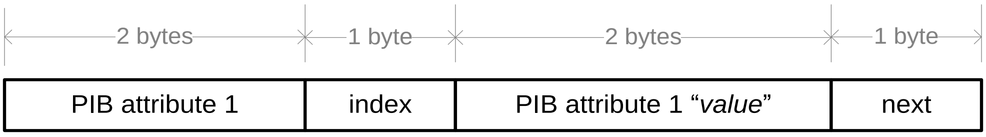


Figure B.89 – Get PIB attribute response.Payload

Fields of a GET request are summarized in table below:

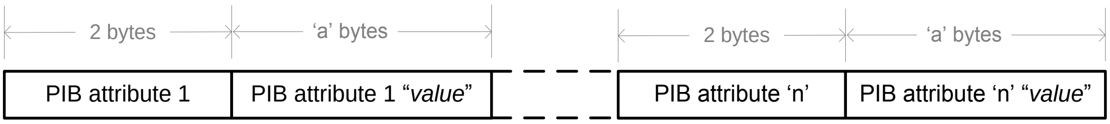
Table B.113 – GET PIB attribute response fields

Name	Length	Description
PIB Attribute id	2 bytes	16 bit PIB attribute identifier.
Index	1 byte	Index of entry returned for corresponding PIB Attribute id. This field is only of relevance while returning PIB list attributes. index = 0; if PIB Attribute is not a list. index = 1 to 255; Returned list record is at given index.
PIB Attribute value	'a' bytes	Values of requested PIB attribute. In case of a list attribute, value shall comprise of entire record corresponding to given index of PIB attribute
Next	1 byte	Index of next entry returned for corresponding PIB Attribute id. This field is only of relevance while returning PIB list attributes. next = 0; if PIB Attribute is not a list or if no records follow the one being returned for a list PIB attribute, i.e., given record is last entry in list. next = 1 to 255; index of next record in list maintained for given PIB attribute.

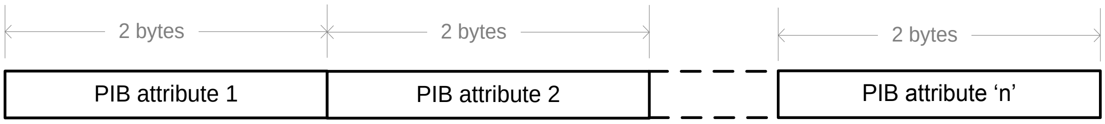
Response to PIB attribute query can span across several MAC GPDUs. This shall always be the case when an aggregated (comprising of several PIB attributes) PIB query's response if longer than the maximum segment size allowed to be carried over the NULL SCSS.

B.6.4.2.3 Set PIB attribute

This management data shall be used to set specific PIB attributes. Such management payload comprises of a 2 byte PIB attribute identifier, followed by the relevant length of PIB attribute information corresponding to that identifier. For reasons of efficiency, it shall be possible to aggregate SET command on several PIB attributes in one GPDU. The format of such an aggregated payload is shown in figure below:



For cases where the corresponding PIB attribute is only a trigger (all ACTION PIB attributes), there shall be no associated value and the request data format shall be as shown below:



It is assumed that the management entity sending out this information has already determined if the corresponding attributes are supported at target device. This may be achieved by a previous query and its response.

B.6.4.2.4 Reset statistics

This command has optional payload. In case there is no associated payload, the receiving device shall reset all of its PIB statistical attributes.

For cases when a remote management entity only intends to perform reset of selective PIB statistical attributes, the payload shall contain a list of attributes that need to be reset. The format shall be the same as shown in clause B.6.4.2.1

Since there is no confirmation message going back from the device complying with this specification, the management entity needs to send a follow-up PIB attribute query, in case it wants to confirm successful completion of appropriate action.

B.6.4.2.5 Reboot device

There is no corresponding payload associated with this command. The command is complete in itself. The receiving compliant device with this specification shall reboot itself on receipt of this message.

It is mandatory for all implementations compliant with this specification to support this command and its corresponding action.

B.6.4.2.6 Firmware upgrade

The payload in this case shall comprise of firmware upgrade commands and responses described in clause B.6.3 of the specification.

B.6.4.3 NULL SSCS communication profile

This communication profile enables exchange of management information described in previous clauses over the NULL SSCS.

The management entities at both transmitting and receiving ends are applications making use of the NULL SSCS enumerated in clause B.5.3 of this specs. Data is therefore exchanged as MAC Generic PDUs.

B.6.4.4 Serial communication profile

B.6.4.4.1 Physical layer

The PHY layer maybe any serial link (e.g., RS232, USB Serial). The serial link is required to work 8N1 configuration at one of the following data rates:

9600 bps, 19200 bps, 38400 bps, 57600 bps

B.6.4.4.2 Data encapsulation for management messages

In order ensure robustness, the stream of data is encapsulated in HDLC-type frames which include a 2 byte header and 4 byte CRC. All data is encapsulated between a starting flag-byte 0x7E and ending flag-byte 0x7E as shown in Figure below:

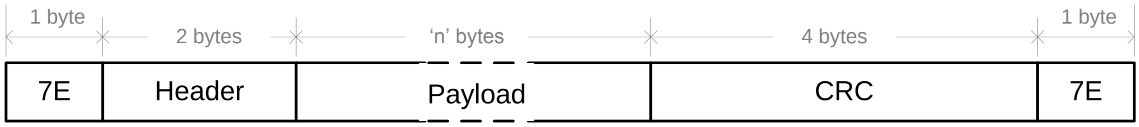


Figure B.90 – Data encapsulations for management messages

If any of the intermediate data characters has the value 0x7E, it is preceded by an escape byte 0x7D, followed by a byte derived from XORing the original character with byte 0x20. The same is done if there is a 0x7D within the character stream. An example of such case is shown here:

```
Msg to Tx:          0x01 0x02 0x7E      0x03 0x04 0x7D      0x05 0x06

Actual Tx sequence: 0x01 0x02 0x7D 0x5E 0x03 0x04 0x7D 0x5D 0x05 0x06

                              Escape              Escape

                              sequence            sequence
```

The 32 bit CRC at end of the frame covers both '*Header*' and '*Payload*' fields. The CRC is calculated over the original data to be transmitted, i.e., before byte stuffing of escape sequences described above is performed. CRC calculation is

The input polynomial M(x) is formed as a polynomial whose coefficients are bits of the data being checked (the first bit to check is the highest order coefficient and the last bit to check is the coefficient of order zero). The Generator polynomial for the CRC is $G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x+1$. The remainder R(x) is calculated as the remainder from the division of M(x)·x32 by G(x). The coefficients of the remainder will then be the resulting CRC.

B.6.5 List of mandatory PIB attributes

B.6.5.1 General

PIB attributes listed in this clause shall be supported by all implementations. PIB attributes that are not listed in this clause are optional and vendors may implement them at their choice. In addition to the PIB attributes, the management command to reboot a certain device (as specified in B.6.4.2.5) shall also be universally supported.

B.6.5.2 Mandatory PIB attributes common to all device types

B.6.5.2.1 PHY PIB attribute

(See Table B.84)

Table B.114 – PHY PIB common mandatory attributes

Attribute name		Id	
phyStatsRxTotalCount		0x00A4	
phyStatsBlkAvgEvm		0x00A5	
phyEmaSmoothing		0x00A8	

B.6.5.2.2 MAC PIB attributes

(See Table B.86, Table B.88 and Table B.89)

Table B.115 – MAC PIB common mandatory attributes

Attribute name	Id				
macEMASmoothing	0x0019				

Attribute name	Id			
MacCapabilities	0x002C			

List Attribute Name	Id
macListPhyComm	0x0057

B.6.5.2.3 Application PIB attributes

(See Table B.92)

Table B.116 – Applications PIB common mandatory attributes

Attribute name	Id
AppFwVersion	0x0075
AppVendorId	0x0076
AppProductId	0x0077

B.6.5.4 Mandatory Base Node attributes

B.6.5.4.1 MAC PIB attributes

(See Table B.86 and Table B.90)

Table B.117 – MAC PIB base node mandatory attributes

Attribute name	Id
macBeaconsPerFrame	0x0013

List attribute name	Id
macListRegDevices	0x0050
macListActiveConn	0x0051

B.6.5.4 Mandatory service node attributes

B.6.5.4.1 MAC PIB attributes

(See Table B.88, Table B.90 and Table B.91)

Table B.118 – MAC PIB service node mandatory attributes

Attribute name	Id
macLNID	0x0020
MacLSID	0x0021
MacSID	0x0022
MacSNA	0x0023
MacState	0x0024
MacSCPLength	0x0025
MacNodeHierarchyLevel	0x0026
MacBeaconSlotCount	0x0027
macBeaconRxSlot	0x0028
MacBeaconTxSlot	0x0029
MacBeaconRxFrequency	0x002A
MacBeaconTxFrequency	0x002B

List attribute name	Id	
macListSwitchTable	0x0053	
macListAvailableSwitches	0x0056	

Attribute name		Id		
MACActionTxData		8		
MACActionConnClose		8		
MACActionRegReject		8		
MACActionProReject		8		
MACActionUnregister		8		

B.6.5.4.2 Application PIB attributes

(See Table B.93)

Table B.119 – APP PIB service node mandatory attributes

Attribute name	Id				
AppFwdlRunning	0x0070				
AppFwdlRxPktCount	0x0071				

PRIME Annex A

MAC layer constants

(Normative)

(This annex forms an integral part of this Recommendation.)

This clause defines all the MAC layer constants.

Table B.A.1 – Table of MAC constants

Constant	Value	Description
MACBeaconLength	4 symbols	Length of beacon in symbols.
MACMinSCPLength	64 symbols	Minimum length of SCP.
MACPriorityLevels	4	Number of levels of priority supported by the system.
MACCFPMaxAlloc	32	Maximum contention-free periods that may be allocated within a frame.
MACFrameLength	276 symbols	Length of a frame in number of symbols.
MACRandSeqChgTime	32767 seconds (approx 9 hours)	Maximum duration of time after which the Base Node should circulate a new random sequence to the Subnetwork for encryption functions.
MACMaxPRNIgnore	3	Maximum number of Promotion-Needed messages a Terminal can ignore.
$N_{miss-beacon}$	5	Number of times a Service Node does not receive an expected beacon before considering its Switch Node as unavailable.
ARQMaxTxCount	5	Maximum Transmission count before declaring a permanent failure.
ARQCongClrTime	2 sec	When the receiver has indicated congestion, this time must be waited before retransmitting the data.
ARQMaxCongInd	7	After ARQMaxCongInd consecutive transmissions which failed due to congestion, the connection should be declared permanently dead.
ARQMaxAckHoldTime	7 sec	Time the receiver may delay sending an ACK in order to allow consolidated ACKs or piggyback the ACK with a data packet.

PRIME Annex B

Convergence layer constants

(Normative)

(This annex forms an integral part of this Recommendation.)

The following TYPE values are defined for use by convergence layers from chapter B.5.

Table B.B.1 – TYPE value assignments

TYPE symbolic name	Value
TYPE_CL_IPv4_AR	1
TYPE_CL_IPv4_UNICAST	2
TYPE_CL_432	3
TYPE_CL_MGMT	4
TYPE_CL_IPv6_AR	5
TYPE_CL_IPv6_UNICAST	6

The following LCID values apply for broadcast connections defined by Convergence layers from chapter B.5.

Table B.B.2 – LCID value assignments

LCID symbolic name	Value	MAC scope
LCI_CL_IPv4_BROADCAST	1	Broadcast.
LCI_CL_432_BROADCAST	2	Broadcast.

The following Result values are defined for Convergence layer primitives.

Table B.B.3 – Result values for convergence layer primitives

Result	Description
Success = 0	The SSCS service was successfully performed.
Reject = 1	The SSCS service failed because it was rejected by the base node.
Timeout = 2	A timed out occurs during the SSCS service processing
Not Registered = 6	The service node is not currently registered to a subnetwork.

Annex I

Application protocol convergence sub-layer

(This annex forms an integral part of this Recommendation.)

This annex applies to the main body of this Recommendation.

Application protocol convergence specific sub-layer (APC) maps the primitives of the application protocol used by the AE into the native protocol of the data link layer. It is the responsibility of the APC to convert incoming data units of the particular application protocol used by the AE into APDUs.

By default, the APC shall support an interface to IPv6 while interfaces to other protocols can also be supported. The APC supporting IPv6 is denoted as IP6-APC. If more than one APC is associated with the particular AE, the bridging function between APCs associated with the AE shall be implemented by the AE and is beyond the scope of this standard.

Each of the APCs generates an APDU based on its particular application protocol that determines the content of the APDU, as described in this annex. At the receive side, the type of the APC that generated a particular APDU is indicated in the LLC frame header (LLCFT field of the LFH). The APC priority and other detailed description of multiple APC support is further study.

The DLL management primitives related to the APC (APC_MGMT) may depend on the application protocol, as defined in this annex, while the set of management primitives related to the lower sub-layers of the ITU-T G.9956 transceiver is unified.

The description of each APC in this annex is partitioned into data plane and management plane. The data plane part specifies conversion of the AE data units into APDUs and back. The management plane part specifies APC address resolution, classification, and other primitives and protocols related to support APC peer-to-peer management services.

I.1 IP6-APC

The IP6-APC is intended to operate with the IPv6 AE and supports transmission of IPv6 datagrams over the medium using ITU-T G.9956 as the underlying L1/L2 layers technology.

In the transmit direction, IP6-APC converts the standard set of IPv6 primitives of the incoming ADP at the A-interface into an APDU, which is further communicated through the ITU-T G.9956 domain(s) to the peer(s) IP6-APC.

At the receiver direction, the IP6-APC converts the APDU received from the LLC into a standard primitive set of an IPv6 datagram0. The IPv6-to-APDU frame conversion shall be as described in clause I.1.1. The IP6-APC shall support address configuration methods, as specified in clause I.1.2.

I.1.1 Frame conversion

The incoming set of primitives (AIF_DATA.REQ) and the outgoing set of primitives (AIF_DATA.IND) at the A-interface of IP6-APC represent an IPv6 datagram submitted by the AE to the APC (for transmission over the medium) and submitted by the APC to the AE (after it has been received from the medium), respectively. Each datagram is defined as a set of IPv6 primitives specified by [IETF RFC 2460] of IPv6_datagram.request and IPv6_datagram.indication, respectively – see Table I.1 and Table I.2.

Table I.1 – A-interface primitives

AIF_DATA.REQ (AE → IP6APC)	**AIF_DATA.IND (IP6APC → AE)**
IPv6_datagram.request (version traffic class flow label payload length next header hop limit source address destination address data_payload)	IPv6_datagram.indication (version traffic class flow label payload length next header hop limit source address destination address data_payload)

Table I.2 – A-interface primitives description (RFC 2460)

Primitive name	**Description**
version	4-bit Internet Protocol version number = 6
traffic class	8-bit traffic class field
flow label	20-bit flow label
payload length	16-bit integer – length of the data_payload in octets
next header	8-bit selector – identifies the type of the header immediately following the IPv6 header.
hop limit	8-bit unsigned integer – decremented by 1 by each node that forwards the datagram (at L3)
source address	The IPv6 datagram source address in a standard 16-octet format
destination address	The IPv6 datagram destination address in a standard 16-octet format (Note 1)
data_payload	The IPv6 datagram payload to carry from the source address to the destination address.
NOTE 1 – The IPv6 Neighbour Discovery protocol, [IETF RFC 4861], provides a link layer final destination address in a format that meets the underlying data link layer, as defined in clause 8 of [IETF RFC 4944]. For ITU-T G.9956, short and extended format of link layer address are defined, see clause 8.3.1.	

The primitives described in Table I.1 for AIF_DATA.REQ shall be converted into the APDU format presented in Figure I.1, containing a header and a payload. The Table I.1 includes only primitives of the IPv6 header (as per clause 3 of [IETF RFC 2460]), while IPv6 header extensions, if present, are considered as a part of the *data_payload* primitive. The total length of the APDU shall not exceed 1522 octets.

Header		LSB MSB
	≤ 40 octets	IPv6 header of the incoming datagram, uncompressed or compressed using one of the valid compression methods described in Annex J (Note 1)
Payload	≤ 1482 octets	The *data_payload* primitive of the incoming IPv6 datagram (Note 2)
NOTE 1 – The IPv6 header format is defined in clause 3 of [IETF RFC 2460] and shall be used as a default (uncompressed header). The valid compression methods are defined in Annex J, which can be used in conjunction with this annex. The used compression method is indicated in the LLCFT field of LFH (see clause 8.3.1.2). NOTE 2 – For extension headers (e.g., for UDP header), if present, only the compression options associated with the applied IPv6 header compression method defined in Annex J shall be used.		

Figure I.1 – IP6-APC APDU format

Bits of APDU shall be transmitted towards LLC (x1 reference point) starting from the first octet of the header. LSB shall be transmitted first.

The order of outgoing APDUs at the x1 reference point associated with a particular destination and particular user priority shall be the same as the order of incoming IPv6 datagrams of these same user priority and destination. No re-ordering inside the same user priority group for the same destination is allowed.

The receive path of the IP6-APC shall decompress the header and the payload of the APDU received from the x1 reference point, if compression was applied, as defined in Annex J, and reconstruct the original datagram primitives required for the AIF_DATA.IND (see Table I.1).

I.1.2 Address assignment and address resolution

I.1.2.1 Stateless address auto-configuration

An ITU-T G.9956 node is assigned with a short address (16-bit NODE_ID) upon its registration into the domain. In case of the L2 registration procedure, defined in clause 8.6.3, this short address is communicated to the AE via the A-interface (A_MGMT.IND primitive, clause 8.4.3.4.3). Further, the AE shall configure its IPv6 interface identifier of the link local address (local scope) based on the received node address, its IPv6 interface identifier of the link local address (global scope) based on the node EUI-64 address according to clause 3.2.2 of [IETF RFC 6282]. In case of L3 registration procedure, the IPv6 link local address and the corresponding short address are generated by the applied IPv6-based protocol, which is beyond the scope of the Recommendation. These addresses shall be communicated to the DLL via the A-interface (A_MGMT.REQ primitive, clause I.1.6.1.1).

I.1.2.2 Stateful address assignment

With stateful assignment, the IPv6 link local address is assigned independently from the node short address (16-bit NODE_ID, 16-bit DOMAIN_ID) or extended address. Thus, there is no generic rule to derive a short address or extended address from IPv6 address and vice versa. The address resolution shall be performed as defined in I.1.2.3.

I.1.2.3 Address resolution function (ARF)

The IPv6 Neighbour Discovery protocol, [IETF RFC 4861], provides a link layer final destination address of a node corresponding to the destination IPv6 address, thus providing the address resolution.

In the transmit direction, the short address and extended destination addresses are passed from the AE to the DLL via the AIF_DATA.REQ primitive defined in clause I.1.1.

The APC shall pass the resolved 16-bit or 64-bit DA (unicast, multicast, or broadcast) as a control primitive IP6-APC_MGMT.IND to the LLC via x1-reference point (see clause I.1.6.2.3). At the destination node, short or extended addresses recovered from the received APDUs (DA, SA) are passed to the APC from the LLC via x1-reference point as a control primitive IP6-APC_MGMT.REQ (see clause I.1.6.2.1).

In case the assignment of IPv6 address is by using stateless auto-configuration (see clause I.1.2.1), the IPv6 interface identifier of the link local address can be derived from the 16-bit NODE_ID of a node or from the EUI-64 address of a node as defined in clause 3.2.2 of [IETF RFC 6282].

I.1.3 Classification

IP6-APC may classify the outgoing APDUs based on the following AIF_DATA.REQ primitives:

- Traffic Class
- Flow label
- Destination Address
- Source Address
- Special management primitives (e.g., emergency indicators) associated with the AE

As a result of classification, the outgoing APDU is associated with a particular user priority: 0, 1, 2, or 3. The presented criteria indicate possible classification options; the rules of how to classify APDUs, assign a priority, and associate it with a particular user priority, except the restrictions presented below, depend of the used L3 protocol and are beyond the scope of this Recommendation. The restrictions are:

1. Priority 3 shall be assigned only to APDUs that carry emergency primitives definition of these emergency primitives is for further study.
2. The number of user priorities assigned by the APC shall not exceed the number of MA priority queues supported by the node (which depends on profile).
3. The APDUs generated from ADPs with no or irrelevant QoS requirements shall be assigned priority 0.

Other criteria and restrictions for classification of the APDUs are beyond the scope of this Recommendation.

I.1.4 Flow control

Flow control at the A-interface is necessary to avoid packet loss in the case when the traffic generated by the source AE exceeds the throughput of the link between the source and the destination nodes. The flow control may be implemented by communicating an appropriate set of AIF_DATA.IND primitives from the APC to AE (e.g., causing a pause in the stream of incoming datagrams) or a set of AIF_DATA.CNF primitives, or by appropriate signalling at the management plane. The format of AIF_DATA.CNF and signalling used for flow control is vendor discretionary.

I.1.5 Encryption

In case the LLC frame carrying an APDU shall be encrypted, the header of the APDU shall be left unencrypted ("Unencrypted part of the APDU" in Figure 8-22), while the payload of the APDU shall be encrypted.

I.1.6 Management plane

I.1.6.1 A_MGMT primitives

The DLL management entity supports operation of IP6-APC using a set of management primitives A_MGMT presented in Table I.3 (A-reference point).

Table I.3 – IP6-APC A_MGMT primitives

Primitive	Direction	Description
A_MGMT.REQ	AE → DLL	Management request from AE to DLL
A_MGMT.CNF	DLL → AE	Confirmation from DLL to AE on A_MGMT.REQ
A_MGMT.IND	DLL → AE	Management parameters from DLL to AE

I.1.6.1.1 A_MGMT.REQ

This primitive requests the DLL or PHY management to use particular parameters for associated functions. The attributes of the primitive are defined in Table I.4.

Table I.4 – The attributes of the A_MGMT.REQ primitive

Name	Type	Valid range	Description

Table left blank intentionally. Additional primitives are for further study.

I.1.6.1.2 A_MGMT.CNF

This primitive confirms the particular parameters used by the DLL or PHY. The attributes of the primitive are as defined in Table I.5.

If the DLL or PHY is unable to comply with a particular attribute in the A_MGMT.REQ, it shall set this primitive to one, which means that the request is denied. Otherwise the value of the A_MGMT.CNF primitive shall be set to zero.

Table I.5 – The attributes of the A_MGMT.CNF primitive

Name	Type	Valid range	Description
Status	Integer	0, 1	0: success 1: request is denied

I.1.6.1.3 A_MGMT.IND

This primitive provides the AE management with management parameters received or derived by the DLL or PHY management and submitted to the A-interface. The attributes of the primitive are defined in Table I.6.

Table I.6 – The attributes of the A_MGMT.IND primitive

Name	Type	Valid range	Description
NODE_ID	16-bit integer	See Table 8-7	NODE_ID of the node obtained during the registration to the domain
Extended address	64-bit integer	See EUI-64™	EUI-64 address of the node (assigned by the manufacturer)
L3 routing primitives	N/A	N/A	A set of primitives to support IPv6 L3 routing. The nomenclature and format is for further study.
Performance primitives	N/A	N/A	A set of L1 and L2 performance primitives to support IPv6 L3 routing. The nomenclature and primitives are for further study.

Additional primitives are for further study.

I.1.6.2 IP6-APC_MGMT primitives

I.1.6.2.1 IP6-APC_MGMT.REQ

This primitive requests the APC to use particular parameters for associated functions. The attributes of the primitive are defined in Table I.7.

Table I.7 – The attributes of the IP6-APC_MGMT.REQ primitive

Name	Type	Valid range	Description
SA (Note 1)	16-bit integer	See Table 8-12	Short address (NODE_ID) of the node that is an originating source address of the received APDU
	64-bit integer	See EUI-64™	Extended address (EUI-64) of the node that is an originating source address of the received APDU
SA-TYPE (Note 2)	1-bit integer	0, 1	0 – SA of the received APDU is represented in extended address format 1 – SA of the received APDU is represented in short address format
RX-FrameType	4-bit integer	0010-1111	Indicates the format of the received APDU (APC type, uncompressed or compressed), with coding as defined in Table 8-3.
NOTE 1 – Only one of the representations shall be available. NOTE 2 – In case the LFH of the received LLC frame carrying the APDU includes a mesh header (see clause 8.1.2.1.2), the value and the format of the SA shall be the one indicated in the mesh header. Otherwise, the SA shall be a short address indicated in the SNID field of the MPH of the MPDUs conveyed by the segments of the received LLC frame (see clause 8.1.3.1).			

Additional primitives are for further study.

NOTE – In case LLC frame is conveyed using more than one MPDU, the MPH of these MPDUs have identical SNID fields and identical DNID fields.

I.1.6.2.2 IP6-APC_MGMT.CNF

This primitive confirms the particular parameters used by the APC. The attributes of the primitive are as defined in Table I.8.

If the APC is unable to comply with a particular attribute in the IP6-APC_MGMT.REQ, it shall set this primitive to one, which means that the request is denied. Otherwise the value of the IP6-APC_MGMT.CNF primitive shall be set to zero.

Table I.8 – The attributes of the IP6-APC_MGMT.CNF primitive

Name	Type	Valid range	Description
Status	Integer	0, 1	0: success 1: request is denied

Additional primitives are for further study.

I.1.6.2.3 IP6-APC_MGMT.IND

This primitive provides the DLL management with parameters of the datagram received by the IP6-APC from the A-interface. The attributes of the primitive are defined in Table I.9.

Table I.9 – The attributes of the IP6-APC_MGMT.IND primitive

Name	Type	Valid range	Description
LPRI	2-bit integer	00_2-11_2	User priority of the APDU
DA (Note 1)	16-bit integer	See Table 8-12	Short address (NODE_ID) of the node that is a final destination address of the transmitted APDU
	64-bit integer	See EUI-64™	Extended address (EUI-64) of the node that is a final destination address of the transmitted APDU
DA-TYPE (Note 2)	1-bit integer	0, 1	0 – DA of the transmitted APDU to be represented in extended address format 1 – DA of the transmitted APDU to be represented in short address format
TX-FrameType	4-bit integer	0010_2-1111_2	Indicates the format APDU shall be transmitted (APC type, uncompressed or compressed) with coding as defined in Table 8-3.
APDU length	16-bit integer	0001_{16} – $05F2_{16}$	Indicate the APDU length in bytes
NOTE 1 – Only one of the representations shall be available. NOTE 2 – In case the LFH of the transmitted LLC frame carrying the APDU includes a mesh header (see clause 8.1.2.1.2), the value and the format of the DA shall be one indicated in the mesh header. Otherwise, the DA shall be a short address indicated in the DNID field of the MPH of the MPDUs conveyed the segments of the received LLC frame (see clause 8.1.3.1).			

Additional primitives are for further study.

I.1.6.2.4 IP6-APC_MGMT.RES

This primitive is for further study.

Annex J

IP6 header compression methods

(This annex forms an integral part of this Recommendation.)

This annex applies to the main body of this Recommendation.

This annex describes valid compression methods for IPv6 header and header extensions. It is intended to be used in conjunction with Annex I, clause I.1.

NOTE – Some of the IPv6 compression methods defined in this annex are not transparent to some of AIF_DATA primitives (e.g., traffic class, flow labels).

The default compression method shall use the APDU format that includes a compressed IPv6 header and non-compressed APDU payload. The format of the APDU compressed header shall be as presented in Table J.1.

Table J.1 – APDU header – description

IPv6 header field	Description, [IETF RFC 2460]	Compressed format
version	4-bit Internet Protocol version number = 6	0 bits
traffic class	8-bit traffic class field	0 bits
flow label	20-bit flow label	0 bits
payload length	16-bit integer – length of the data_payload in octets	0 bits
next header	8-bit selector – identifies the type of the header immediately following the IPv6 header.	Octet 0, original format
hop limit	8-bit unsigned integer – decremented by 1 by each node that forwards the datagram (at L3)	Octet 1, original format
source address	The IPv6 datagram source address in a standard 16-octet format	0 bits (Note 1)
destination address	The IPv6 datagram destination address in a standard 16-octet format (Note 1)	Unicast: 0 bits Multicast: octets 2-17 original format (Note 2)
NOTE 1 – The IID (IPv6 Interface Identifier) shall be recovered by the receiver based on the IPv6 address association with L2 addresses. The IPv6 address prefix shall be set to the value of the receiver's address prefix. NOTE 2 – For unicast, the address shall be recovered as defined in Note 1. For multicast, the transmitter shall use the original format of the destination addresses.		

NOTE – The value of payload length field of the IPv6 header is communicated using parameters of the MPH: LPDU-L, NOS, and LSPL. Receiver uses these parameters to compute the size of the received APDU as defined in clause 8.1.2.2, clause 8.1.3.1.

Besides default compression, the IPv6 datagram may be either sent uncompressed or, if applicable, use one of the alternative valid compression options described in Table J.2. Support of valid compression methods is optional, thus one can be used only in case this compression method is supported by the peer IP6-APC.

NOTE – The default compression is only suitable to L3 networks where all nodes have the same prefix.

Table J.2 – Alternative valid compression methods

Compression method	LFH indication code	Description
HC1 and HC2 of RFC 4944	4	Compression as per clause 10 of [IETF RFC 4944]
LOWPAN_IPHC and LOWPAN_NHC of RFC 6282	5	Compression as per [IETF RFC 6282]
NOTES: 1 – For extension headers, if present, only the compression options defined by the applied IPv6 header compression method shall be used. For instance, with the IPV6 compression method defined in clause 10.1 of [IETF RFC 4944], the UDP header, if present, may be compressed as defined in clause 10.2 of [IETF RFC 4944]. 2 – In [IEEE 802.3] and clause 10 of [IETF RFC 4944], it is required to use the "Frame Length" field of [IEEE 802.15.4] to recover the length of the PPDU. For an equivalent computation in case of ITU-T G.9956, to recover the length of the APDU it is required to use the relevant fields of MPH. In ITU-T G.9956 the 6LoWPAN fragmentation header is not used.		

Other compression methods are for further study.

Bibliography

[b-TR-069] Broadband Forum TR-069 (2007), *CPE WAN Management Protocol v1.1*.

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	**Transmission systems and media, digital systems and networks**
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Cable networks and transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Construction, installation and protection of cables and other elements of outside plant
Series M	Telecommunication management, including TMN and network maintenance
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Terminals and subjective and objective assessment methods
Series Q	Switching and signalling
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks, open system communications and security
Series Y	Global information infrastructure, Internet protocol aspects and next-generation networks
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2014