



**IEEE Recommended Practice for
Information technology—
Telecommunications and information
exchange between systems—
Local and metropolitan area networks—
Specific requirements**

**Part 15.5: Mesh Topology Capability in
Wireless Personal Area Networks (WPANs)**

IEEE Computer Society

Sponsored by the
LAN/MAN Standards Committee

IEEE
3 Park Avenue
New York, NY 10016-5997, USA

8 May 2009

IEEE Std 802.15.5™-2009

**IEEE Recommended Practice for
Information technology—
Telecommunications and information
exchange between systems—
Local and metropolitan area networks—
Specific requirements**

**Part 15.5: Mesh Topology Capability in
Wireless Personal Area Networks (WPANs)**

Sponsor

**LAN/MAN Standards Committee
of the
IEEE Computer Society**

Approved 19 March 2009

IEEE-SA Standards Board

Abstract: This IEEE recommended practice defines the architectural framework that enables WPAN devices to promote interoperable, stable, and scaleable wireless mesh topologies and, if needed, to provide the amendment text to the current WPAN standards that is required to implement this recommended practice.

Keywords: address assignment, block addressing, broadcast, carrier sense multiple access/collision avoidance, high-rate WPAN mesh, HR-WPAN mesh, multicast, low-rate WPAN mesh, LR-WPAN mesh, mesh, mesh coordinator, server routing, wakeup interval, multiple hop, multi-hop, local link state, logical tree, portable, portability, power saving, reliable broadcast, sensor network, traceroute, unicast, wireless PAN, WPAN

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

Copyright © 2009 by the Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 8 May 2009. Printed in the United States of America.

IEEE and 802 are registered trademarks in the U.S. Patent & Trademark Office, owned by the Institute of Electrical and Electronics Engineers, Incorporated.

PDF: ISBN 978-0-7381-5917-1 STD95913
Print: ISBN 978-0-7381-5918-8 STDPD95913

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher.

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. The IEEE develops its standards through a consensus development process, approved by the American National Standards Institute, which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and serve without compensation. While the IEEE administers the process and establishes rules to promote fairness in the consensus development process, the IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

Use of an IEEE Standard is wholly voluntary. The IEEE disclaims liability for any personal injury, property or other damage, of any nature whatsoever, whether special, indirect, consequential, or compensatory, directly or indirectly resulting from the publication, use of, or reliance upon this, or any other IEEE Standard document.

The IEEE does not warrant or represent the accuracy or content of the material contained herein, and expressly disclaims any express or implied warranty, including any implied warranty of merchantability or fitness for a specific purpose, or that the use of the material contained herein is free from patent infringement. IEEE Standards documents are supplied “**AS IS**.”

The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation. When a document is more than five years old and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

In publishing and making this document available, the IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity. Nor is the IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing this, and any other IEEE Standards document, should rely upon his or her independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration. A statement, written or oral, that is not processed in accordance with the IEEE-SA Standards Board Operations Manual shall not be considered the official position of IEEE or any of its committees and shall not be considered to be, nor be relied upon as, a formal interpretation of the IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position, explanation, or interpretation of the IEEE.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Comments on standards and requests for interpretations should be submitted to the following address:

Secretary, IEEE-SA Standards Board
445 Hoes Lane
Piscataway, NJ 08854
USA

Authorization to photocopy portions of any individual standard for internal or personal use is granted by The Institute of Electrical and Electronics Engineers, Inc., provided that the appropriate fee is paid to Copyright Clearance Center. To arrange for payment of licensing fee, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Introduction

This introduction is not part of IEEE Std 802.15.5-2009, IEEE Recommended Practice for Information technology—Telecommunications and information exchange between systems— Local and metropolitan area networks— Specific requirements.

This recommended practice provides the architectural framework enabling WPAN devices to promote interoperable, stable, and scalable wireless mesh topologies. This recommended practice is composed of two parts: low-rate WPAN mesh and high-rate WPAN mesh networks. The low-rate mesh is built on IEEE 802.15.4 MAC, while high rate mesh utilizes IEEE 802.15.3/3b MAC. Common features of both meshes include network initialization, addressing, and multihop unicasting. In addition, low-rate mesh supports multicasting, reliable broadcasting, portability support, trace route and energy saving function, and high-rate mesh supports multihop time-guaranteed service.

Notice to users

Laws and regulations

Users of these documents should consult all applicable laws and regulations. Compliance with the provisions of this standard does not imply compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Copyrights

This document is copyrighted by the IEEE. It is made available for a wide variety of both public and private uses. These include both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making this document available for use and adoption by public authorities and private users, the IEEE does not waive any rights in copyright to this document.

Updating of IEEE documents

Users of IEEE standards should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect. In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit the IEEE Standards Association web site at <http://ieeexplore.ieee.org/xpl/standards.jsp>, or contact the IEEE at the address listed previously.

For more information about the IEEE Standards Association or the IEEE standards development process, visit the IEEE-SA web site at <http://standards.ieee.org>.

Errata

Errata, if any, for this and all other standards can be accessed at the following URL: <http://standards.ieee.org/reading/ieee/updates/errata/index.html>. Users are encouraged to check this URL for errata periodically.

Interpretations

Current interpretations can be accessed at the following URL: <http://standards.ieee.org/reading/ieee/interp/index.html>.

Patents

Attention is called to the possibility that implementation of this recommended practice may require use of subject matter covered by patent rights. By publication of this recommended practice, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this recommended practice are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

Participants

At the time this recommended practice was completed, the IEEE P802.15 Working Group had the following officers:

Dr. Robert F. Heile, *Working Group Chair*
Rick Alfvén, *Working Group Vice Chair*
Patrick Kinney, *Working Group Vice Chair & Secretary*
Dr. James P.K. Gilb, *Working Group Technical Editor*
Dr. John Barr, *Working Group Treasurer*

At the time this recommended practice was completed, the IEEE P802.15 Working Group had the following officers:

Dr. Myung Jong Lee, *Task Group 5 Chair*
Dr. Ho-in Jeon, *Task Group 5 Vice Chair*
Dr. Tae Rim Park, *Task Group 5 Secretary*
Dr. Chunhui (Allan) Zhu, *Task Group 5 Low-Rate Editor*
Dr. Sang Sung Choi, *Task Group 5 High-Rate Editor*

Authur Astrin
Taehan Bae
Jay Bain
Gal Basson
Tuncer Baykas
Phil Beecher
Bruce Bosco
Andre Bourdoux
Pat Carson
Philippe Chambelin
Huor-Hsin Chang
Chang-Soon Choi
Carlos Cordeiro
Alexey Davydov
Hendricus De Ruijter
Paul Dixon
Kai Dombrowski
John Dorsey
Bas Driesen
Amal Ekbal
Yossi Erlich
Robert Fanfelle
John Farserotu
Reed Fisher
Yoshitsugu Fujita
Kiyoshi Fukui
Shigeru Fukunaga
Ryuhei Funada
Uhland Goebel
Paul Gorday
Giriraj Goyal
Eckhard Grass
Mark Grodzinsky
Vivek Gupta
Robert Hall
Christopher J. Hansen
Shinsuke Hara
Hiroshi Harada
Seockdeock Hong
Tian-Wei Huang
Ichirou Ida
Hideto Ikeda
Tetsushi Ikegami
Akio Iso
Beomjin Jeon
Young-Ae Jeon
Seong-Soon Joo
Chol Su Kang
Tae-Gyu Kang
Yasunao Katayama
Shuzo Kato

Yasutaka Kawamoto
Stuart Kerry
Jaehwa Kim
Jae-Hyon Kim
Jinkyong Kim
Kihong Kim
Kyeongpyo Kim
Seong Kim
Yongsun Kim
Ryota Kimura
Kursat Kimyacioglu
Ryuji Kohno
Fumihide Kojima
Edwin Kwon
Hyoungjin Kwon
Ismail Lakkis
John Lampe
Zhou Lan
Jae Lee
Jeong Lee
Myung Lee
Seong-hee Lee
Taehoon Lee
Wooyong Lee
Zhongding Lei
Daniel Lewis
Huan-Bang Li
Liang Li
Sheung Li
Yong Liu
Alexander Maltsev
Abbie Mathew
Taisuke Matsumoto
Michael Mcinnis
Michael Mclaughlin
Klaus Meyer
Dino Miniutti
Rajendra Moorti
Jorge Myszne
Yukimasa Nagai
Ken Naganuma
Chiu Ngo
Paul Nikolich
Yoshinori Nishiguchi
Hiroyo Ogawa
Jisung Oh
Laurent Ouvry

Pascal Pagani
Tae Rim Park
Maulin Patel
Stephane Pinel
Frank Poegel
Stephen Pope
Clinton Powell
Chang Woo Pyo
Xiangping Qin
Ivan Reede
Richard Roberts
Benjamin A. Rolfe
Ali Sadri
Katsuyoshi Sato
Hirokazu Sawada
Kamran Sayrafian
Michael Schmidt
Jean Schwoerer
Huai-Rong Shao
Stephen Shellhammer
Shusaku Shimada
Yukihiro Shimakata
Chang Sub Shin
Michael Sim
Harkirat Singh
Carl Stevenson
Paul Strauch
Chin Sum
Kazuaki Takahashi
Kenichi Takizawa
Arnaud Tonnerre
Ichihiko Toyoda
Jason Trachewsky
Solomon Trainin
Alberto Valdes Garcia
Magnus Wiklund
Gerald Wineinger
Ludwig Winkel
Eun Tae Won
Jongeeun Won
Pengfei Xia
Kamya Yazdandoost
James Yee
Kaoru Yokoo
Su Yong
Zhan Yu
Bin Zhen
Chunhui Zhu

Major contributions for this recommended practice were received from the following individuals:

James Allen
Jay Bain
Phil Beecher
John Boot
Ishikawa Chiaki
Sungrae Cho
Sang Sung Choi
Wonsuk Choi
Chun-Ting Chou
Francis daCosta
Klaus Fosmark
James P.K. Gilb
Uhland Goebel
Guido R. Hiertz

Ho-In Jeon
Young-Ae Jeon
Seong-Soon Joo
Hakyung Jung
Patrick Kinney
Seongjae Kwon
Inhwan Lee
Myung Jong Lee
Seong-hee Lee
Yong Liu
Sebastian Max
Klaus Meyer
Laihyuk Park

Sung-Woo Park
Frank Poegel
Hans-Jürgen Reuerman
Seung Hyong Rhee
Benjamin A. Rolfe
Michael Schmidt
Huai-Rong Shao
Chang Sub Shin
Michael Sim
Marcus Wong
June Yoon
Rui Zhang
Jianliang Zheng
Chunhui Zhu

The following members of the individual balloting committee voted on this recommended practice. Balloters may have voted for approval, disapproval, or abstention.

Richard Alfvén
James Allen
Danilo Antonelli
Philip Beecher
Lyle Bullock
William Byrd
James Carlo
Sang Sung Choi
Keith Chow
Charles Cook
Todor Cooklev
Thomas Dineen
Sourav Dutta
Paul Eastman
Richard Eckard
Marc Emmelmann
Avraham Freedman
Devon Gayle
James P.K. Gilb
Randall Groves
C. Guy
Rainer Hach
John Hawkins
Marco Hernandez
Atsushi Ito

Beomjin Jeon
Young-Ae Jeon
Bobby Jose
Kaku, Shinkyō
Stuart J. Kerry
Yongbum Kim
Patrick Kinney
Cees Klik
Joseph Kubler
Myung Jong Lee
Seong-hee Lee
Jan-Ray Liao
Arthur Light
William Lumpkins
Peter Martini
Michael Mcinnis
Wade Midkiff
Avygdor Moise
Hiroyuki Nakase
Michael S. Newman
Charles Ngethe
John Notor
Satoshi Obara
Knut Odman
Okundu Omeni
Satoshi Oyama

Tae Rim Park
Eldad Perahia
Subburajan Ponnuswamy
Clinton Powell
Robert Robinson
Benjamin Rolfe
Shigenobu Sasaki
Peter Saunderson
Bartien Sayogo
Chang Sub Shin
Kapil Sood
Amjad Soomro
Thomas Starai
Rene Struik
Walter Struppler
Mark Sturza
Thomas Tullia
Dmitri Varsanofiev
Prabodh Varshney
Stanley Wang
Andreas Wolf
Paul Work
Oren Yuen
Wenhao Zhu
Juan Zuniga

When the IEEE-SA Standards Board approved this recommended practice on 19 March 2009, it had the following membership:

Robert M. Grow, *Chair*
Thomas Prevost, *Vice Chair*
Steve M. Mills, *Past Chair*
Judith Gorman, *Secretary*

John Barr
Karen Bartleson
Victor Berman
Ted Burse
Dick DeBlasio
Andy Drozd
Mark Epstein

Alexander Gelman
Jim Hughes
Richard Hulett
Young Kyun Kim
Joseph L. Koepfinger*
John Kulick
David J. Law

Ted Olsen
Glenn Parsons
Ron Petersen
Narayanan Ramachandran
Jon Walter Rosdahl
Sam Sciacca
Howard Wolfman

*Member Emeritus

Also included are the following nonvoting IEEE-SA Standards Board liaisons:

Satish K. Aggarwal, *NRC Representative*
Michael Janezic, *NIST Representative*

Michelle D. Turner
IEEE Standards Program Manager, Document Development

Michael D. Kipness
IEEE Standards Program Manager, Technical Program Development

Contents

1. Overview	1
1.1 Scope	1
1.2 Purpose	2
2. Normative references.....	2
3. Definitions	2
4. Acronyms	4
5. Low-rate WPAN mesh	7
5.1 General description.....	7
5.2 Mesh service description	9
5.2.1 Mesh data service.....	9
5.2.2 Mesh management service.....	15
5.3 Frame formats.....	41
5.3.1 General mesh service frame format	41
5.3.2 Format of individual frame types	44
5.3.3 Mesh sublayer information in MAC beacon payload	62
5.4 Mesh sublayer constants, mesh information base and statue values.....	64
5.4.1 Mesh sublayer constants	64
5.4.2 Mesh information base.....	64
5.4.3 Mesh status values	69
5.5 Mesh function description	70
5.5.1 Starting a mesh network.....	70
5.5.2 Joining a mesh network.....	71
5.5.3 Address assignment	72
5.5.4 Mesh topology discovery and formation.....	74
5.5.5 Mesh path selection and data forwarding.....	77
5.5.6 Mesh path maintenance.....	79
5.5.7 Leaving a mesh network	80
5.5.8 Mesh path selection and forwarding for multicast	82
5.5.9 Reliable broadcast.....	93
5.5.10 Energy Saving in Battery-Powered Networks.....	95
5.5.11 Portability support.....	109
5.5.12 Traceroute	111
6. High-rate WPAN mesh.....	114
6.1 General description.....	114
6.2 MHME SAP interface.....	115
6.2.1 Generic management primitives	115
6.2.2 Resetting the mesh sublayer/MHME	117
6.2.3 Scanning for mesh networks	118
6.2.4 Starting a mesh-enabled piconet	122
6.2.5 Stopping a mesh network.....	124
6.2.6 Associating with a mesh network	125
6.2.7 Disassociation from a mesh network	127
6.2.8 TREEID assignment	129
6.3 Mesh SAP interface	130
6.3.1 MESH-ASYNC-DATA.request.....	133
6.3.2 MESH-ASYNC-DATA.confirm.....	133

6.3.3 MESH-ASYNC-DATA.indication	133
6.3.4 MESH-ISOCH-DATA.request	134
6.3.5 MESH-ISOCH-DATA.confirm	134
6.3.6 MESH-ISOCH-DATA.indication	134
6.4 Mesh PIB	135
6.5 Frame formats	135
6.5.1 General mesh frame format	135
6.5.2 Format of individual frame types	137
6.5.3 Command types	138
6.6 Mesh service support	144
6.6.1 Starting a mesh network	144
6.6.2 Tree formation	146
6.6.3 TREEID assignment	148
6.6.4 Leaving a mesh network	151
6.6.5 Routing procedure	152
6.6.6 Routing Alternative	155
Annex A (informative) Amendment to MAC sublayer	158

IEEE Recommended Practice for Information technology— Telecommunications and information exchange between systems— Local and metropolitan area networks— Specific requirements

Part 15.5: Mesh Topology Capability in Wireless Personal Area Networks (WPANs)

IMPORTANT NOTICE: This standard is not intended to ensure safety, security, health, or environmental protection in all circumstances. Implementers of the standard are responsible for determining appropriate safety, security, environmental, and health practices or regulatory requirements.

This IEEE document is made available for use subject to important notices and legal disclaimers. These notices and disclaimers appear in all publications containing this document and may be found under the heading “Important Notice” or “Important Notices and Disclaimers Concerning IEEE Documents.” They can also be obtained on request from IEEE or viewed at <http://standards.ieee.org/IPR/disclaimers.html>

1. Overview

A Wireless Personal Area Network (WPAN) mesh is a network of WPANs with mesh topology. This document defines a recommended practice for WPAN meshes.

1.1 Scope

The scope of this standard is to provide a recommended practice to provide the architectural framework enabling WPAN devices to promote interoperable, stable, and scaleable wireless mesh topologies and, if needed, to provide the amendment text to the current WPAN standards that is required to implement this recommended practice.

1.2 Purpose

The purpose of this project is to facilitate wireless mesh topologies optimized for IEEE 802.15 WPANs.

Mesh Topology provides the following features to WPANs:

- Extension of network coverage without increasing the transmit power or the receiver sensitivity
- Enhanced reliability via route redundancy
- Easier network configuration
- Better device battery life

2. Normative references

The following referenced documents are indispensable for the application of this document (i.e., they must be understood and used, so each referenced document is cited in text and its relationship to this document is explained). For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments or corrigenda) applies.

IEEE Std 802.15.3™-2003, IEEE Standard for Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements Part 15.3: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for High Rate Wireless Personal Area Networks (WPANs).^{1,2}

IEEE Std 802.15.3b™-2005, IEEE Standard for Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements Part 15.3: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for High Rate Wireless Personal Area Networks (WPANs) Amendment 1: MAC Sublayer.

IEEE Std 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 Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (WPANs).

3. Definitions

For the purposes of this draft recommended practice, the following terms and definitions apply. *The Authoritative Dictionary of IEEE Standards Terms* should be referenced for terms not defined in this clause.

3.1 active duration: A time period defined in both synchronous and asynchronous energy saving modes in the mesh sublayer during which a mesh device accesses the common channel using carrier sense multiple access with collision avoidance mechanism.

¹ IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, Piscataway, NJ 08854, USA (<http://standards.ieee.org/>).

² The IEEE standards or products referred to in this clause are trademarks of the Institute of Electrical and Electronics Engineers, Inc.

- 3.2 ancestor:** A device that is closer to the root of the logical tree than this device along the tree.
- 3.3 child device:** A device that associates with the device. A child device's tree level is one more than that of the device. If a device's tree level equals to i , then all of its child devices (children) will have tree level equal to $i+1$.
- 3.4 descendant:** A device that is farther from the root of the logical tree than the device.
- 3.5 end device:** A device that is not capable of relaying data frames for other devices.
- 3.6 flooding:** A broadcast method in which each device, other than the original sender, receives and relays the same broadcast data frames exactly once.
- 3.7 group coordinator:** The controller of a specific multicast group.
- 3.8 group member:** A device participates in a multicast communication.
- 3.9 inactive duration:** A time period defined in both synchronous and asynchronous energy saving modes in mesh sublayer during which a mesh device may turn off its receiver circuitry to save energy. An inactive duration follows an active duration in time.
- 3.10 extended neighbor:** A device less than or equal to meshExtendedNeighborHopDistance-hops away from the device, where meshExtendedNeighborHopDistance is a Mesh Information Base (MeshIB) attribute. Neighbors more than one hop away are not able to talk to each other directly.
- 3.11 mesh address:** A logical address assigned to a mesh device in the mesh sublayer. Each mesh address is unique in a mesh network so that a mesh device can be uniquely identified.
- 3.12 mesh coordinator:** An entity that manages the mesh network. It refers to the root of the logical tree of a mesh network whose tree level equals zero.
- 3.13 mesh device:** A device that is capable of relaying data frames for other devices.
- 3.14 mesh discovery table:** A temporary table that records the information of all neighbor devices discovered by the mesh discovery process.
- 3.15 mesh sublayer timer:** A timer for an optional energy saving mode for mesh sublayer.
- 3.16 mobility:** A feature that allows a device to be moved from one location to another within the network coverage while maintaining the communication session during the move.
- 3.17 multicast router:** A device that is not a multicast group member but relays multicast data frames for one or more multicast groups.
- 3.18 multicast agent:** A device with at least one of its end devices that is a member of a multicast group.
- 3.19 neighbor:** A device that is in the communications range of the current device.
- 3.20 neighbor MPNC:** MPNCs that are single hop away from an MPNC with a direct link.
- 3.21 optimal route:** A minimal cost route which is provided by topology server.
- 3.22 portability:** A feature that allows a device to be moved from one location to another within the network coverage and to resume operation at the new location. In contrast to mobility, the communication session is not maintained during the move.

3.23 region synchronizer: A mesh device that is responsible for sending synchronization request frames to its child devices.

3.24 residual CTA time: The channel time assignment time available at the MPNC.

3.25 server routing: A routing method which is done by topology servers distributed over the mesh network.

3.26 sibling device: A device's one-hop neighbor that is neither its parent nor its children.

3.27 synchronization duration: The time duration of the synchronization procedure in the optional synchronous power saving mode.

3.28 synchronization interval: The time between two consecutive synchronization processes.

3.29 synchronization region: The hop distance to be synchronized by a mesh coordinator or region synchronizer in the optional synchronous power saving mode.

3.30 sync-child: A logical child device that is responsible for maintaining synchronization with its sync-parent device.

3.31 sync-parent: A logical parent device that is responsible for maintaining synchronization with its sync-child devices.

3.32 tree level: The distance, in the number of hops, between a device and the root along the logical tree. The root of the tree always has a tree level of zero.

3.33 wakeup interval: The time period defined for the optional energy saving mode for mesh sublayer. It consists of an active duration and an inactive duration.

4. Acronyms

ABAT	adaptive block addressing table
AD	active duration
ASES	asynchronous energy saving
AO	active order
APDU	application protocol data unit
BAT	block addressing table
BcastID	broadcast identifier
BSID	beacon source identifier
CTA	channel time allocation
DestID	destination identifier
DA	destination address

DEV	device
DEVID	device identifier
Dly_ACK	delayed acknowledgment
DME	device management entity
DO	destination active order
EDEV	end device
ARC	asynchronous energy saving retry counter
EREP	extension reply
EREQ	extension request
FC	frame control
FCSL	frame convergence sublayer
GC	group coordinator
G-JREQ	group join request
G-JREP	group join reply
GM	group member
HDTV	high definition TV
IE	information element
IMM_ACK	immediate acknowledgment
LLC	logical link control
LST	link state table
LQI	link quality indication
MAC	medium access control
MC	mesh coordinator
MDEV	mesh-capable device
MeshID	mesh network identifier
MHME	mesh sublayer management entity
MHPDU	mesh protocol data unit
MHSDU	mesh service data unit

MeshIB	mesh information base
MPNC	mesh capable piconet coordinator
MPNCID	mesh-capable piconet coordinator identifier
MSC	message sequence chart
MTR	multicast transaction record
MTT	multicast transaction table
MDT	mesh discovery table
OUI	organizationally unique identifier
PDU	protocol data unit
PHY	physical layer
PIB	PAN information base
PLME	PHY layer management entity
PN	piconet
PNC	piconet coordinator
PNCID	piconet coordinator identifier
PNID	piconet identifier
RT	router
SA	source address
SAP	service access point
SES	synchronous energy saving
TREEID	tree network identifier
TTL	time to live
WI	wakeup interval
WN	wakeup notification
WO	wakeup order
WPAN	wireless personal area network

5. Low-rate WPAN mesh

5.1 General description

This clause of the recommended practice provides an architectural framework that enables low-power, low-rate WPAN devices to promote interoperable, stable, and scalable wireless mesh topologies. Mesh network topologies, as illustrated in Figure 1, allow devices with IEEE Std 802.15.4-2006³ compatible MAC/PHY to extend the network coverage without increasing the transmission power or the receiver sensitivity. Another key advantage of the mesh network is the enhanced reliability via route redundancy. Mesh networking topology is one of the two topologies IEEE Std. 802.15.4-2006 was designed to support, as described in 5.3 of IEEE Std 802.15.4-2006. However, the tasks of defining and implementing the mesh networking functions are out of the scope of IEEE Std 802.15.4-2006. This recommended practice provides a standard way of doing mesh networking over IEEE Std 802.15.4-2006 within the IEEE standard body.

Supported features for LR-WPAN include the following:

- unicast, multicast, and reliable broadcast mesh data forwarding
- synchronous and asynchronous power saving for mesh devices
- trace route function
- portability of end devices

End-to-end reliability and security are not provided in the mesh sublayer in this version of the recommended practice.

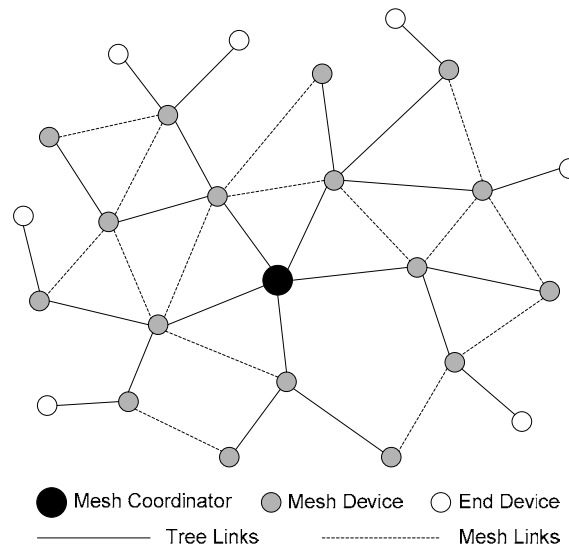


Figure 1—An illustration of mesh network topology

The applications of low rate WPAN mesh networks are numerous; including but not limited to, home and industrial automation and control, security and environmental monitoring, situational awareness and asset

³ Information on references can be found in Clause 2.

tracking, automatic meter reading, personal health monitoring, and etc. Example use cases could be as follows:

- a) A RF remote control that can control all AV devices in all rooms of a residential house without line of sight requirement
- b) Wireless light switches in a commercial building, e.g. a department store, control lights of the entire floor, with the capability of grouping lights in different ways in a dynamic manner and turn them on/off with one push of the button
- c) Smart utility meters form a mesh network, collect meter reading data and send the data to utility companies through gateways

In Clause 5 of this document, the terms *mesh network* and *PAN* are used alternatively to refer to the mesh network since all devices in a mesh network share the same PAN ID. In other words, inter-PAN communication is not supported in the current version of the standard.

The reference model of low rate WPAN mesh is illustrated in Figure 2.

In the data plane, the WPAN Mesh Sublayer resides between the Service Specific Convergence Sublayer (SSCS) of IEEE Std 802.2-1998 Logical Link Control (LLC) and the IEEE Std 802.15.4-2006 Media Access Control (MAC) Sublayer. Mesh Sublayer provides services to the next higher layer via Mesh Service Access Point (Mesh SAP). For implementers to add mesh function to the existing low rate WPAN applications with the least effort, the Mesh SAP is made almost identical to the MCPS SAP.

In the management plane, the mesh sublayer management entity (MHME) resides between the application management entity (AME) and the MAC layer management entity (MLME). MHME also interfaces with the mesh sublayer at the same level. As a reference, the device management entity (DME) that has access and control to all layers is also shown in the figure. However, the specifications of both AME and DME are beyond the scope of this document.

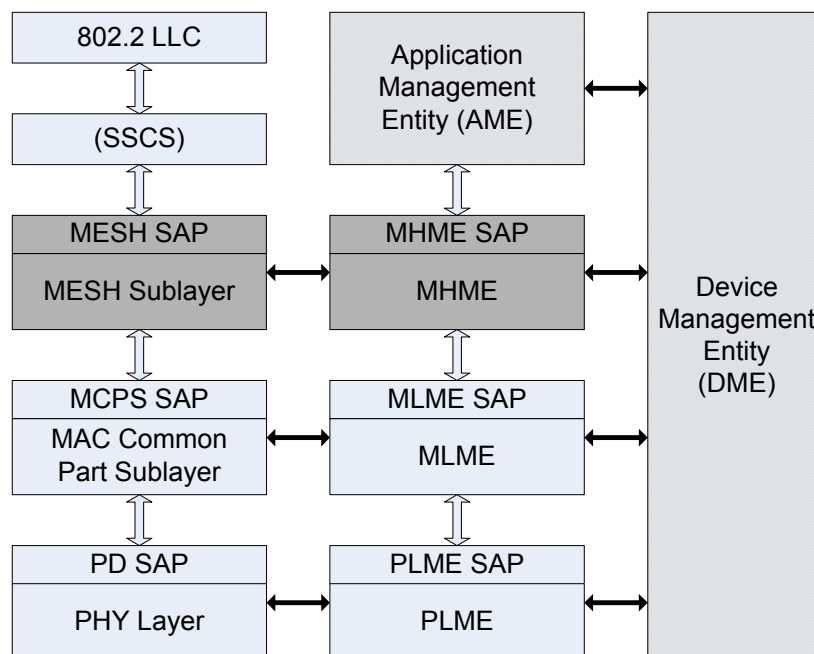


Figure 2—The reference model for low-rate WPAN mesh

5.2 Mesh service description

The mesh sublayer provides two types of services, the mesh data service and the mesh management service, to the next higher layer via two corresponding service access points, the mesh SAP and MHME SAP. There is also an internal interface between the mesh sublayer and the MHME allowing MHME to utilize the mesh sublayer data service. On the other hand, the mesh sublayer may also need to get information from MHME through this internal interface to compose mesh sublayer data frames.

5.2.1 Mesh data service

The function of the mesh sublayer data service is to support the transport of application protocol data units (APDUs) between peer application entities residing at different devices that can be multiple hops away from each other. As described in 5.1, the mesh data service primitives are very similar to the MCPS SAP primitives. The implementers should pay careful attention to the differences between them. Table 1 lists the primitives supported by the mesh SAP.

Table 1—Mesh SAP primitives

MESH SAP primitives	Request	Confirm	Indication
MESH-DATA	5.2.1.1	5.2.1.2	5.2.1.3
MESH-PURGE	5.2.1.4	5.2.1.5	

5.2.1.1 MESH-DATA.request

The MESH-DATA.request primitive requests the transfer of an SPDU (SSCS protocol data unit), from a local SSCS entity to one or more peer SSCS entities. It is very similar to the MCPS-DATA.request primitive at MAC layer.

The semantics of the MESH-DATA.request primitive is as follows:

```
MESH-DATA.request (
    SrcAddrMode,
    DstAddrMode,
    DstAddr,
    mhsduLength,
    mhsdu,
    mhsduHandle,
    AckTransmission,
    McstTransmission,
    BcstTransimission,
    ReliableBcst
)
```

Table 2 specifies the parameters for the MESH-DATA.request primitive.

Table 2—MESH-DATA.request parameters

Name	Type	Valid range	Description
SrcAddrMode	Integer	0x00–0x03	The source addressing mode for this primitive and subsequent MHPDU. This value can take one of the following values: 0x00 = no address (addressing fields omitted). 0x01 = reserved. 0x02 = 16-bit short address. 0x03 = 64-bit extended address.
DstAddrMode	Integer	0x00–0x03	The destination addressing mode for this primitive and subsequent MHPDU. This value can take one of the following values: 0x00 = no address (addressing fields omitted). 0x01 = reserved. 0x02 = 16-bit short address. 0x03 = 64-bit extended address.
DstAddr	Device address	As specified by the DstAddrMode parameter	The device address of the entity, or entities in the case of multicast and broadcast, to which the MHSDU is being transferred.
mhsduLength	Integer	$\leq (aMaxMACSafePayloadSize - meshcMaxMeshHeaderLength)$	The number of octets contained in the MHSDU to be transmitted by the mesh sublayer entity.
mhsdu	Set of octets	–	The set of octets forming the MHSDU to be transmitted by the mesh sublayer entity.
mhsduHandle	Integer	0x00–0xff	The handle associated with the MHSDU to be transmitted by the mesh sublayer entity.
AckTransmission	Boolean	TRUE, FALSE	This field is set to TRUE if an acknowledgement is required from the receiver; otherwise, it is set to FALSE.
McastTransmission	Boolean	TRUE, FALSE	This field is set to TRUE if the data is to be multicast; otherwise, it is set to FALSE.
BcastTransmission	Boolean	TRUE, FALSE	This field is set to TRUE if the data is to be broadcast; otherwise, it is set to FALSE.
ReliableBcast	Boolean	TRUE, FALSE	This field is set to TRUE if reliable broadcast is required; otherwise, it is set to FALSE. It is meaningful only when BcastTransmission is set to TRUE.

The MESH-DATA.request primitive is generated by a local SSCS entity when an SPDU is to be transferred to a peer SSCS entity.

On receipt of the MESH-DATA.request primitive, the mesh sublayer of a device first looks at the value of the DstAddrMode field. If the value indicates a 64-bit extended address is used (0x03), the data may be sent to its one-hop neighbors only. Therefore, the mesh sublayer should compose an MCPS-DATA.request and send it to the MAC layer. In this case, the mesh sublayer simply passes the request from the higher layer without calling the mesh sublayer functions. b_0 of the TxOptions field of the MCPS-DATA.request primitive should reflect the value of AckTransmission; it is set to one when the AckTransmission field is set to TRUE and set to zero when the AckTransmission field is set to FALSE. Fields b_1 and b_2 of the TxOptions of the MCPS-DATA.request primitive should always be set to zero. The rest of the transmit option fields, McastTransmission, BestTransmission, and ReliableBest, are created to indicate to the mesh sublayer the transmission mode of the data. Therefore they are only meaningful to the mesh sublayer and should not be reflected in the MCPS-DATA.request primitive.

If the value of the DstAddrMode field indicates a 16-bit short address is used (0x02) the destination could be either a one-hop neighbor or a device multiple hops away. Therefore, the device should first determine whether the destination is one of its one-hop neighbors by looking at the DstAddr and searching the destination address in its neighbor list as illustrated in Table 46. If it is, the mesh sublayer should compose an MCPS-DATA.request and send it to the MAC layer as described above.

If the destination is not a one-hop neighbor, the mesh sublayer should determine the next hop in the mesh network to route the data message to its final destination. The mesh sublayer should first compose an MHPDU to transmit the SPDU received from SSCS as described in 5.3.2.1. The mesh sublayer should then look up the next hop toward the destination using the method described in 5.5.5 and compose an MCPS-DATA.request with the MAC layer destination address set to the address of the next hop. The MCPS-DATA.request primitive is then sent to the MAC layer.

Because the MHSDU has now been wrapped by the mesh header, the payload to be sent to the MAC layer will be the MHPDU. The mhsduLength should be equal to or less than ($aMaxMACFrameSize - meshcMaxMeshHeaderLength$), where $meshcMaxMeshHeaderLength$ is a mesh constant in Table 41. The values in fields other than those discussed above should be kept unchanged and pass through to the MAC layer.

If the destination is not an one-hop neighbor, the values of all four transmit option fields should be recorded in the corresponding bits of the Transmission Mode field of the Frame Control field in the composed mesh sublayer data frame as illustrated in Figure 5 and transmitted to the next hops so that the transmission mode information can be carried while the data is propagated. Note values of the transmit option fields only have local meanings. For example, when AckTransmission is set to TRUE to indicate an acknowledged transmission, it only means the MAC layer of this device should transmit the data frame using acknowledged transmission and does not mean end-to-end acknowledgement is supported.

If the MHPDU was successfully transmitted and an acknowledgment, if requested, was received, the mesh sublayer will issue the MESH-DATA.confirm primitive with a status of SUCCESS.

If any parameter in the MESH-DATA.request primitive is not supported or is out of range, the mesh sublayer will issue the MESH-DATA.confirm primitive with a status of INVALID_PARAMETER.

5.2.1.2 MESH-DATA.confirm

The MESH-DATA.confirm primitive reports the results of a request to transfer a data SPDU (MHSDU) from a local SSCS entity to one or more peer SSCS entities.

The semantics of the MESH-DATA.confirm primitive is as follows:

```
MESH-DATA.confirm (
    mhsduHandle,
    Status
)
```

Table 3 specifies the parameters for the MESH-DATA.confirm primitive.

Table 3—MESH-DATA.confirm parameters

Name	Type	Valid range	Description
mhsduHandle	Integer	0x00–0xff	The handle associated with the MHSU being confirmed.
Status	Enumeration	SUCCESS, TRANSACTION_OVERFLOW, TRANSACTION_EXPIRED, CHANNEL_ACCESS_FAILURE, NO_ACK, UNAVAILABLE_KEY, FRAME_TOO_LONG, INVALID_PARAMETER	The status of the last MHSU transmission.

The MESH-DATA.confirm primitive is generated by the mesh sublayer in response to an MESH-DATA.request primitive. The MESH-DATA.confirm primitive returns a status of either SUCCESS, indicating that the request to transmit was successful, or an error code of TRANSACTION_OVERFLOW, TRANSACTION_EXPIRED, CHANNEL_ACCESS_FAILURE, UNAVAILABLE_KEY, NO_ACK, FRAME_TOO_LONG, or INVALID_PARAMETER. The reasons for these status values are fully described in 7.1.1.1.3 of IEEE Std 802.15.4-2006.

On receipt of the MESH-DATA.confirm primitive, the SSCS of the initiating device is notified of the result of its request to transmit. If the transmission attempt was successful, the status parameter will be set to SUCCESS. Otherwise, the status parameter will indicate the error.

5.2.1.3 MESH-DATA.indication

The MESH-DATA.indication primitive indicates the transfer of a data SPDU (i.e., MHSU) from the mesh sublayer to the local SSCS entity.

The semantics of the MESH-DATA.indication primitive is as follows:

```
MESH-DATA.indication (
    SrcAddrMode,
    SrcPANId,
    SrcAddr,
    mhsduLength,
    mhsdu
)
```

Table 4 specifies the parameters for the MESH-DATA.indication primitive.

The MESH-DATA.indication primitive is generated by the mesh sublayer and issued to the SSCS upon receiving the corresponding MCPS-DATA.indication from the MAC layer that passes the appropriate message filtering operations.

On receipt of the MESH-DATA.indication primitive, the SSCS is notified of the arrival of data at the device.

Table 4—MESH-DATA.indication parameters

Name	Type	Valid range	Description
SrcAddrMode	Integer	0x00–0x03	The source addressing mode for this primitive corresponding to the received MHPDU. This value can take one of the following values: 0x00 = no address (addressing fields omitted). 0x01 = reserved. 0x02 = 16-bit short address. 0x03 = 64-bit extended address.
SrcPANId	Integer	0x0000–0xffff	The 16-bit PAN identifier of the entity from which the MHSDU was received.
SrcAddr	Device address	As specified by the SrcAddrMode parameter	The individual device address of the entity from which the MHSDU was received.
mhsduLength	Integer	$\leq aMaxMACFrameSize$	The number of octets contained in the MHSDU being indicated by the MESH sublayer entity.
mhsdu	Set of octets	—	The set of octets forming the MHSDU being indicated by the MESH sublayer entity.

5.2.1.4 MESH-PURGE.request

The MESH-PURGE.request primitive allows the next higher layer to purge an MHSDU from the transaction queue.

The semantics of the MESH-PURGE.request primitive is as follows:

MESH-PURGE.request (mhsduHandle)

Table 5 specifies the parameters for the MESH-PURGE.request primitive.

Table 5—MESH-PURGE.request parameters

Name	Type	Valid range	Description
mhsduHandle	Integer	0x00–0xff	The handle of the MHSDU to be purged from the transaction queue.

The MESH-PURGE.request primitive is generated by the next higher layer whenever an MHSDU is to be purged from the transaction queue.

On receipt of the MESH-PURGE.request primitive, the mesh sublayer attempts to find in its transaction queue the MHSDU indicated by the mhsduHandle parameter. If an MHSDU matching the given handle is found, the MHSDU is discarded from the transaction queue, and the mesh sublayer issues the MESH-PURGE.confirm primitive with a status of SUCCESS. If an MHSDU matching the given handle is not found, the mesh sublayer issues the MESH-PURGE.confirm primitive with a status of INVALID_HANDLE.

5.2.1.5 MESH-PURGE.confirm

The MESH-PURGE.confirm primitive allows the mesh sublayer to notify the next higher layer of the success of its request to purge an MHSDU from the transaction queue.

The semantics of the MESH-PURGE.confirm primitive is as follows:

MESH-PURGE.confirm (
 mhsduHandle,
 Status
)

Table 6 specifies the parameters for the MESH-PURGE.confirm primitive.

Table 6—MESH-PURGE.confirm parameters

Name	Type	Valid range	Description
mhsduHandle	Integer	0x00–0xff	The handle of the MHSDU requested to be purged from the transaction queue.
Status	Enumeration	SUCCESS or INVALID_HANDLE	The status of the request to purge an MHSDU from the transaction queue.

The MESH-PURGE.confirm primitive is generated by the mesh sublayer in response to an MESH-PURGE.request primitive. The MESH-PURGE.confirm primitive returns a status of either SUCCESS, indicating that the purge request was successful, or INVALID_HANDLE, indicating an error. The reasons for these status values are fully described in 7.1.1.4.3 of IEEE Std 802.15.4-2006.

On receipt of the MESH-PURGE.confirm primitive, the next higher layer is notified of the result of its request to purge an MHSDU from the transaction queue. If the purge request was successful, the status parameter will be set to SUCCESS. Otherwise, the status parameter will indicate the error.

5.2.1.6 Data service message sequence chart

Figure 3 illustrates the sequence of messages necessary for a successful data transfer between two devices.

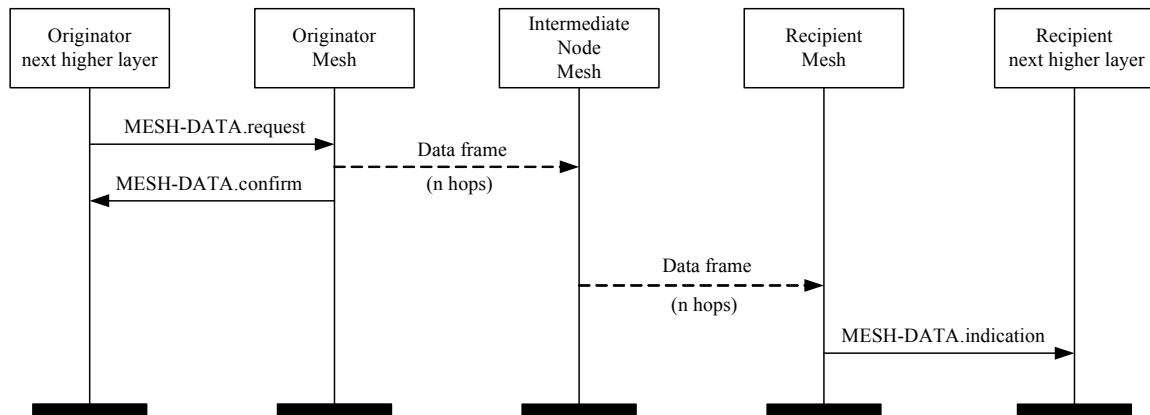


Figure 3—Message sequence chart describing the mesh data service

5.2.2 Mesh management service

The MHME-SAP allows the transport of management commands between SCS and the MHME Table 7 summarizes the primitives supported by the MHME through the MHME-SAP interface. The primitives are discussed in the subclauses referenced in the table.

Table 7—Summary of the primitives accessed through the MHME-SAP

Name	Subclause Number in This Recommended Practice				Mandatory/ Optional
	Request	Indication	Response	Confirm	
MHME-DISCOVER	5.2.2.1			5.2.2.2	M
MHME-START-NETWORK	5.2.2.3			5.2.2.4	M
MHME-START-DEVICE	5.2.2.5			5.2.2.6	M
MHME-JOIN	5.2.2.7	5.2.2.8		5.2.2.9	M
MHME-LEAVE	5.2.2.10	5.2.2.11		5.2.2.12	M
MHME-RESET	5.2.2.13			5.2.2.14	M
MHME-GET	5.2.2.15			5.2.2.16	M
MHME-SET	5.2.2.17			5.2.2.18	M
MHME-START-SYNC	5.2.2.19			5.2.2.20	O

Table 7— Summary of the primitives accessed through the MHME-SAP (continued)

Name	Subclause Number in This Recommended Practice				Mandatory/ Optional
	Request	Indication	Response	Confirm	
MHME-TRACE-ROUTE	5.2.2.21	5.2.2.22		5.2.2.23	O
MHME-MULTICAST-JOIN	5.2.2.24			5.2.2.25	M
MHME-MULTICAST-LEAVE	5.2.2.26			5.2.2.27	M

5.2.2.1 MHME-DISCOVER.request

This primitive allows the next higher layer to request the mesh sublayer to discover mesh networks currently operating within the neighborhood.

The semantics of the primitive is as follows:

MHME-DISCOVER.request (
 ScanChannels,
 ScanDuration,
 ChannelPage,
 ReportCriteria
)

Table 8 specifies the parameters of the MHME-DISCOVER.request primitive.

Table 8—MHME-DISCOVER.request parameters

Name	Type	Valid range	Description
ScanChannels	Bit field	27-bit field	The 27 bits ($b_0, b_1, \dots, b_{26}$) indicate which channels are to be scanned (1 = scan, 0 = do not scan) for each of the 27 channels supported by the ChannelPage parameter as defined in IEEE Std 802.15.4-2006.
ScanDuration	Integer	0–14	A value used to calculate the length of time to spend scanning each channel: The time spent scanning each channel is ($aBaseSuperframeDuration * (2^n + 1)$) symbols, where n is the value of the ScanDuration parameter. For more information on MAC sublayer scanning as defined in IEEE Std 802.15.4-2006.
ChannelPage	Integer	0–31	The channel page on which to perform the scan.

Table 8— MHME-DISCOVER.request parameters (continued)

Name	Type	Valid range	Description
ReportCriteria	Enumeration	LINK_QUALITY, TREE_LEVEL	The field indicates which criterion is used to select the best neighbor to be reported to the next higher layer. LINK_QUALITY = The neighbor with highest link quality is reported. TREE_LEVEL = The neighbor with lowest tree level is reported. Combination of multiple criteria is allowed despite the algorithm is out of scope of this recommended practice.

This primitive is generated by the next higher layer and issued to the mesh sublayer to request the discovery of networks operating within the device's operating area.

On receipt of this primitive, the mesh sublayer will attempt to discover networks operating within the device's operating area by performing an active scan over the channels specified in the ScanChannels argument, and the channel pages specified in the ChannelPage argument, for the period specified in the ScanDuration parameter. The scan is performed by means of the MLME-SCAN.request primitive.

On receipt of the MLME-SCAN.confirm primitive, a device should first record the neighbor devices from which it received the signals in the mesh discovery table (MDT), if the PANDescriptorList of the MLME-SCAN.confirm is not empty. Each beacon found during a scan will create an entry, PANDescriptor, in the PANDescriptorList at the MAC layer and a corresponding entry, MeshDescriptor, in the MDT at the mesh sublayer. The elements of a MeshDescriptor are not exactly the same as that of a PANDescriptor. Elements of MeshDescriptor are illustrated in Table 9. Note the MDT is a temporary table and its content can be cleared after the device has successfully started or joined a mesh network. The mesh sublayer information in beacon payload is defined in Figure 37.

Table 9—Elements of MESHDescriptor

Name	Type	Valid range	Description
PANId	Integer	0x0000–0xffff	The 16-bit PAN identifier of the network discovered.
Address	Integer	0x0000–0xffffd	The 16-bit short address of the beacon sender.
LogicalChannel	Integer	Selected from the available logical channels supported by the PHY as defined in IEEE Std 802.15.4-2006.	The current logical channel occupied by the network.
ChannelPage	Integer	Selected from the available channel pages supported by the PHY.	The current channel page occupied by the network.

Table 9— Elements of MESHDescriptor (continued)

Name	Type	Valid range	Description
MeshVersion	Integer	0x00–0x0f	The version of the mesh sublayer protocol in use in the discovered network.
BeaconOrder	Integer	0x0f	This specifies how often the MAC sublayer beacon is to be transmitted by a given device on the network. For this version of the recommended practice, the value is always set to 0x0f indicating no periodic beacons are transmitted. For a discussion of MAC sublayer beacon order, please refer to IEEE Std 802.15.4-2006.
SuperframeOrder	Integer	0x0f	For beacon-oriented networks, that is, beacon order < 15, this specifies the length of the active period of the superframe. For this version of the recommended practice, the value is always set to 0x0f indicating no periodic beacons are transmitted. For a discussion of MAC sublayer superframe order, please refer to IEEE Std 802.15.4-2006.
LinkQuality	Integer	0x00–0xff	The LQI value at which the network beacon was received. Lower values represent lower LQI.
TreeLevel	Integer	0x00–0xff	The tree level of the beacon sender.
AcceptMeshDevice	Boolean	TRUE or FALSE	This field indicates whether the mesh network accepts new mesh devices. A value of TRUE indicates the device is capable of accepting join requests from other mesh devices; the value should be set to FALSE otherwise.
AcceptEndDevice	Boolean	TRUE or FALSE	This field indicates whether the mesh network accepts new end devices. A value of TRUE indicates the device is capable of accepting join requests from end devices; the value should be set to FALSE otherwise.
SyncEnergySaving	Boolean	TRUE or FALSE	This field indicates whether the optional synchronous energy saving feature is supported. A value of TRUE indicates the device is capable of supporting synchronous energy saving; the value should be set to FALSE otherwise.

Table 9— Elements of MESHDescriptor (continued)

Name	Type	Valid range	Description
AsyncEnergySaving	Boolean	TRUE or FALSE	This field indicates whether the optional asynchronous energy saving feature is supported. A value of TRUE indicates the device is capable of supporting asynchronous energy saving; the value should be set to FALSE otherwise.

The mesh sublayer should then issue the MHME-DISCOVER.confirm primitive containing the information about the discovered networks with a Status parameter value equal to that returned with the MLME-SCAN.confirm.

5.2.2.2 MHME-DISCOVER.confirm

This primitive reports to its next higher layer the result of a network discovery operation.

The semantics of the primitive is as follows:

```
MHME-DISCOVER.confirm    (  
    Status,  
    NetworkCount,  
    MeshDescriptorList  
)
```

Table 10 specifies the parameters of the MHME-DISCOVER.confirm primitive.

Table 10—MHME-DISCOVER.confirm parameters

Name	Type	Valid range	Description
Status	Status	Any Status value returned with the MLME-SCAN.confirm primitive	Defined in 7.1.11.2 of IEEE Std 802.15.4-2006.
NetworkCount	Integer	0x00–0xff	Gives the number of networks discovered by the search.
MeshDescriptorList	List of mesh network descriptors	The list contains the number of elements given by the NetworkCount parameter	A list of descriptors, one for each of the mesh network discovered.

This primitive is generated by the MHME and issued to its next higher layer on completion of the discovery task initiated by an MHME-DISCOVERY.request primitive.

Table 9 gives a detailed account of the contents of a MeshDescriptor from the MeshDescriptorList parameter. However, not all the entries in the MDT are reported to the next higher layer through the MHME-DISCOVER.confirm primitive; among the beacon senders with the same PANId, only the best one with regard to the ReportCriteria should be reported to the next higher layer. Determining the report criteria is an implementation issue. This standard suggests consider either LinkQuality or TreeLevel or the combination of the two of the discovered devices.

On receipt of this primitive, the next higher layer is notified of the results of a network search.

5.2.2.3 MHME-START-NETWORK.request

This primitive allows the next higher layer to request that the device start a new mesh network with itself as the MC.

The semantics of the primitive is as follows:

MHME-START-NETWORK.request (ScanChannels, ScanDuration, ChannelPage, BeaconOrder, SuperframeOrder)

Table 11 specifies the parameters for the MHME-START-NETWORK.request primitive.

Table 11 —MHME-START-NETWORK.request parameters

Name	Type	Valid range	Description
ScanChannels	Bit field	27-bit field	The 27 bits ($b_0, b_1, \dots, b_{26}$) indicate which channels are to be scanned (1 = scan, 0 = do not scan) for each of the 27 channels supported by the ChannelPage parameter as defined in IEEE Std 802.15.4-2006.
ScanDuration	Integer	0–14	A value used to calculate the length of time to spend scanning each channel: The time spent scanning each channel is ($aBaseSuperframeDuration * (2^n + 1)$) symbols, where n is the value of the ScanDuration parameter. For more information on MAC sublayer scanning, please refer to 7.1.11.1 of IEEE Std 802.15.4-2006.
ChannelPage	Integer	0–31	The channel page on which to perform the scan.
BeaconOrder	Integer	0x0f	The beacon order of the network that the higher layers wish to form. For this version of the recommended practice, the value is always set to 0x0f indicating no periodic beacons are transmitted.
SuperframeOrder	Integer	0x0f	The superframe order of the network that the higher layers wish to form. For this version of the recommended practice, the value is always set to 0x0f indicating no periodic beacons are transmitted.

On receipt of this primitive by a device that is not capable of being a mesh coordinator (MC) (by checking the mesh sublayer constant, *meshcCoordinatorCapability*), the MHME issues the MHME-START-NETWORK.confirm primitive with the Status parameter set to INVALID_REQUEST.

If only a single channel is provided in the higher layer request, the MHME does not need to request an energy scan prior to starting the network. An active scan is still conducted to ensure a PAN identifier is selected that does not conflict with that of any network known to be operating.

5.2.2.4 MHME-START-NETWORK.confirm

The semantics of the primitive is as follows:

Table 12 specifies the parameters for the MHME-START-NETWORK.confirm primitive.

Table 12—MHME-START-NETWORK.confirm parameters

Name	Type	Valid range	Description
Status	Status	INVALID_REQUEST, STARTUP_FAILURE or any status value returned from the MLME-START.confirm primitive	The result of the attempt to initialize a mesh coordinator

This primitive is generated by the MHME and issued to its next higher layer in response to an MHME-START-NETWORK.request primitive. The primitive returns a status value of INVALID_REQUEST, STARTUP_FAILURE or any status value returned from the MLME-START.confirm primitive. Conditions under which these values may be returned are described in 5.2.2.3.

On receipt of this primitive, the next higher layer is notified of the result of its request to initialize the device as a mesh coordinator. If the MHME has been successful, the status parameter will be set to SUCCESS. Otherwise, the status parameter indicates the error.

5.2.2.5 MHME-START-DEVICE.request

This primitive allows the next higher layer of a mesh device to initiate the activities expected of a mesh device, including the routing of data frames and the accepting of requests to join the network from other devices.

The semantics of the primitive is as follows:

```
MHME-START-DEVICE.request (
    BeaconOrder,
    SuperframeOrder
)
```

Table 13 specifies the parameters for MHME-START-DEVICE.request.

Table 13—MHME-START-DEVICE.request parameters

Name	Type	Valid range	Description
BeaconOrder	Integer	0x0f	The beacon order of the network that the higher layers wish to form. For this version of the recommended practice, the value is always set to 0x0f indicating no periodic beacons are transmitted.
SuperframeOrder	Integer	0x0f	The superframe order of the network that the higher layers wish to form. For this version of the recommended practice, the value is always set to 0x0f indicating no periodic beacons are transmitted.

This primitive is generated by the next higher layer of a new device and issued to its MHME to request the initialization of itself as a mesh device.

On receipt of this primitive by a device that is not already joined to a mesh network as a mesh device, the MHME issues the MHME-START-DEVICE.confirm primitive with the Status parameter set to INVALID REQUEST.

On receipt of this primitive by the MHME of a device that is joined to a mesh network as a mesh device, the MHME issues the MLME-START.request primitive to the MAC sublayer. The BeaconOrder and SuperframeOrder parameters of the MLME-START.request primitive will have the values given by the corresponding parameters of the MHME-START-DEVICE.request. The PANCoordinator parameter in the MLME-START.request primitive is set to FALSE indicating this device is not the PAN coordinator of the network. The CoordRealignment parameter in the MLME-START.request primitive is set to FALSE if the primitive is issued to start as a mesh device for the first time. The CoordRealignment parameter is set to TRUE thereafter if the primitive is issued to change any of the PAN configuration attributes.

On receipt of the associated MLME-START.confirm primitive, the MHME issues the MHME-START-DEVICE.confirm primitive to the next higher layer with the status returned from the MLME-START.confirm primitive. If, and only if, the status returned from the MLME-START.confirm primitive is SUCCESS, the device may then begin to engage in the routing of data frames and the accepting of requests to join the network from other devices. Otherwise, the device shall not engage in these activities.

5.2.2.6 MHME-START-DEVICE.confirm

This primitive reports the results of the request to initialize a mesh device.

The semantics of the primitive is as follows:

[illegible]

Table 14 specifies the parameters for MHME-START-DEVICE.confirm.

Table 14—MHME-START-DEVICE.confirm parameter

Name	Type	Valid range	Description
Status	Status	INVALID_REQUEST or any status value returned from the MLME-START.confirm primitive	The result of the attempt to initialize a mesh device

This primitive is generated by the MHME and issued to its next higher layer in response to an MHME-START-MESH-DEVICE.request primitive. This primitive return a status value of INVALID_REQUEST or any status value returned from the MLME-START.confirm primitive. Conditions under which these values may be returned are described in 5.2.2.5.

On receipt of this primitive, the next higher layer is notified of the results of its request to initialize a mesh device. If the MHME has been successful, the status parameter will be set to SUCCESS. Otherwise, the status parameter indicates the error.

5.2.2.7 MHME-JOIN.request

This primitive allows the next higher layer of a device to request to join or rejoin a network.

The semantics of the primitive is as follows:

```
MHME-JOIN.request (
    DirectJoin,
    ParentDevAddr,
    PANId,
    RejoinNetwork,
    JoinAsMeshDevice,
    ScanChannels,
    ScanDuration,
    ChannelPage,
    CapabilityInformation
)
```

Table 15—MHME-JOIN.request parameters

Name	Type	Valid range	Description
DirectJoin	Boolean	TRUE or FALSE	The value is set to TRUE if direct joining is chosen; otherwise, its value is FALSE.
ParentDevAddr	Integer	0x0000–0xffff	The 16-bit short address of the parent device to join. This field will be read only when the DirectJoin parameter has a value equal to TRUE.
PANId	Integer	0x0000–0xffff	The 16-bit PAN identifier of the network to join. This field will be read only when the DirectJoin parameter has a value equal to TRUE.
RejoinNetwork	Integer	0x00–0x01	This parameter controls the method of joining the network. The parameter is 0x00 if the device is requesting to join a network through association. The parameter is 0x01 if the device is joining the network using the mesh sublayer rejoining procedure. This field will be read only when the DirectJoin parameter has a value equal to TRUE.
JoinAsMeshDevice	Boolean	TRUE or FALSE	The parameter is set to TRUE if the device is going to function as a mesh device; it is set to FALSE if the device is going to function as an end device.

Table 15—MHME-JOIN.request parameters (continued)

Name	Type	Valid range	Description
ScanChannels	Bit field	27-bit field	The 27 bits ($b_0, b_1, \dots, b_{26}$) indicate which channels are to be scanned (1 = scan, 0 = do not scan) for each of the 27 channels supported by the ChannelPage parameter. This field will be read only when the DirectJoin parameter has a value equal to FALSE.
ScanDuration	Integer	0–14	A value used to calculate the length of time to spend scanning each channel: The time spent scanning each channel is ($aBaseSuperframeDuration * (2^n + 1)$) symbols, where n is the value of the ScanDuration parameter. For more information on MAC sublayer scanning, please refer to IEEE Std 802.15.4-2006. This field will be read only when the DirectJoin parameter has a value equal to FALSE.
ChannelPage	Integer	0–31	The channel page on which to perform the scan. This field will be read only when the DirectJoin parameter has a value equal to FALSE.
CapabilityInformation	Boolean fields	As described in Table 16	The operating capabilities of the device being directly joined.

Table 16—Capability Information bit-fields

Name	Type	Valid range	Description
DeviceType	Boolean	TRUE, FALSE	This field is set to TRUE if the joining device is a mesh device. Otherwise, it is set to FALSE.
PowerSource	Boolean	TRUE, FALSE	This field is set to TRUE if it is mains-powered. Otherwise, it is set to FALSE.
ReceiverOnWhenIdle	Boolean	TRUE, FALSE	This field is set to TRUE if the receiver is enabled when the device is idle. Otherwise, it is set to FALSE.
AllocateAddress	Boolean	TRUE, FALSE	This field is always set to TRUE in the implementations of this recommended practice, indicating that the joining device should be issued a 16-bit network address.

The next higher layer of a device generates this primitive to request to:

- Join a new network
- Locate and rejoin a network after being disconnected

On receipt of this primitive by a device that is currently joined to a network and with the RejoinNetwork parameter equal to 0x00, the MHME issues an MHME-JOIN.confirm primitive with the status parameter set to INVALID_REQUEST.

On receipt of this primitive by a device that is not currently joined to a network and with the RejoinNetwork parameter equal to 0x00, the device attempts to join the network. If the DirectJoin parameter is set to TRUE, the device attempts to join the parent device whose address is specified by the ParentDevAddr field. The MHME issues an MLME-ASSOCIATE.request with its CoordAddress parameter set to ParentDevAddr if the following occur:

- the parent device belongs to the network identified by the PANId, and
- the parent device is open to join requests and is advertising capacity of the correct device type.

If the DirectJoin parameter is set to FALSE, the device attempts to join the network through a neighbor device found by the discovery procedure. The MHME issues an MLME-ASSOCIATE.request with its CoordAddress parameter set to the address of a mesh device for which the following conditions are true:

- The mesh device belongs to the network identified by the PANId
- The mesh device is open to join requests and is advertising capacity of the correct device type
- The link quality for frames received from this device is such that a link quality of at least 128 is produced (the link quality ranges from 0, the lowest, to 255, the highest)

If a device exists in the mesh discovery table as illustrated in Table 9 for which these conditions are true, the LogicalChannel parameter of the MLME-ASSOCIATE.request primitive is set to that found in the MDT, corresponding to the coordinator address of the potential parent. The fields of the CapabilityInformation parameter should have the values shown in Table 16 and the capability information assembled here should be stored as the value of the meshCapabilityInformation MeshIB attribute. If more than one device meets these requirements, then the joining device may select the parent with the smallest tree level or/and highest link quality. How to weight the tree level and link quality is not defined in this document. The implementers are given the flexibility to choose the way that meets their application requirements best.

If no device exists in the mesh discovery table for which the conditions are true, the mesh sublayer should issue an MHME-JOIN.confirm with the Status parameter set to NOT_PERMITTED. Otherwise, the MHME issues the MHME-JOIN.confirm with the Status parameter set to the status parameter value returned from the MLME-ASSOCIATE.confirm primitive.

If the RejoinNetwork parameter is 0x00 and the JoinAsMeshDevice parameter is set to TRUE, the device will function as a mesh device in the network. If the JoinAsMeshDevice parameter is FALSE, then it will join as an end device and not participate in routing.

If a device receives this primitive and the RejoinNetwork parameter is equal to 0x01 then it should join the network using the mesh sublayer rejoin process as described in 5.5.11.2.

5.2.2.8 MHME-JOIN.indication

This primitive allows the next higher layer of a mesh coordinator or a mesh device to be notified when a new device has successfully joined its network by association or rejoined using mesh sublayer rejoin procedure.

The semantics of the primitive is as follows:

```
MHME-JOIN.indication (
    NetworkAddress,
    ExtendedAddress,
    CapabilityInformation,
    RejoinNetwork
)
```

Table 17 specifies the parameters for the MHME-JOIN.indication primitive.

Table 17—MHME-JOIN.indication parameters

Name	Type	Valid range	Description
NetworkAddress	Network address	0x0001–0xffffd, or 0xffffe	When a short network address of an entity has been assigned, this will be the short address that has been added to the network. Otherwise, the value of this parameter will equal to 0xffffe indicating the short address has not been assigned and the device can only be reached by its 64-bit extended address.
ExtendedAddress	64-bit IEEE address	Any 64-bit, IEEE address	The 64-bit IEEE address of an entity that has been added to the network.
CapabilityInformation	Bit field	As defined in IEEE Std 802.15.4-2006.	Specifies the operational capabilities of the joining device.
RejoinNetwork	Integer	0x00–0x02	The RejoinNetwork parameter indicating the method used to join the network. The parameter is 0x00 of the device joined through association. The parameter is 0x01 if the device joined through mesh sublayer rejoin procedure.

This primitive is generated by the MHME of a mesh coordinator or mesh device and is issued to its next higher layer on successfully adding a new device to the network using the MAC association, or on allowing a device to rejoin the network using the rejoining procedure. Note when the NetworkAddress parameter has a value equal to 0xffffe, it means the network is in the initialization state and the short address has not been assigned to the joining device yet. This device should later assign the addresses to all its child devices through the address assignment command frame as defined in Figure 11.

On receipt of this primitive, the next higher layer of a mesh coordinator or mesh device is notified that a new device has joined its network.

5.2.2.9 MHME-JOIN.confirm

This primitive allows the next higher layer to be notified of the result of its request to join a network.

The semantics of the primitive is as follows:

```
MHME-JOIN.confirm (
    Status,
    NetworkAddress,
    PANId,
    ChannelPage,
    ActiveChannel
)
```

Table 18 specifies the parameters for the MHME-JOIN.confirm primitive.

Table 18—MHME-JOIN.confirm parameters

Name	Type	Valid range	Description
Status	Status	INVALID_REQUEST, NOT_PERMITTED, NO_NETWORKS or any status value returned from the MLME-ASSOCIATE.confirm primitive or the MLME-SCAN.confirm primitive.	The status of the corresponding request.
NetworkAddress	Integer	0x0001–0xffff	The 16-bit network address that was allocated to this device. This parameter will be equal to 0xffff if the join attempt was unsuccessful.
PANId	Integer	0x0000–0xfffe	The 16-bit PAN identifier of the network of which the device is now a member.
ChannelPage	Integer	0–31	The channel page on which the ActiveChannel was found.
ActiveChannel	Integer	The number of any channel supported by the PHY as defined in IEEE Std 802.15.4-2006.	The value of phyCurrentChannel parameter which is equal to the current channel of the network that has been joined.

This primitive is generated by the initiating MHME and issued to its next higher layer in response to an MHME-JOIN.request primitive.

If the request was not successful, this primitive should be sent to the next higher layer as soon as the status of the join request is known. The status parameter should indicate an error code of `INVALID_REQUEST`, `NOT_PERMITTED`, `NO_NETWORKS` or any status value returned from either the `MLME-ASSOCIATE.confirm` primitive or the `MLME-SCAN.confirm` primitive. The reasons for these status values are fully described in 5.2.2.8.

If the request was successful, this primitive is not sent to the next higher layer until a short address is successfully assigned by receiving an address assignment command frame from the parent device. Whether or not the joining device can obtain a short address immediately by using this join request depends on the operating states (initialization or operation) of its parent device. In the case its parent device is in the operation state, this device can get a short address from the parent device immediately. Otherwise, this device is just associated with the parent device using its 64-bit extended address. Devices in this status should use their 64-bit extended addresses to report their child numbers to parent devices. The short address will be assigned after the parent device gets the address block assignment from its own parent. Therefore, the mesh sublayer should hold the confirmation message until the short address is assigned from the parent device. Note that the next higher layer is not capable of transmitting any data until the confirmation message is received.

On receipt of this primitive, the next higher layer of the initiating device is notified of the result of its request to join a network using the MAC sublayer association procedure, or to rejoin a network once it has temporarily left.

5.2.2.10 MHME-LEAVE.request

This primitive allows the next higher layer to request that it or a child device leaves the network.

The semantics of the primitive is as follows:

```
MHME-LEAVE.request (
    DeviceAddress,
    RemoveSelf,
    RemoveChildren
)
```

Table 19 specifies the parameters for the MHME-LEAVE.request primitive.

Table 19—MHME-LEAVE.request parameters

Name	Type	Valid range	Description
DeviceAddress	Device address	Any 64-bit, IEEE address	The 64-bit IEEE address of a child device to be removed from the network.
RemoveSelf	Boolean	TRUE, FALSE	This parameter has a value of TRUE if the device is asked to remove itself from the network. Otherwise it has a value of FALSE.
RemoveChildren	Boolean	TRUE, FALSE	This parameter has a value of TRUE if the device being asked to leave the network is also being asked to remove its child devices, if any. Otherwise it has a value of FALSE.

The next higher layer of a device generates this primitive to request to leave the network. The next higher layer of a mesh coordinator or mesh device may also generate this primitive to remove a device from the network.

On receipt of this primitive by the MHME of a device that is not currently joined to a network, the MHME issues the MHME-LEAVE.confirm primitive with a status of INVALID_REQUEST. On receipt of this primitive by the MHME of a device that is currently joined to a network, with the RemoveSelf parameter set to TRUE and the RemoveChildren parameter set to FALSE, the MHME will remove itself from the network as described in 5.5.7.1. Basically, the MHME should send out a hello command frame to its neighbors to indicate its leaving and clear all of its references to the mesh network. It should then issue an MLME-RESET.request primitive to the MAC sublayer. If the MHME-LEAVE.request primitive is received with the RemoveSelf parameter set to TRUE and the RemoveChildren parameter equal to TRUE, then the MHME should first attempt to remove the device's children before removing itself from the network.

On receipt of this primitive by a mesh coordinator or mesh device and with the RemoveSelf parameter set to FALSE, the MHME determines whether the specified device is a child device. If the requested device does not exist, the MHME issues the MHME-LEAVE.confirm primitive with a status of UNKNOWN_CHILD_DEVICE. If the requested device exists, the MHME will attempt to remove it from the network following the procedure described in 5.5.7.2. Basically the MHME should first send a leave command to the requested device and the device will remove itself following the procedure described in 5.5.7.1. If the RemoveChildren parameter is equal to TRUE then the device will be requested to remove its children as well. Upon receiving the acknowledgement of the leave command, the MHME will issue the MHME-LEAVE.confirm primitive with the DeviceAddress equal to the 64-bit IEEE address of the removed device and the Status parameter equal to the status delivered by the MCPS-DATA.confirm primitive.

5.2.2.11 MHME-LEAVE.indication

This primitive allows the next higher layer of a device to be notified if that device has left the network or if a neighboring device has left the network.

The semantics of the primitive is as follows:

MHME-LEAVE.indication (DeviceAddress)

Table 20 specifies the parameters for the MHME-LEAVE.indication primitive.

Table 20—MHME-LEAVE.indication parameters

Name	Type	Valid range	Description
DeviceAddress	64-bit IEEE address	Any 64-bit, IEEE address	The 64-bit IEEE address of an entity that has removed itself from the network

This primitive is generated by the MHME of a mesh device and issued to its next higher layer on the receipt of a leave command asking it to leave the network. It is also generated by the MHME of a mesh device and issued to its next higher layer to indicate that a neighbor has left the network upon receiving a hello command frame with the leaving network flag set to one.

On receipt of this primitive, the next higher layer of a mesh coordinator or mesh device is notified that a device, formerly on the network, has left. The primitive can also indicate that the next higher layer of a mesh device or end device is informed that it has been removed from the network by its associated mesh device or mesh coordinator.

5.2.2.12 MHME-LEAVE.confirm

This primitive allows the next higher layer to be notified of the results of its request for itself or another device to leave the network.

The semantics of the primitive is as follows:

```
MHME-LEAVE.confirm (
    Status,
    DeviceAddress
)
```

Table 21 specifies the parameters for the MHME-LEAVE.confirm primitive.

Table 21 —MHME-LEAVE.confirm parameters

Name	Type	Valid range	Description
Status	Status	SUCCESS, INVALID_REQUEST, UNKNOWN_CHILD_DEVICE or any status returned by the MCPS-DATA.confirm primitive	The status of the corresponding request
DeviceAddress	64-bit IEEE address	Any 64-bit, IEEE address	The 64-bit IEEE address in the request to which this is a confirmation or null if the device requested to remove itself from the network

This primitive is generated by the initiating MHME and issued to its next higher layer in response to an MHME-LEAVE.request primitive. If the request was successful, the status parameter indicates a successful leave attempt. Otherwise, the status parameter indicates an error code of INVALID_REQUEST, UNKNOWN_CHILD_DEVICE or a status delivered by the MCPS-DATA.confirm primitive. The reasons for these status values are fully described in 5.2.2.10.

On receipt of this primitive, the next higher layer of the initiating device is notified of the results of its request for itself or a child device to leave the network.

5.2.2.13 MHME-RESET.request

This primitive allows the next higher layer to request that the mesh sublayer performs a reset operation.

The semantics of the primitive is as follows:

MHME-RESET.request ()

This primitive has no parameters.

This primitive is generated by the next higher layer and issued to its MHME to request the reset of the mesh sublayer to its initial condition.

On receipt of this primitive, the MHME first checks whether it is connected to any networks. If it is connected a network, the MHME should broadcasts a hello command frame to indicate its leaving to its neighbors up to *meshTTLofHello* hops. If it is not connected to any networks, the MHME should skip the checking and go directly to issue the MLME-RESET.request primitive to the MAC sublayer with the SetDefaultPIB parameter set to TRUE. On receipt of the corresponding MLME-RESET.confirm primitive, the mesh sublayer resets it by clearing all internal variables, mesh discovery table entries and by setting all MeshIB attributes to their default values. Once the mesh sublayer is reset, the MHME issues the MHME-RESET.confirm with the Status parameter set to SUCCESS if the MAC sublayer was successfully reset.

If this primitive is issued to the MHME of a device that is currently joined to a network, any required leave attempts using the MHME-LEAVE.request primitive should be made a priority at the discretion of the next higher layer.

5.2.2.14 MHME-RESET.confirm

This primitive allows the next higher layer to be notified of the results of its request to reset the mesh sublayer.

The semantics of the primitive is as follows:

MHME-RESET.confirm (Status)

Table 22 specifies the parameters for this primitive.

Table 22—MHME-RESET.confirm parameter

Name	Type	Valid range	Description
Status	Status	Any status value returned from the MLME-RESET.confirm primitive as defined in IEEE Std 802.15.4-2006.	The result of the reset operation

This primitive is generated by the MHME and issued to its next higher layer in response to an MHME-RESET.request primitive. If the request was successful, the Status parameter will have a value of SUCCESS. Otherwise, the status parameter will have an error code indicating the reason of failure. The reasons for these status values are fully described in 5.2.2.13.

5.2.2.15 MHME-GET.request

This primitive allows the next higher layer to read the value of an attribute from the MeshIB, which is defined in Table 42.

The semantics of the primitive is as follows:

```
MHME-GET.request    (  
                      MIBAttribute  
                      )
```

Table 23 specifies the parameters for this primitive.

Table 23—MHME-GET.request parameter

Name	Type	Valid range	Description
MeshIB Attribute	Integer	As described in Table 42	The identifier of the MeshIB attribute to read

On receipt of the MHME-GET.request primitive, the MHME checks to see if the MeshIB Attribute is a MeshIB attribute or a MAC/PHY PIB attribute. If the requested attribute is a MeshIB attribute, the MHME attempts to retrieve the requested MeshIB attribute from its database. If the identifier of the MeshIB attribute is not found in the database, the MHME will issue the MHME-GET.confirm primitive with a status of `UNSUPPORTED_ATTRIBUTE`. If the requested MeshIB attribute is successfully retrieved, the MHME will issue the MHME-GET.confirm primitive with a status of `SUCCESS`.

If the requested attribute is a MAC/PHY PIB attribute, the request is passed to the MAC by issuing the MLME-GET.request primitive. The MHME will issue the MHME-GET.confirm primitive to the next higher layer with the status parameter resulting from the translation and the MIBAttribute and MIBAttributeValue parameters equal to those returned by the MLME primitive.

5.2.2.16 MHME-GET.confirm

This primitive reports the results of an attempt to read the value of an attribute from the MIB.

The semantics of the primitive is as follows:

```
MHME-GET.confirm    (  
                      Status,  
                      MIBAttribute,  
                      MIBAttributeLength,  
                      MIBAttributeValue  
                      )
```

Table 24 specifies the parameters for this primitive.

Table 24—MHME-GET.confirm parameters

Name	Type	Valid range	Description
Status	Enumeration	SUCCESS, UNSUPPORTED_ATTRIBUTE	The results of the request to read an MeshIB attribute value.
MIBAttribute	Integer	As described in Table 42	The identifier of the MeshIB attribute that was read.
MIBAttributeLength	Integer	0x0000–0xffff	The length, in octets, of the attribute value being returned.
MIBAttributeValue	Various	Attribute specific, as defined in Table 42	The value of the MeshIB attribute that was read.

This primitive is generated by the MHME and issued to its next higher layer in response to an MHME-GET.request primitive. This primitive returns either a status of SUCCESS, indicating that the request to read a MeshIB attribute was successful, or an error code of UNSUPPORTED_ATTRIBUTE. The reasons for these status values are fully described in 5.2.2.15.

On receipt of this primitive, the next higher layer is notified of the results of its request to read a MeshIB attribute.

5.2.2.17 MHME-SET.request

This primitive allows the next higher layer to write the value of an attribute into the MIB.

The semantics of the primitive is as follows:

```
MHME-SET.request (
    MIBAttribute,
    MIBAttributeLength,
    MIBAttributeValue
)
```

Table 25 specifies the parameters for this primitive.

Table 25—MHME-SET.request parameters

Name	Type	Valid range	Description
MIBAttribute	Integer	As described in Table 42	The identifier of the MeshIB attribute to be written.
MIBAttributeLength	Integer	0x0000—0xffff	The length, in octets, of the attribute value being set.
MIBAttributeValue	Various	Attribute specific, as defined in Table 42	The value of the MeshIB attribute that should be written.

This primitive is to be generated by the next higher layer and issued to its MHME to write the value of an attribute in the MIB.

On receipt of the MHME-SET.request primitive, the MHME checks to see if the MIBAttribute is a MeshIB attribute or MAC/PHY PIB attribute. If the requested attribute is a MeshIB attribute, the MHME attempts to write the given value to the indicated MeshIB attribute in its database. If the MIBAttribute parameter specifies an attribute that is a read-only attribute, the MHME will issue the MHME-SET.confirm primitive with a status of READ_ONLY. If the MIBAttribute parameter specifies an attribute that is not found in the database, the MHME will issue the MHME-SET.confirm primitive with a status of UNSUPPORTED_ATTRIBUTE. If the MIBAttributeValue parameter specifies a value that is out of the valid range for the given attribute, the MHME will issue the MHME-SET.confirm primitive with a status of INVALID_PARAMETER. If the requested MeshIB attribute is successfully written, the MHME will issue the MHME-SET.confirm primitive with a status of SUCCESS.

If the requested attribute is a MAC/PHY PIB attribute, the request is passed to the MAC by issuing the MLME-SET.request primitive. The MHME will issue the MHME-SET.confirm primitive to the next higher layer with the status parameter resulting from the translation and the MIBAttribute parameter equal to that returned by the MLME primitive.

5.2.2.18 MHME-SET.confirm

The primitive reports the results of an attempt to write a value to a MeshIB attribute.

The semantics of the primitive is as follows:

```
MHME-SET.confirm    (
                      Status,
                      MIBAttribute
                      )
```

Table 26—MHME-SET.confirm parameters

Name	Type	Valid range	Description
Status	Enumeration	SUCCESS, INVALID_PARAMETER or UNSUPPORTED_ATTRIBUTE	The results of the request to write the MeshIB attribute.
MIBAttribute	Integer	As described in Table 42	The identifier of the MeshIB attribute that was written.

The primitive is generated by the MHME and issued to its next higher layer in response to an MHME-SET.request primitive. This primitive returns a status of either SUCCESS, indicating that the requested value was written to the indicated MeshIB attribute, or an error code of INVALID_PARAMETER or UNSUPPORTED_ATTRIBUTE. The reasons for these status values are fully described in 5.2.2.17.

On receipt of this primitive, the next higher layer is notified of the results of its request to write the value of a MeshIB attribute. If the requested value was written to the indicated MeshIB attribute, the status parameter will be set to SUCCESS. Otherwise, the status parameter indicates the error.

5.2.2.19 MHME-START-SYNC.request

The primitive allows the next higher layer of the mesh coordinator to request the mesh sublayer to start synchronization for synchronous energy saving (SES).

The semantics of the primitive is as follows:

```
MHME-START-SYNC.request (
    AO,
    WO,
    SyncRegion,
    SyncInterval
)
```

Table 27 specifies the parameters of the MHME-START-SYNC.request primitive.

Table 27 —MHME-START-SYNC.request parameters

Name	Type	Valid range	Description
AO	Integer	0–WO or 15	The length of the active portion of the time structure. If the WO has a value of 15, this parameter is ignored. Refer to 5.5.10.2.1 for an explanation of the relationship between the AO and the active duration.
WO	Integer	0–15	The time interval that a device periodically wakeup in energy saving mode. A value of 15 indicates SES will not be used. Refer to 5.5.10.2.1 for an explanation of the relationship between the WO and the wakeup interval.
SyncRegion	Integer	0x00–0xff	The hop distance to be synchronized by a mesh coordinator or region synchronizer during synchronization duration.
SyncInterval	Integer	0x00–0xff	The number of wakeup interval times for periodic synchronization.

This primitive is generated by the next higher layer of the mesh coordinator and issued to its MHME to request the synchronization procedure for SES.

On receipt of this primitive by a device that is not capable of being a mesh coordinator (MC) (by checking the mesh sublayer constant, *meshcCoordinatorCapability*), the MHME issues the MHME-START-SYNC.confirm primitive with the status parameter set to INVALID_REQUEST. Otherwise, the MHME of the mesh coordinator should attempt to start the synchronization procedure by sending a sync request command frame with the parameters AO, WO, SyncRegion, SyncInterval and timestamp in the frame.

The mesh coordinator should transmit two successive synchronization request command frames with timestamp parameters in the frames.

If the MHME of mesh coordinator receives a synchronization reply frame from each of its child devices, it should issue a MHME-START-SYNC.confirm primitive to the next higher layer and start to operate in SES mode from the following wakeup interval.

5.2.2.20 MHME-START-SYNC.confirm

This primitive reports to the next higher layer the results of the synchronization procedure.

The semantics of the primitive is as follows:

```
MHME-START-SYNC.confirm (
    Status
)
```

Table 28 specifies the parameters of the MHME-START-SYNC.confirm primitive.

Table 28—MHME-START-SYNC.confirm parameters

Name	Type	Valid range	Description
Status	Enumeration	INVALID_REQUEST, SYNC_SUCCESS, SYNC_FAILURE	The results of the attempt to start synchronization procedure using synchronous energy saving scheme.

This primitive is generated by the MHME and issued to its next higher layer in response to an MHME-START-SYNC.request primitive. It is sent when the MHME of mesh coordinator receives a synchronization reply frame from each of its child devices. The primitive returns a status value of INVALID_REQUEST, SYNC_SUCCESS or SYNC_FAILURE depending on the status of the initialization of the synchronization procedure.

5.2.2.21 MHME-TRACE-ROUTE.request

This primitive allows the next higher layer of a mesh device to request the mesh sublayer to trace the route to a specific destination.

The semantics of the primitive is as follows:

```
MHME-TRACE-ROUTE.request (
    DestAddr,
    ResponseTimeout,
    BatchSize,
    maxTTL
)
```

Table 29 specifies the parameters of the MHME-TRACE-ROUTE.request primitive.

Table 29—MHME-TRACE-ROUTE.request parameters

Name	Type	Valid range	Description
DestAddr	Integer	0x0000–0xffffd	Specifies the mesh address of the destination.
ResponseTimeout	Integer	0–16,777,215	Timeout limitation for trace route reply frames in millisecond.
BatchSize	integer	0–255	Number of trace route request frames to be transmitted for each hop.
maxTTL	Integer	0–255	Maximum number of hops the traceroute request command frame allowed to propagate.

This primitive is generated by the next higher layer of a device and issued to its MHME to trace the route to the destination specified in this primitive.

On receipt of this primitive, the mesh sublayer should issue a traceroute request command frame toward the destination, using the provided parameters.

5.2.2.22 MHME-TRACE-ROUTE.indication

This primitive allows the next higher layer of a mesh device to be notified when a traceroute reply command frame has been successfully received. It returns round trip times between the originator of traceroute request to an intermediate mesh device or the destination device.

The semantics of the primitive is as follows:

```
MHME-TRACE-ROUTE.indication (
    Status,
    SrcAddr,
    RTTStamp,
    HopCount
)
```

Table 30 specifies the parameters of the MHME-TRACE-ROUTE.indication primitive.

Table 30—MHME-TRACE-ROUTE.indication parameters

Name	Type	Valid range	Description
Status	Boolean	TRUE, FALSE	If TRUE, indicates successful reception of traceroute reply frame within ResponseTimeout.
SrcAddr	Integer	0x0000–0xffff7	Specifies the mesh address of the source of the traceroute reply frame.
RTTStamp	Integer	0–65535	Time duration, in milliseconds, between the traceroute requests is sent to the traceroute reply frame is received.
HopCount	Integer	0x00–0x0f	The hop count between the originator of the traceroute reply and this device.

This primitive is generated by the mesh sublayer of a device and issued to its next higher layer.

On receipt of this primitive, the next higher layer is notified the result of current step of trace route.

5.2.2.23 MHME-TRACE-ROUTE.confirm

This primitive is used to report the final result of the MHME-TRACE-ROUTE request.

The semantics of the primitive is as follows:

```
MHME-TRACE-ROUTE.confirm (
    DestAddr,
    Status
)
```